SIEMENS

SIMATIC HMI

WinCC V7.2
WinCC: Scripting (VBS, ANSI-C, VBA)

Systemhandbuch

Ausdruck der Online-Hilfe

02/2013
A5E32315919-AA

Rechtliche Hinweise

Warnhinweiskonzept

Dieses Handbuch enthält Hinweise, die Sie zu Ihrer persönlichen Sicherheit sowie zur Vermeidung von Sachschäden beachten müssen. Die Hinweise zu Ihrer persönlichen Sicherheit sind durch ein Warndreieck hervorgehoben, Hinweise zu alleinigen Sachschäden stehen ohne Warndreieck. Je nach Gefährdungsstufe werden die Warnhinweise in abnehmender Reihenfolge wie folgt dargestellt.

⚠ GEFAHR
bedeutet, dass Tod oder schwere Körperverletzung eintreten **wird**, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

⚠ WARNUNG
bedeutet, dass Tod oder schwere Körperverletzung eintreten **kann**, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

⚠ VORSICHT
bedeutet, dass eine leichte Körperverletzung eintreten kann, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

ACHTUNG
bedeutet, dass Sachschaden eintreten kann, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

Beim Auftreten mehrerer Gefährdungsstufen wird immer der Warnhinweis zur jeweils höchsten Stufe verwendet. Wenn in einem Warnhinweis mit dem Warndreieck vor Personenschäden gewarnt wird, dann kann im selben Warnhinweis zusätzlich eine Warnung vor Sachschäden angefügt sein.

Qualifiziertes Personal

Das zu dieser Dokumentation zugehörige Produkt/System darf nur von für die jeweilige Aufgabenstellung **qualifiziertem Personal** gehandhabt werden unter Beachtung der für die jeweilige Aufgabenstellung zugehörigen Dokumentation, insbesondere der darin enthaltenen Sicherheits- und Warnhinweise. Qualifiziertes Personal ist auf Grund seiner Ausbildung und Erfahrung befähigt, im Umgang mit diesen Produkten/Systemen Risiken zu erkennen und mögliche Gefährdungen zu vermeiden.

Bestimmungsgemäßer Gebrauch von Siemens-Produkten

Beachten Sie Folgendes:

⚠ WARNUNG
Siemens-Produkte dürfen nur für die im Katalog und in der zugehörigen technischen Dokumentation vorgesehenen Einsatzfälle verwendet werden. Falls Fremdprodukte und -komponenten zum Einsatz kommen, müssen diese von Siemens empfohlen bzw. zugelassen sein. Der einwandfreie und sichere Betrieb der Produkte setzt sachgemäßen Transport, sachgemäße Lagerung, Aufstellung, Montage, Installation, Inbetriebnahme, Bedienung und Instandhaltung voraus. Die zulässigen Umgebungsbedingungen müssen eingehalten werden. Hinweise in den zugehörigen Dokumentationen müssen beachtet werden.

Marken

Alle mit dem Schutzrechtsvermerk ® gekennzeichneten Bezeichnungen sind eingetragene Marken der Siemens AG. Die übrigen Bezeichnungen in dieser Schrift können Marken sein, deren Benutzung durch Dritte für deren Zwecke die Rechte der Inhaber verletzen kann.

Haftungsausschluss

Wir haben den Inhalt der Druckschrift auf Übereinstimmung mit der beschriebenen Hard- und Software geprüft. Dennoch können Abweichungen nicht ausgeschlossen werden, so dass wir für die vollständige Übereinstimmung keine Gewähr übernehmen. Die Angaben in dieser Druckschrift werden regelmäßig überprüft, notwendige Korrekturen sind in den nachfolgenden Auflagen enthalten.

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
DEUTSCHLAND

A5E32315919-AA
℗ 02/2013 Änderungen vorbehalten

Inhaltsverzeichnis

VBS zum Erstellen von Prozeduren und Aktionen 1

1.1 VBS zum Erstellen von Prozeduren und Aktionen

Inhalt

WinCC bietet Ihnen die Möglichkeit, Ihre Runtime-Umgebung mit Visual Basic Script zu dynamisieren. Sie können mit VBS sowohl globale Aktionen und Prozeduren programmieren, als auch Grafikobjekte in Runtime dynamisieren und Aktionen auslösen.

Dieses Kapitel zeigt Ihnen,

- wie Sie mit den VBScript-Editoren arbeiten
- wie Sie Prozeduren erstellen und bearbeiten
- wie Sie Aktionen erstellen und bearbeiten
- wie Sie VB-Skripte in Runtime aktivieren
- wie Sie eine Diagnose Ihrer Skripte in Runtime durchführen
- das Objektmodell des grafischen Runtime-Systems
- ausführliche Beispiele zur Anwendung von VBScript

1.2 Einsatz von Visual Basic Script in WinCC

Einleitung

In WinCC steht Ihnen zusätzlich zu C-Script die Programmiersprache VBScript als Programmierschnittstelle zur Verfügung, um die WinCC Runtime-Umgebung zu dynamisieren.

Zielgruppe der Dokumentation

Diese Dokumentation richtet sich an Projekteure mit Kenntnissen in Visual Basic oder des bisherigen WinCC Scriptings (C).

Einsatzmöglichkeiten

Mit VBScript (VBS) haben Sie in Runtime Zugriff auf Variablen und Objekte des grafischen Runtimesystems und können bildunabhängige Aktionen ausführen:

- Variablen: Sie können Variablenwerte lesen und schreiben, um z.B. per Mausbedienung an einem Button Variablenwerte für die Steuerung vorzugeben.
- Objekte: Sie können Objekteigenschaften mit Aktionen dynamisieren und Aktionen über Ereignisse an Objekten auslösen.
- Bildunabhängige Aktionen: Sie können bildunabhängige Aktionen zyklisch oder von Variablenwerten gesteuert auslösen, z.B. die tägliche Übertragung von Werten in eine Excel-Tabelle.

Sie können VBS an folgenden Stellen in WinCC einsetzen:

- Im Global Script Editor: Hier projektieren Sie bildunabhängige Aktionen und Prozeduren. Die Prozeduren können in bildabhängigen und bildunabhängigen Aktionen verwendet werden. Mehrere Prozeduren werden in einem Modul thematisch zusammengefasst.
- Im Graphics Designer: Hier projektieren Sie bildabhängige Aktionen, mit denen Sie Eigenschaften von Grafikobjekten dynamisieren oder auf Ereignisse in Runtime reagieren können.
- In benutzerdefinierten Menüs und Symbolleisten: Hier projektieren Sie Prozeduren, die Sie mit Hilfe der Menü und Symbolleisten in Runtime aufrufen.

Hinweis

Geänderte Konfiguration in Runtime aktualisieren

Ein geändertes VB-Skript, das mit "Menüs und Symbolleisten" verbunden ist, wird erst nach einem Neustart von Runtime aktualisiert.

Wenn Sie die Eigenschaften von "Menüs und Symbolleisten" in Runtime ändern, werden die Änderungen erst in folgenden Fällen übernommen:

- Nach einem Bildwechsel, wenn die Konfigurationsänderung nicht das Grundbild betrifft.
- Durch das Laden einer anderen Konfigurationsdatei und das erneute Laden der geänderten Konfigurationsdatei.

Angemeldetete Variablen in Menüs und Symbolleisten

Die angemeldeten Variablen in den Skripten von "Menüs und Symbolleisten" bleiben bei Bildabwahl angemeldet. Beim indirekten Lesen aus dem Prozessabbild werden die Variablen angemeldet und nach Bildabwahl wieder abgemeldet. In den Skripten von "Menüs und Symbolleisten" bleiben bei Bildabwahl die Variablen jedoch angemeldet.

Anwendungs-Szenarien

Mit VBS können Sie in Runtime z.B.:

- Sollwertvorgaben für Variablen an die Bedienung eines Grafikobjektes projektieren, um z.B. per Mausklick einen Wert für die Steuerung vorzugeben.
- Die Umschaltung der Runtime-Sprache an die Bedienung eines Grafikobjektes projektieren.
- Farbumschläge projektieren, z.B. zyklisch (blinken) oder zur Darstellung von Zuständen (Motor ein).

Neben den WinCC-spezifischen Anwendungen können Sie auch die allgemeine VBS-Funktionalität zur Anpassung Ihrer Windows-Umgebung nutzen, z.B.:

- Daten in andere Anwendungen übertragen (z.B. Excel).
- Externe Applikationen aus WinCC heraus starten.
- Dateien und Ordner erstellen.

Zur Anpassung Ihrer Windows-Umgebung stehen Ihnen die Automation-Objekte Ihrer Umgebung zur Verfügung.

Hinweis

Sie können alle Objekte, die mit dem Windows Script Host (WSH) von Microsoft ausgeliefert werden, in Ihre Umgebung über die Standard-VBS-Methode CreateObject einbinden. Sie haben jedoch mit VBS aus WinCC heraus keinen direkten Zugriff auf das WSH-Objekt selbst.

Für die VBS-Funktionalität zur Anpassung der Windows-Umgebung kann keine Gewährleistung und kein WinCC-Support gegeben werden.

Abgrenzung zu anderen Programmier-Sprachen in WinCC

VBS und C

Sie können VBScript in WinCC parallel zu C-Script verwenden, die Skript-Arten aber nicht mischen:

- Innerhalb eines Bildes und eines Projektes können Sie VB-Skripte und C-Skripte projektieren.
- Sie können C-Skripte nicht in VB-Skripten aufrufen und umgekehrt.
- In VBS stehen Ihnen interne Schnittstellen zu Variablen und Bildobjekten zur Verfügung, während Sie im C-Umfeld auch auf andere Subsysteme von WinCC (z.B. das Reportsystem) zugreifen können.

VBS und VBA

VBA verwenden Sie in WinCC Configuration bei der Projektierung, um den Graphics Designer Ihren individuellen Bedürfnissen anzupassen und die Projektierung zu vereinfachen und zu automatisieren. VBA-Programme laufen nur in der Projektierungsumgebung von WinCC.

Im Gegensatz zu VBA laufen VB-Skripte nur in WinCC Runtime ab und ermöglichen Ihnen dort den Zugriff auf grafische Objekte und Variablen. Sie können mit VBS im Gegensatz zu VBA Objekte und Bilder weder erzeugen noch dauerhaft ändern.

Wesentliche sprachliche Unterschiede zwischen VBA und VBS sind z.B.:

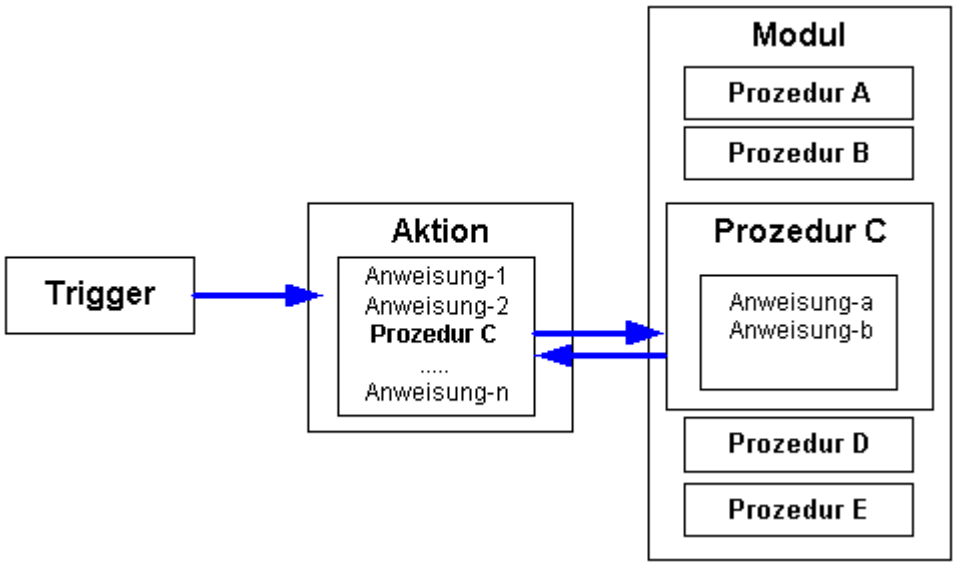
- VBS wurde entwickelt für den Einsatz im Internet, VBA für die Automatisierung von Software-Anwendungen.
- Der Datentyp von VBS-Variablen ist immer VARIANT. VBA hingegen unterscheidet die einzelnen Datentypen wie INT, DOUBLE, STRING etc.
- Bestimmte Sprachkonstrukte von VBA wurden in VBS entfernt bzw. ergänzt.
- Die Fehlerbehandlung in VBS ist anders gelöst als in VBA.

Eine vollständige Auflistung der Unterschiede zwischen VBA und VBS finden Sie im Anhang unter "Grundlagen von VBScript".

Prozeduren, Module und Aktionen

VBS in WinCC ermöglicht Ihnen den Einsatz von Prozeduren, Modulen und Aktionen zur Dynamisierung Ihrer Runtime-Umgebung:

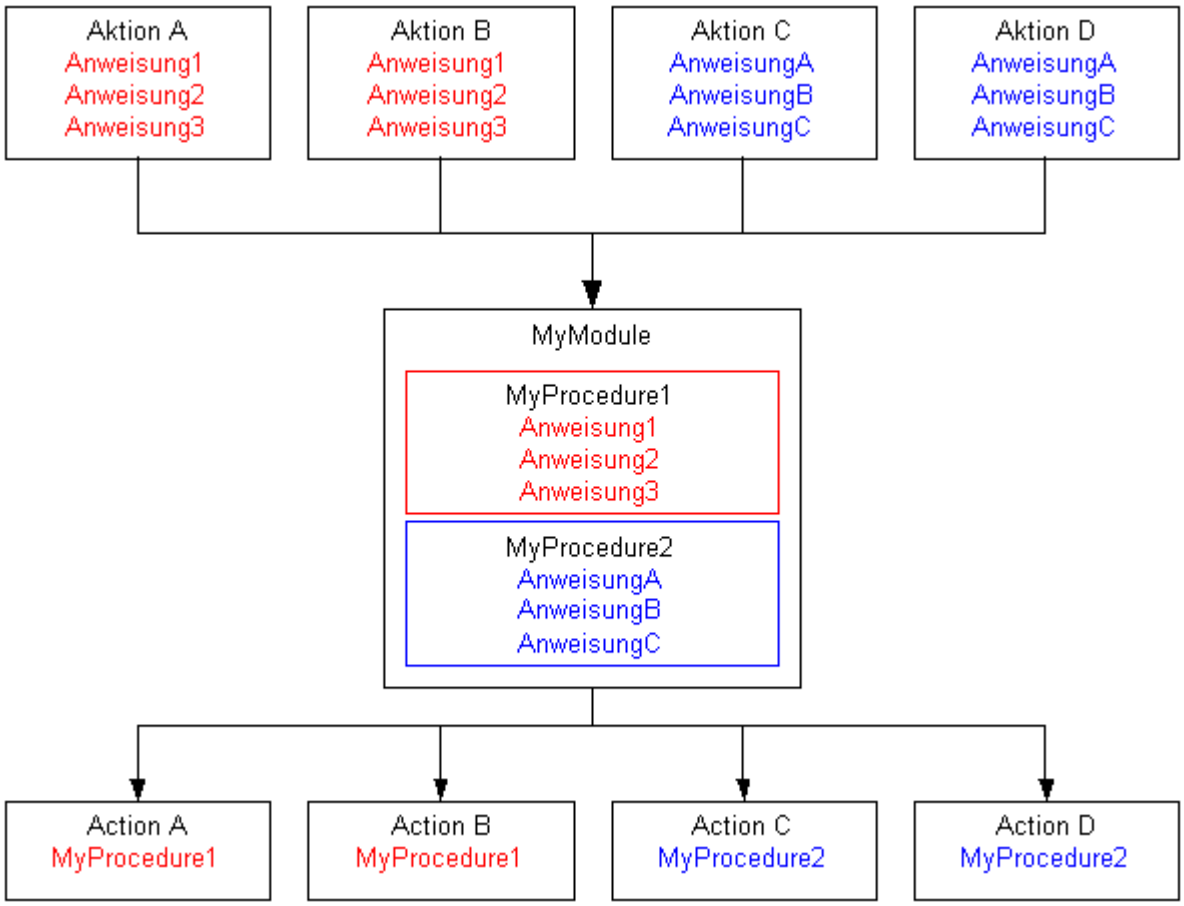
- Prozeduren: Eine Prozedur entspricht einer Funktion in C. In Prozeduren legen Sie Code ab, den Sie an mehreren Stellen in Ihrer Projektierung verwenden möchten. Sie rufen den Code in einer Aktion oder einer anderen Prozedur auf, indem Sie den Prozedurnamen aufrufen. Sie können in WinCC Prozeduren mit oder ohne Rückgabewert erstellen. Prozeduren besitzen keinen eigenen Trigger, der Aufruf erfolgt immer über eine Aktion.
- Module: In Modulen fassen Sie Prozeduren zu sinnvollen Einheiten zusammen. Sie erstellen z.B. Module für Prozeduren, die in einem bestimmten Bild verwendet werden oder zu einer bestimmten Thematik gehören, beispielsweise mathematische Hilfsfunktionen oder Datenbank-Zugriffsfunktionen.
- Aktionen: Aktionen werden immer durch einen Trigger, also durch ein auslösendes Ereignis zum Ablauf gebracht. Aktionen projektieren Sie an Eigenschaften von Grafikobjekten, an Ereignissen, die an einem Grafikobjekt eintreten, oder global im Projekt. In Aktionen können Sie mehrfach verwendeten Code in Form von Prozeduren aufrufen.


Modul
Prozedur A
Prozedur B
Prozedur C
Anweisung-a
Anweisung-b
Prozedur D
Prozedur E
Aktion
Anweisung-1
Anweisung-2
Prozedur C
.....
Anweisung-n
Trigger

1.3 Module und Prozeduren

Einleitung

Prozeduren verwenden Sie, um Code an mehreren Stellen Ihres Projekts verfügbar zu machen, aber nur einmal zu erstellen. Statt den Code mehrfach einzugeben, rufen Sie nur die entsprechende Prozedur auf. Ihr Code ist übersichtlicher und leichter zu pflegen.



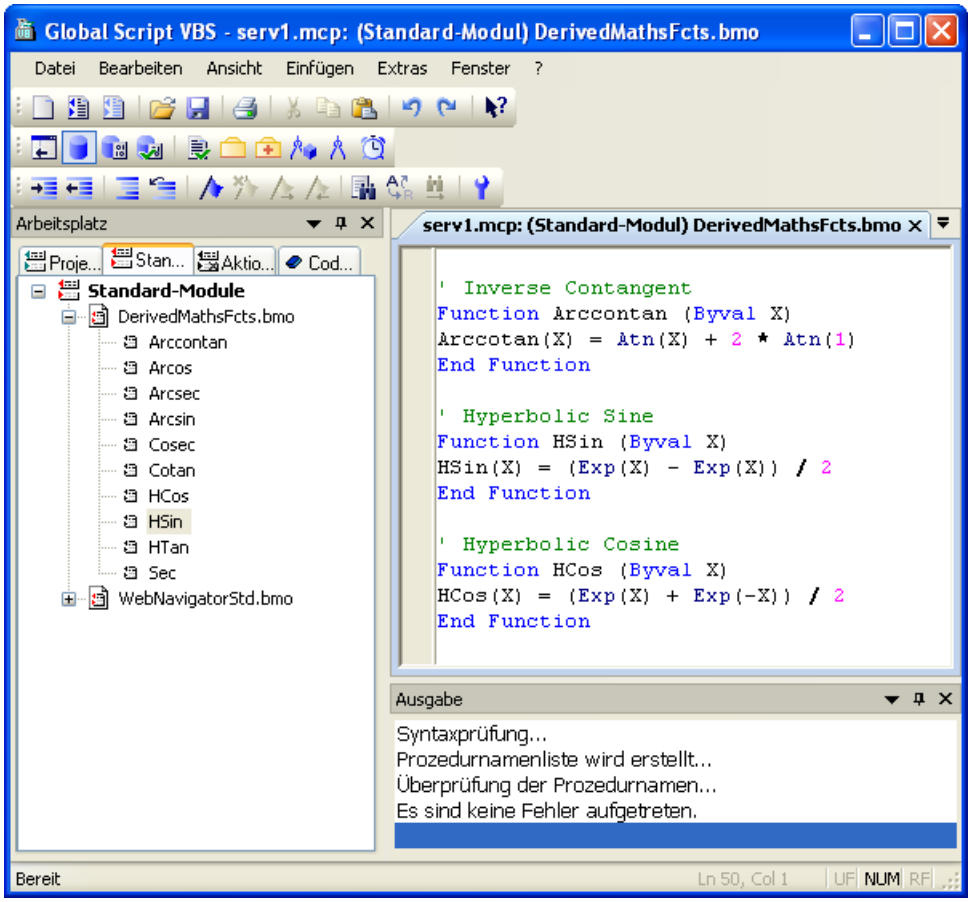
Zusammengehörige Prozeduren speichern Sie immer in Modulen. Wenn Sie eine bestimmte Prozedur über eine Aktion in Runtime aufrufen, wird immer auch das Modul geladen, in dem die Prozedur enthalten ist. Beachten Sie also bei der Strukturierung Ihrer Module und Prozeduren folgendes:

- Je mehr Module beim Aufruf eines Bildes geladen werden müssen, desto schlechter ist die Performance in Runtime.
- Je größer ein Modul ist, also je mehr Prozeduren es enthält, desto länger ist die Ladezeit für das Modul.

Gliedern Sie Ihre Module also sinnvoll, z.B. ein Modul mit Prozeduren für ein bestimmtes Anlagenteil/Bild.

Eine andere Art der Strukturierung von Prozeduren in Modulen ist die funktionale Strukturierung, z.B. ein Modul mit mathematischen Funktionen. Verwenden Sie diese Strukturierung z.B. für Module, die Sie projektübergreifend verwenden möchten. Das folgende

Beispiel zeigt ein Modul, in dem aus den Standardfunktionen abgeleitete mathematische Funktionen abgelegt sind:



Merkmale von Prozeduren

Prozeduren in WinCC haben folgende Eigenschaften:

- Sie werden von Ihnen selbst erstellt und geändert.
- Sie können gegen Änderung und Einsicht mit einem Passwort geschützt werden.
- Sie besitzen keinen Trigger.
- Sie sind in einem Modul gespeichert.

WinCC stellt keine vordefinierten Prozeduren zur Verfügung, bietet Ihnen aber z.B. Codevorlagen und Intellisense zur Vereinfachung der Programmierung. Prozeduren unterscheiden sich je nach Modulzugehörigkeit in:

- Standard-Prozeduren sind projektübergreifend in allen Projekten auf dem Rechner bekannt, auf dem sie erstellt wurden.
- Projekt-Prozeduren können nur innerhalb des Projekts verwendet werden, in dem sie erstellt wurden.

Merkmale von Modulen

Ein Modul ist eine Datei, in der eine oder mehrere Prozeduren gespeichert werden. Module in WinCC haben folgende Eigenschaften:

- Sie können gegen Änderung und Einsicht mit einem Passwort geschützt werden.
- Sie haben die Dateiendung *.bmo.

Module unterscheiden sich je nach der Gültigkeit Ihrer Prozeduren in:

- Standard-Module: enthalten Prozeduren, die projektübergreifend zur Verfügung stehen. Standard-Module sind im WinCC-Dateisystem abgelegt unter: <WinCC-Installationsverzeichnis>\ApLib\ScriptLibStd\<Modulname>.bmo
- Projekt-Module: enthalten projektspezifische Prozeduren. Projekt-Module sind im WinCC-Dateisystem abgelegt unter: <Projektverzeichnis>\ScriptLib\<Modulname>.bmo. Da die Projekt-Module im Projektverzeichnis abgelegt sind, werden sie beim Kopieren eines WinCC-Projekts mit kopiert.

Hinweis

Wenn Sie WinCC neu installieren, und die Standard-Prozeduren und -Module wieder verwenden möchten, sichern Sie die Modul-Dateien vor der Neu-Installation in einem anderen Verzeichnis, und kopieren sie nach der Installation wieder in das entsprechende WinCC-Verzeichnis. Andernfalls werden die Standard-Module im WinCC-Installationsverzeichnis bei der Installation gelöscht.

Verwendung von Prozeduren und Modulen

Prozeduren werden verwendet in:

- Aktionen (im Graphics Designer und in Global Script)
- anderen Prozeduren (in Global Script)
- Benutzerdefinierte Menüs und Symbolleisten

In Modulen strukturieren Sie Prozeduren.

Siehe auch

Prozeduren erstellen und bearbeiten (Seite 39)

Die VBScript-Editoren (Seite 28)

Grundlagen von VBScript (Seite 829)

Aktionen (Seite 19)

Einsatz von Visual Basic Script in WinCC (Seite 12)

1.4 Aktionen

Einleitung

Eine Aktion wird immer durch einen Trigger ausgelöst. In Runtime wird eine Aktion z.B. ausgeführt, wenn ein Objekt per Mausklick bedient wird, ein bestimmter Zeitpunkt eingetreten ist oder ein Variablenwert sich geändert hat.

Merkmale von Aktionen

Aktionen in Global Script werden einmal definiert und sind dann bildunabhängig verfügbar. Global Script Aktionen sind nur in dem Projekt gültig, in dem sie definiert wurden. Aktionen, die mit Grafikobjekten verbunden sind, sind nur in dem Bild gültig, in dem sie definiert wurden.

Hinweis

In VBS ist derzeit die Erstellung von rechnerspezifischen Aktionen nicht möglich.

Für Clients in einem Mehrplatz-System gilt: Alle auf einem Server projektierten globalen Aktionen werden beim Öffnen eines Projektes auf einem Client ebenfalls ausgeführt.
Für Clients in einem Verteilten System gilt: Wenn Sie Aktionen auf einem Client-Rechner verwenden möchten, kopieren Sie die Aktions-Dateien in das entsprechende Projektverzeichnis auf dem Client.

Aktionen haben folgende Eigenschaften:

- Aktionen werden von Ihnen erstellt und geändert.
- Aktionen im Global Script können gegen Änderungen und Einsicht mit einem Passwort geschützt werden.
- Aktionen besitzen mindestens einen Trigger.
- Aktionen im Global Script besitzen die Dateiendung *.bac.
- Aktionen des Global Script sind im WinCC-Dateisystem abgelegt unter: <Projektverzeichnis>\ScriptAct\Aktionsname.bac

Trigger für Aktionen

Trigger sind notwendig, um Aktionen in Runtime auszuführen. Ein Trigger wird mit einer Aktion verbunden und bildet das auslösende Ereignis für den Aufruf der Aktion. Aktionen ohne Trigger werden nicht ausgeführt.

In WinCC stehen Ihnen folgende Triggerarten zur Verfügung:

- Timer: Azyklische oder zyklische Trigger, z.B. bei Aufruf eines Bildes oder jede Stunde.
- Variablen: Wertänderung
- Ereignisse: Änderung von Objekteigenschaften (z.B. Farbwechsel) oder Ereignisse an einem Objekt (z.B. Mausklick).

Abarbeitung von Aktionen in Runtime

Im Graphics Designer

In Runtime können keine zwei Aktionen vom gleichen Typ gleichzeitig ausgeführt werden. Damit z.B. zyklische Aktionen nicht durch eine Aktion behindert wird, die auf Mausklick ausgeführt wird, werden ereignisgetriggerte Aktionen und zyklische/variablengetriggerte Aktionen im Graphics Designer unabhängig voneinander ausgeführt.

Hinweis

Beachten Sie, dass eine Synchronisation zwischen den beiden Aktionsarten in WinCC nur durch das DataSet-Objekt oder interne WinCC-Variablen erfolgen kann. Durch die getrennte Abarbeitung besteht kein gemeinsamer Datenbereich zwischen ereignisgetriggerten und zyklischen/variablengetriggerten Aktionen.

Wenn die Abarbeitung zyklischer Aktionen in Bildern z.B. durch hohe Systemlast oder andere Aktionen behindert wird, wird die Aktion bei der nächstmöglichen Gelegenheit wieder einmalig ausgeführt. Die nicht ausgeführten Zyklen werden nicht in einer Warteschleife gehalten, sondern verworfen.

Nach einem Bildwechsel noch laufende Skripte werden 1 Minute nach dem Bildwechsel automatisch beendet.

Skripte, die beim Beenden von Runtime noch laufen, werden nach 5 Sekunden beendet.

In Global Script

Bildunabhängige Aktionen aus Global Script werden in Runtime bei Auslösung nacheinander ausgeführt. Wird eine Aktion angestoßen, während gerade eine andere Aktion läuft, wird die zweite Aktion in einer Warteschleife gehalten, bis die Ausführung möglich ist.

Hinweis

Beachten Sie, dass eine Synchronisation zwischen Aktionen im Global Script und im Graphics Designer nur durch das DataSet-Objekt oder interne WinCC-Variablen erfolgen kann. Es besteht kein gemeinsamer Datenbereich zwischen Aktionen im Graphics Designer und in Global Script.

Verwendung von Aktionen

Aktionen können wie folgt eingesetzt werden:

- In Global Script: Die hier definierten globalen Aktionen laufen in Runtime bildunabhängig ab.
- Im Graphics Designer: Die hier definierten Aktionen laufen nur im projektierten Bild ab. Eine Aktion im Graphics Designer projektieren Sie immer an eine Objekteigenschaft oder ein Ereignis an einem Grafikobjekt.

Siehe auch

Aktionen erstellen und bearbeiten (Seite 56)
Grundlagen von VBScript (Seite 829)
Module und Prozeduren (Seite 16)
Einsatz von Visual Basic Script in WinCC (Seite 12)

1.5 So verwenden Sie Prozeduren und Aktionen mehrfach

Einleitung

Eine mit VBS in WinCC projektierte Aktion gilt immer für das Projekt, in dem sie definiert wurde.

Prozeduren haben folgende Geltungsbereiche:

- Standard-Prozeduren sind projektübergreifend in allen Projekten auf dem Rechner bekannt, auf dem sie erstellt wurden.
- Projekt-Prozeduren können nur innerhalb des Projektes verwendet werden, in dem sie erstellt wurden. Wenn Sie ein Projekt kopieren, werden die Projekt-Prozeduren (-Module) mit dem Projekt kopiert.

Prozeduren und Aktionen mehrfach verwenden

Wenn Sie Aktionen oder Prozeduren/Module in anderen Projekten oder auf anderen Rechnern verwenden möchten, können Sie entweder:

- Die Funktion "Speichern unter" verwenden, um die Aktion oder das Modul in einem anderen Projektverzeichnis oder z.B. auf einer Diskette zu speichern.
- Die Datei der Aktion oder des Moduls im Windows-Explorer kopieren und in das entsprechende Projekt- bzw Standardverzeichnis auf dem Zielrechner kopieren.

Die projektierten Eigenschaften und Trigger bleiben beim Kopieren erhalten. Kopierte Module stehen in Runtime direkt zur Verfügung. Kopierte Aktionen werden in Runtime ausgeführt, nachdem sie einmal geöffnet und gespeichert wurden.

Hinweis

Variablen, die Sie in einer Aktion oder Prozedur verwenden, müssen auf dem Zielrechner ebenfalls vorhanden sein. Ist die Variable nicht vorhanden, wird die Aktion oder Prozedur nicht ausgeführt.

Prozeduren, die Sie in einer Aktion aufrufen, müssen auf dem Zielrechner vorhanden sein. Ist die Prozedur nicht vorhanden, kommt es zu einem Laufzeitfehler in Runtime.

Ablage von Prozeduren

Wenn Sie Prozeduren in andere Projektverzeichnisse kopieren möchten, um sie in anderen Projekten oder auf anderen Rechnern zu verwenden, beachten Sie den Ablagepfad der Prozeduren innerhalb der WinCC-Dateisystems:

- Standard-Prozeduren: <WinCC-Installationsverzeichnis>\ApLib\ScriptLibStd \Modulname.bmo
- Projekt-Prozeduren: <Projektverzeichnis>\ScriptLib\Modulname.bmo

Hinweis

Da Prozeduren immer in Modulen gespeichert werden, kopieren Sie immer das Modul (*.bmo), in dem die Prozedur enthalten ist.

Die kopierten Prozeduren/Module werden nach Aktualisierung des Navigationsfensters von Global Script sichtbar (Kontextmenübefehl "Aktualisieren") oder nach Neustart des Editors.

Ablage von Aktionen

Wenn Sie Aktionen in andere Projektverzeichnisse kopieren möchten, um sie z.B. in anderen Projekten oder auf anderen Rechnern zu verwenden, beachten Sie den Ablagepfad der Aktionen innerhalb der WinCC-Dateisystems:

<Projektverzeichnis>\ScriptAct\Aktionsname.bac

Jede Aktion wird in einer eigenen Datei gespeichert. Beim Kopieren von Aktionen werden auch alle zur Aktion gehörigen Trigger mit kopiert.

Hinweis

Nur Aktionen, die im Global Script erstellt wurden, sind im WinCC-Dateisystem abgelegt. Aktionen, die Sie im Graphics Designer programmieren, werden immer mit dem aktuellen Bild gespeichert und sind nicht einzeln übertragbar. Wenn Sie ein Bild des Graphics Designer in ein anderes Projektverzeichnis kopieren, werden die mit dem Bild gespeicherten Aktionen aber mit übertragen.

Die kopierten Aktionen werden nach Aktualisierung des Navigationsfensters von Global Script sichtbar (Kontextmenübefehl "Aktualisieren") oder nach Neustart des Editors.

Siehe auch

Module und Prozeduren (Seite 16)

So benennen Sie eine Prozedur oder ein Modul um (Seite 54)

So speichern Sie eine Prozedur (Seite 52)

So schützen Sie ein Modul mit einem Passwort (Seite 51)

So fügen Sie modulbegleitende Informationen hinzu (Seite 49)

So verwenden Sie Standard- und Projektprozeduren (Seite 48)

So schreiben Sie den Prozedurcode (Seite 45)

So legen Sie eine neue Prozedur an (Seite 43)

Prozeduren erstellen und bearbeiten (Seite 39)

1.6 Zusammenhänge mit der CrossReference

CrossReference und Variablentrigger

Mit der CrossReference von WinCC können Sie alle Verwendungsstellen von Variablen auch in VBS-Aktionen schnell finden. Variablentrigger aus Aktionen im Graphics Designer können Sie mit CrossReference "umverdrahten", also an allen oder an ausgewählten Stellen durch andere Variablen ersetzen.

Hinweis

Sie können Variablen auch direkt im Graphics Designer umverdrahten, indem Sie das Grafikobjekt markieren und aus dem Kontextmenü den Befehl "Umverdrahten..." wählen.

Weiterführende Hinweise erhalten Sie in der WinCC-Dokumentation zu CrossReference.

Aktionen und CrossReference

Sie können sich alle in einem Bild verwendeten Aktionen über die Eigenschaften des Bildes anzeigen lassen. Markieren Sie dazu das Bild im WinCCExplorer und wählen Sie den Kontextmenübefehl "Eigenschaften". Durch Doppelklick auf einen Eintrag erhalten Sie detaillierte Informationen zur Art der Dynamisierung.

Sie können sich außerdem alle in Aktionen verwendeten Variablen und Bilder über die WinCC CrossReference anzeigen lassen. Mit der CrossReference können Sie auch Variablenanbindungen von Aktionen des Graphics Designer bequem umverdrahten.

Variablen und CrossReference

Alle Variablen, die Sie mit der folgenden Standardformulierung adressieren, werden durch die CrossReference von WinCC automatisch erfasst und sind in den Bildeigenschaften aufgeführt.

```
' VBS1
HMIRuntime.Tags("Tagname")
```

Wenn Sie Variablen mit anderen Formulierungen in Ihrem Code ansprechen, können Sie diese über folgende Sektion der CrossReference bekanntmachen:

```
' WINCC:TAGNAME_SECTION_START
Const TagNameInAction = "TagName"
' WINCC:TAGNAME_SECTION_END
```

Sie können diese Sektion beliebig oft in VBS-Aktionen einfügen.

Hinweis

Die Erfassung zusammengesetzter Variablennamen von der CrossReference kann nicht garantiert werden.

Bilder und CrossReference

Alle Bilder, die Sie mit der folgenden Standardformulierung adressieren, werden durch die CrossReference von WinCC automatisch erfasst und sind in den Bildeigenschaften aufgeführt.

```
'VBS2
HMIRuntime.BaseScreenName = "Screenname"
```

Wenn Sie Bilder mit anderen Formulierungen in Ihrem Code ansprechen, können Sie diese über folgende Sektion der CrossReference bekanntmachen:

```
' WINCC:SCREENNAME_SECTION_START
Const ScreenNameInAction = "ScreenName"
' WINCC:SCREENNAME_SECTION_END
```

Sie können diese Sektion beliebig oft in VBS-Aktionen einfügen.

Hinweis

Bildnamen sind aus Kompatibilitätsgründen zu zukünftigen Versionen immer ohne die Dateierweiterung ".PDL" zu schreiben.

Siehe auch

VBS Referenz (Seite 113)

Die VBScript-Editoren (Seite 28)

Grundlagen von VBScript (Seite 829)

Aktionen (Seite 19)

Module und Prozeduren (Seite 16)

Einsatz von Visual Basic Script in WinCC (Seite 12)

1.7 Verwendung von globalen Variablen in VBS

Einleitung

Sie können im Global Script Editor globale Variablen definieren, die dann in allen Aktionen und Prozeduren verwendet werden können.

Verwendung von globalen Variablen in Graphics Designer und Global Script

Beachten Sie bei der Verwendung von Variablen im Graphics Designer und im Global Script folgende Randbedingungen:

- Um eine globale Variable in einer Aktion im Graphics Designer zu verwenden, müssen Sie die Prozedur aufrufen, in der die Variable definiert ist, damit das dazugehörige Modul in Runtime geladen wird.
- Um eine globale Variable in einer Aktion im Global Script zu verwenden, müssen Sie in mindestens einer globalen Aktion mindestens eine Prozedur aus dem Modul aufrufen, in dem die Variable definiert ist, damit das Modul in Global Script Runtime geladen wird. Dies muss nicht die Prozedur sein, in dem die Variable definiert wurde.

Diese Vorgehensweise ist notwendig, da Aktionen aus Global Script und aus dem Graphics Designer in Runtime unabhängig voneinander abgearbeitet werden. Es besteht kein gemeinsamer Datenbereich zwischen den beiden Runtime-Systemen.

Wenn Sie Aktionen aus Global Script und aus dem Graphics Designer synchronisieren müssen, verwenden Sie das DataSet-Objekt oder interne WinCC-Variablen.

Verwendung von globalen Variablen innerhalb des Graphics Designer

Wenn Sie globale Variablen Aktionen im Graphics Designer verwenden, beachten Sie dabei folgende Randbedingungen:

- Um eine globale Variable in einer zyklischen oder variablengetriggerten Aktion im Graphics Designer zu verwenden, müssen Sie die Prozedur aufrufen, in der die Variable definiert ist. Dies gilt auch, wenn die Variable bereits in einer ereignisgetriggerten Aktion aufgerufen wurde.
- Um eine globale Variable in einer ereignisgetriggerten Aktion im Graphics Designer zu verwenden, müssen Sie die Prozedur aufrufen, in der die Variable definiert ist. Dies gilt auch, wenn die Variable bereits in einer zyklischen oder variablengetriggerten Aktion aufgerufen wurde.

Diese Vorgehensweise ist notwendig, da zyklische/variablengetriggerte Aktionen und ereignisgetriggerte Aktionen im Graphics Designer in Runtime unabhängig voneinander abgearbeitet werden. Es besteht kein gemeinsamer Datenbereich zwischen den beiden Aktionsarten

Wenn Sie zyklische-/variablengetriggerte Aktionen und ereignisgetriggerte Aktionen synchronisieren müssen, verwenden Sie das DataSet-Objekt oder interne WinCC-Variablen.

Im Graphics Designer ist es zusätzlich möglich, in einem separaten Deklarationsteil globale Variablen zu definieren. Da in Runtime ereignisgetriggerte und zyklische/variablengetriggerte Aktionen getrennt abgearbeitet werden, sind hier die globalen Variablen nur innerhalb der ereignisgetriggerten oder zyklischen/variablengetriggerten Aktionen gemeinsam ansprechbar.

Siehe auch

Grundlagen von VBScript (Seite 829)
Aufbau von VBScript-Dateien (Seite 99)
Aktionen erstellen und bearbeiten (Seite 56)
Prozeduren erstellen und bearbeiten (Seite 39)
Die VBScript-Editoren (Seite 28)
Zusammenhänge mit der CrossReference (Seite 24)
Aktionen (Seite 19)
Module und Prozeduren (Seite 16)

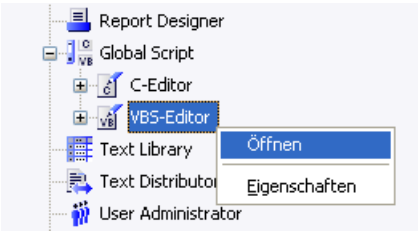
1.8 Die VBScript-Editoren

1.8.1 Die VBScript-Editoren

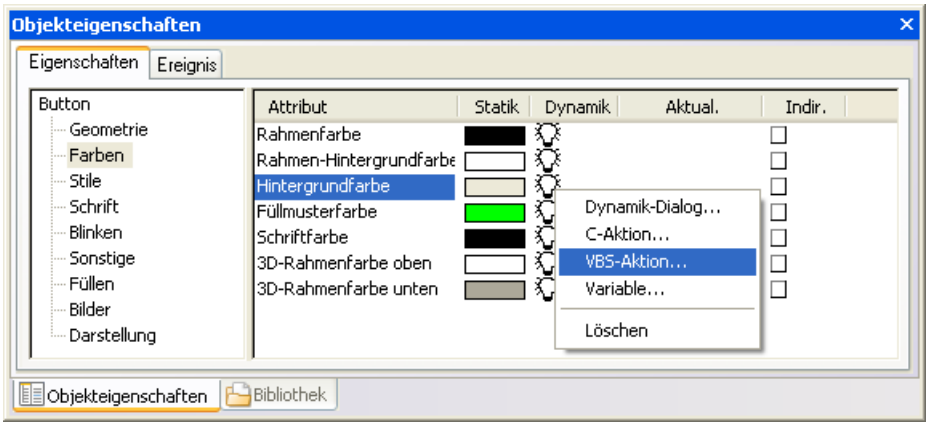
Einleitung

In WinCC können Sie VB-Skripte an zwei Stellen programmieren:

- In Global Script: Global Script ist der zentrale Editor zur VBS-Programmierung. Sie rufen ihn über den WinCCExplorer auf.



- Im Graphics Designer: Im Graphics Designer programmieren Sie Aktionen an Objekteigenschaften oder Ereignisse an Grafikobjekten. Den Aktionseditor im Graphics Designer rufen Sie über das Kontextmenü im Eigenschaftendialog eines Grafikobjektes auf.



Abgrenzung Global Script - Graphics Designer

Im Graphics Designer können Sie Aktionen und bildspezifische Prozeduren programmieren, aber keine projektweit gültigen globalen Prozeduren. Sie können aber globale Prozeduren aufrufen, die in Global Script programmiert wurden.

Hinweis

In dieser Dokumentation wird primär Global Script beschrieben und ggf. auf Abweichungen in der Funktionalität zum Graphics Designer hingewiesen. Eine detaillierte Beschreibung des Aktionseditors im Graphics Designer finden Sie unter dem WinCC-Hilfethema "Dynamisierung".

Weitere Informationen

Weiterführende Informationen erhalten Sie in der WinCC-Dokumentation "Dynamisierung".

Siehe auch

Der Global Script Editor (Seite 29)

1.8.2 Der Global Script Editor

Einleitung

Globale Prozeduren und Aktionen erstellen und bearbeiten Sie im Global Script Editor.

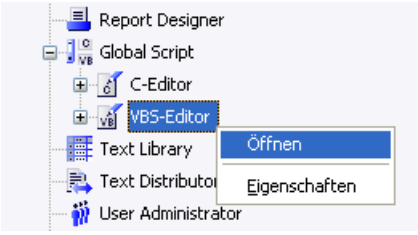
Global Script stellt Ihnen einen ähnlichen Funktionsumfang zur Verfügung wie der C-Skript-Editor in WinCC.

Hinweis

Eine detaillierte Beschreibung des Aktionseditors zur Erstellung von bildbezogenenen Aktionen und Prozeduren im Graphics Designer finden Sie unter dem WinCC-Hilfethema "Dynamisierung".

Aufruf von Global Script

Global Script starten Sie über den Kontextmenübefehl "Öffnen" im Projektfenster des WinCCExplorer.

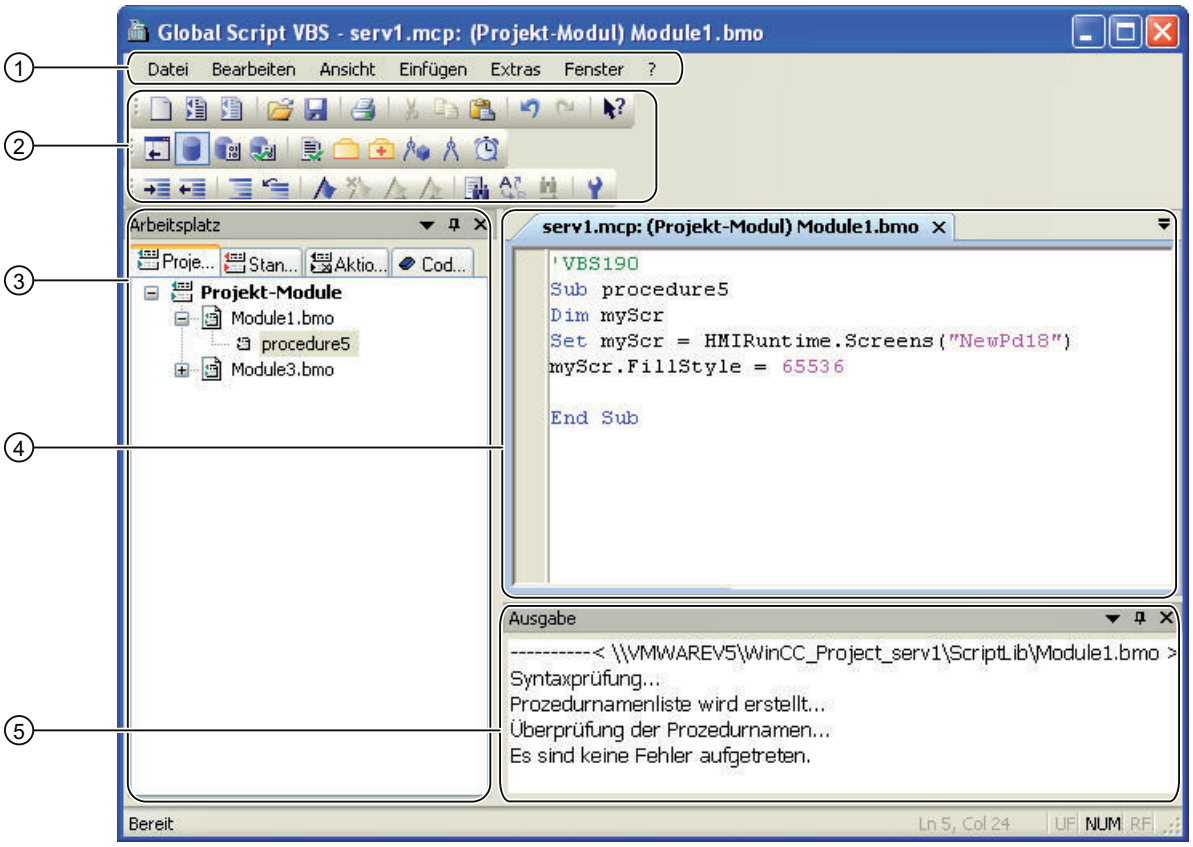


Global Script wird auch automatisch aufgerufen, wenn Sie ein Modul oder eine Aktion per Doppelklick im WinCCExplorer öffnen.

Aufbau von Global Script

Global Script ist nach Windows Standards aufgebaut.

Der Aktionseditor im Graphics Designer stellt Ihnen eine ähnliche Funktionalität zur Verfügung wie Global Script. Eine Beschreibung des Aktionseditors im Graphics Designer finden Sie unter dem WinCC-Hilfethema "Dynamisierung".



Menüleiste (1) und Symbolleisten (2)

In der Menüleiste und den Symbolleisten finden Sie alle Befehle, die Sie zum Erstellen von Prozeduren und Aktionen benötigen.

Die Symbolleisten sind über den Menübefehl "Ansicht" > "Symbolleisten" ein- und ausblendbar und können an jede beliebige Position innerhalb des Editors verschoben werden.

Navigationsfenster (3)

Im Navigationsfenster verwalten Sie Ihre Prozeduren, Module und Aktionen. Außerdem finden Sie hier Codevorlagen, die Sie per drag&drop in Ihre Aktion oder Prozedur einfügen können.

Eine Prozedur können Sie in einer anderen Prozedur oder Aktion aufrufen, in dem Sie sie aus dem Navigationsfenster per drag&drop an die entsprechende Stelle im Code ziehen.

Die Anzeige im Navigationsfenster wird immer erst beim Speichern des bearbeiteten Dokumentes aktualisiert. Wenn Sie eine Datei gerade ändern, wird dies durch einen * hinter dem Dateinamen angezeigt.

Im Navigationsfenster werden unterhalb der Moduldateien die in einem Modul enthaltenen Prozeduren angezeigt. Auf der Registerkarte Aktionen werden zusätzlich die an der Aktion projektierten Trigger und ggf. direkt in einem Aktionsmodul definierte Prozeduren angezeigt.

Sie können im Navigationsfenster weiterhin:

- Unterverzeichnisse zur Strukturierung Ihrer Skripte anlegen.
- Module und Verzeichnisse direkt verschieben, kopieren, einfügen, löschen und umbenennen.

Sie können die Anzeige im Navigationsfenster mit dem Menübefehl "Ansicht" > "Arbeitsplatz" individuell konfigurieren. Hier können Sie wählen, ob alle Dateitypen, nur Script-Dateien oder nur syntaktisch korrekte Script-Dateien angezeigt werden sollen. Mit dem Menübefehl "Ansicht" > "Arbeitsplatz" > "Anzeigen" können Sie das Navigationsfenster ein- oder ausblenden.

Editierfenster (4)

Im Editierfenster schreiben und bearbeiten Sie Ihre Prozeduren und Aktionen. Jede Prozedur oder Aktion wird in einem eigenen Editierfenster geöffnet. Es können mehrere Editierfenster gleichzeitig geöffnet sein.

Im Editierfenster wird der Anwender durch Syntaxhervorhebung und Intellisense unterstützt. Weiterhin sind alle gängige Editorfunktionen (z.B. Undo/Redo, Suchen/Ersetzen, Kopieren, Einfügen, Ausschneiden, Fonteinstellungen, Druckerunterstützung) verfügbar.

Ausgabefenster (5)

Im Ausgabefenster werden Ihnen Fehlermeldungen nach der Syntaxprüfung angezeigt. Mit einem Doppelklick auf die entsprechende Fehlerzeile gelangen Sie in die dazugehörige Stelle im Code.

Statuszeile (6)

Die Statuszeile enthält Informationen zur momentan gewählten Funktionalität oder Tipps zur Programmierung.

Hinweis

Wenn Sie Informationen zu einzelnen Befehlen oder Symbolen des Editors benötigen, wählen Sie den Menübefehl "?" > "Direkthilfe". Klicken Sie dann mit der Maus auf das entsprechende Symbol/den Befehl. Zu allen Bedienelementen des Editors erhalten Sie so eine schnelle Direkthilfe. Wenn Sie den Direkthilfemodus wieder verlassen möchten, drücken Sie "Esc".

Fenster-Docking

Durch Fenster-Docking  haben Sie die Möglichkeit die einzelnen Fenster flexibel anzuordnen. Es steht Ihnen frei die Fenster umpositionieren, freistehend als alleinstehendes Fenster abzulegen oder Fenster in Tabgruppen zu gruppieren. Platzieren Sie beispielsweise Ihre Aktionen nebeneinander, untereinander oder als Tabs. Sie können Fenster automatisch verstecken und bei Bedarf wieder einblendenWeitere Informationen erhalten Sie im Kapitel "Prozessbilder erstellen".

Siehe auch

So löschen Sie Aktionen oder Prozeduren (Seite 38)

Arbeiten mit den Symbolleisten (Seite 35)

Arbeiten im Editierfenster (Seite 32)

1.8.3 Arbeiten im Editierfenster

Einleitung

Im Editierfenster bearbeiten Sie Prozeduren und Aktionen.

Deklarationsbereich in Aktionen (nur Graphics Designer)

Wenn Sie Aktionen im Aktionseditor des Graphics Designer erstellen, können Sie mit der Schaltfläche den Deklarationsbereich der Aktion im Editierfenster einblenden.

Im Deklarationsbereich können Sie zusätzlich allgemeine Festlegungen treffen, die Sie nur für das aktuelle Bild verwenden möchten, z.B.:

- Variablendefinitionen
- Prozeduren, die Sie nur in diesem Bild verwenden möchten

Hinweis

Erstellen Sie keinen direkt ausführbaren Code im Deklarationsbereich!

Beachten Sie, dass beim Anlegen einer Variable diese keinen Wert enthält (Value = VT_EMPTY). Initialisieren Sie Variablen nach der Deklaration mit einem entsprechenden Wert.

Beachten Sie bei Definitionen im Deklarationsbereich den Aufbau der Script-Dateien, wie unter "Aufbau von VBScript-Dateien" beschrieben.

"Option explicit" in Aktionen

Beim Anlegen einer neuen Aktion wird die Anweisung "Option explicit" automatisch und nicht löschbar in den Deklarationsbereich (Graphics Designer) bzw. in die erste Zeile einer Aktion (Global Script) eingetragen. Die Anweisung ist erforderlich um Fehler durch falsche Schreibweise von Variablen ohne Deklaration zu vermeiden. Die Anweisung erfordert, dass Sie Variablen immer mit der Anweisung "Dim" in Ihrem Code definieren.

Hinweis

Verwenden Sie die Anweisung "Option explicit" nicht in Ihrem Code, da es sonst zu Lauzeitfehlern kommen kann.

Anwenderunterstützung im Editierfenster

Sie werden bei Ihrer Arbeit im Editierfenster durch folgende Funktionen unterstützt.

Farbcodierung und Einrückung im Editierfenster

Bestimmte Teile des Codes sind standardmäßig wie folgt farblich gekennzeichnet:

Farbe	Bedeutung	Beispiel
blau	Schlüsselworte Funktionen	Sub, End Sub, Next
grün	Kommentare	' das ist ein Kommentar
rot	Strings (Zeichenketten und Zahlen)	"Object1"
dunkelblau	Präprozessoranweisungen	--
fett schwarz	Konstanten	**vbTrue, vbFalse**
schwarz	sonstiger Code	--

Die Farbcodierung im Editierfenster können Sie über die Editoreinstellungen individuell festlegen. Rufen Sie dazu über den Menübefehl "Extras" > "Optionen" den Dialog "Script Editor Optionen" auf und legen Sie dort die Einstellungen fest.

Um Ihren Code übersichtlich zu gestalten, können Sie den Code durch Einrückungen strukturieren. Im Dialog "Script Editor Optionen" können Sie dazu auch die Tabweite einstellen und das automatische Einrücken während des Schreibens aktivieren.

Intellisense und Syntaxhervorhebung

Während der Eingabe erscheinen kontextabhängige Listen, in denen Ihnen die an der aktuellen Codestelle möglichen Eigenschaften, Methoden und Objekte angeboten werden. Wenn Sie ein Element der Liste einfügen, wird Ihnen automatisch die erforderliche Syntax mit angegeben.

Hinweis

Vollständige Intellisense für alle Objekte ist im Graphics Designer nur nutzbar, wenn über den Objektnamen auf die Auflistung zugegriffen und das Ergebnis einer Variablen zugewiesen wird. Sonst werden Ihnen nur die Standardeigenschaften angeboten.

Bsp. für vollständige Intellisense:

```
Dim Variable
Set Variable = ScreenItems ("Kreis1")
Variable.
```

Wenn bei der Adressierung Bildfenstergrenzen überschritten werden, werden ebenfalls nur die Standardeigenschaften angeboten, da das Bild des Bildfensters nicht geladen ist.

Die Syntaxhervorhebung können Sie im Dialog "Script Editor Optionen" aktivieren und deaktivieren. Den Dialog rufen Sie über den Menübefehl "Extras" > "Optionen" auf.

Allgemeine VBS-Funktionen

Über den Befehl "Funktionsliste" des Kontextmenüs im Editierfenster können Sie sich eine Liste mit allgemeinen VBS-Funktionen anzeigen lassen.

Auflistungen von Objekten, Eigenschaften und Methoden

Über das Kontextmenü im Editierfenster können Sie mit dem Befehl "Objektliste" im Graphics Designer eine Liste der möglichen Objekte anzeigen lassen. Im Global Script erhalten Sie in dieser Liste nur das Objekt "HMIRuntime", da auf die Objekte des Graphics Designer nicht direkt zugegriffen werden kann.

Über den Befehl "Eigenschaften/Methoden" des Kontextmenüs erhalten Sie eine Auflistung der möglichen Eigenschaften und Methoden.

Die gleichen Listen erhalten Sie je nach Kontext im Skript mit der Tastenkombination <Strg +Leertaste>.

Codevorlagen

Im Navigationsfenster des Editors auf der Registerkarte "Codevorlagen" finden Sie eine Auswahl häufig verwendeter Anweisungen, z.B. für Schleifen und bedingte Anweisungen. Sie können diese Vorlagen per "drag&drop" in den Prozedurcode einfügen.

Wenn Sie eine Codevorlage in Ihren Code einfügen, müssen Sie die Platzhalter "_XYZ_" in den Vorlagen durch die entsprechenden Angaben ersetzen.

Auswahldialoge

Wenn Sie WinCC-Variablen oder -Objekte im Code verwenden, stehen Ihnen folgende Auswahldialoge zur Verfügung:

- Öffnet einen Auswahldialog für Variablen und gibt den selektierten Variablennamen als Rückgabewert.
- Öffnet einen Auswahldialog für Variablen und gibt den Variablennamen mit zugehöriger Referenz zurück.
- Öffnet einen Bild/Objektbrowser in dem man ein Bild/Objekt auswählen kann, dessen Name als Rückgabewert geliefert wird.
- Öffnet einen Auswahldialog für Bilder und gibt den Bildnamen ggf. mit Serverpräfix zurück.

Lesezeichen

Sie können in Ihrem Code Lesezeichen setzen, um bestimmte Stellen im Code leichter wiederzufinden:

- Setzt ein Lesezeichen in der Zeile, in der aktuell der Cursor steht.
- Löscht alle Lesezeichen im aktiven Editierfenster.
- Springt im Code ein Lesezeichen weiter vor.
- Springt im Code ein Lesezeichen zurück.

Siehe auch

Aufbau von VBScript-Dateien (Seite 99)

Der Global Script Editor (Seite 29)

1.8.4 Arbeiten mit den Symbolleisten

Zweck

Die Symbolleisten befinden sich in ihrer Standardposition unterhalb der Menüleiste, am oberen Rand des VBS-Editors. Die auf den Symbolleisten angeordneten Tasten erlauben einen schnellen und bequemen Zugriff auf die von Global Script bzw. dem Aktionseditor im Graphics Designer angebotene Funktionalität.

Im Global Script/Graphics Designer stehen Ihnen folgende Symbolleisten zur Verfügung:

Symbolleiste "Standard"

Schaltfläche	Funktion	Tastenkombination
	Erstellt ein neues Projekt-Modul (nur Global Script)	<Alt+F1>
	Erstellt ein neues Standard-Modul (nur Global Script)	<Alt+F2>

Schaltfläche	Funktion	Tastenkombination
	Erstellt eine neue globale Aktion (nur Global Script)	<Alt+F3>
	Öffnet eine vorhandene Aktion oder ein vorhandenes Modul (nur Global Script)	<Strg+O>
	Speichert den Inhalt des aktiven Editierfensters. Die Funktion ist nur dann verfügbar, wenn ein Editierfenster geöffnet ist. Nach dem Speichern wird die Anzeige im Navigationsfenster aktualisiert. (nur Global Script)	<Strg+S>
	Schneidet den markierten Text aus und kopiert ihn in die Zwischenablage. Die Funktion ist nur dann verfügbar, wenn Text markiert ist.	<Strg+X>
	Kopiert den markierten Text in die Zwischenablage. Die Funktion ist nur dann verfügbar, wenn Text markiert ist.	<Strg+C>
	Fügt den Inhalt der Zwischenablage an die Position der Schreibmarke ein. Die Funktion ist nur dann verfügbar, wenn die Zwischenablage nicht leer ist.	<Strg+V>
	Druckt den Inhalt des aktiven Editierfensters in Form einer Projektdokumentation aus. Die Funktion ist nur dann verfügbar, wenn ein Editierfenster geöffnet ist.	<STRG+P>

Inhalt der Symbolleiste "Editor"

Schaltfläche	Funktion	Tastenkombination
	Rückt die Zeile, in welcher der Cursor steht, um eine Position nach rechts ein.	--
	Rückt die Zeile, in welcher der Cursor steht, um eine Position nach links aus.	--
	Markiert die mit der Maus markierten Zeilen als Kommentar. Sind keine Zeilen mit der Maus markiert, wird die Zeile, in welcher der Cursor steht, als Kommentar markiert.	--
	Entfernt die Kommentarmarkierung aus den mit der Maus markierten Zeilen. Sind keine Zeilen mit der Maus markiert, wird die Kommentarmarkierung in der Zeile, in welcher der Cursor steht, entfernt.	--
	Setzt ein Lesezeichen in die aktuelle Zeile. Erneute Betätigung entfernt das Lesezeichen aus der aktuellen Zeile.	<Strg+F9>
	Entfernt alle Lesezeichen aus dem aktuellen Code im Editierfenster.	<Strg+Shift+F9>
	Setzt den Cursor ein Lesezeichen weiter.	<F9>
	Setzt den Cursor ein Lesezeichen zurück.	<Shift+F9>
	Öffnet den "Suchen"-Dialog zur Textsuche im Code.	<Strg+F>
	Öffnet den "Ersetzen"-Dialog zum Suchen und Ersetzen von Text im Code.	<Strg+H>
	Wiederholt den Suchvorgang.	<F3>
	Öffnet den Dialog "Skript Editor Optionen".	--

Schaltfläche	Funktion	Tastenkombination
	Macht die letzte von maximal 30 Editieraktionen rückgängig. Die Funktion ist nur dann verfügbar, wenn eine Editieraktion ausgeführt wurde	<Strg+Z>
	Stellt die letzte rückgängig gemachte Editieraktionen wieder her. Die Funktion ist nur dann verfügbar, wenn eine Editieraktion rückgängig gemacht worden ist.	<Strg+Y>

Inhalt der Symbolleiste "Bearbeiten"

Schaltfläche	Funktion	Tastenkombination
Arbeitsplatz		
	Selektiert die Datei im Navigationsfenster, zu der das aktuelle Editierfenster gehört (nur Global Script).	--
	Zeigt alle Dateien im Navigationsfenster an (nur Global Script).	--
	Zeigt nur Skript-Dateien im Navigationsfenster an (nur Global Script).	--
	Zeigt nur syntaktisch korrekte Skript-Dateien im Navigationsfenster an (nur Global Script).	--
Script		
	Führt die Syntaxprüfung im Code des aktuellen Editierfensters durch.	<F7>
WinCC Objekte		
	Öffnet den Dialog "Info/Trigger".	<Strg+T>
	Zeigt den Namen des Triggers an.	
	Öffnet einen Auswahldialog für Variablen und gibt den selektierten Variablennamen als Rückgabewert zurück.	<Strg+U>
	Öffnet einen Auswahldialog für Variablen und gibt den Variablennamen mit zugehöriger Referenz zurück.	<Strg+W>
	Öffnet einen Bild/Objektbrowser in dem man ein Bild/Objekt auswählen kann, dessen Name als Rückgabewert geliefert wird.	<Strg+Q>
	Öffnet einen Auswahldialog für Bilder und gibt den Bildnamen ggf. mit Serverpräfix zurück.	<Strg+B>

Zusätzliche Schaltflächen im Graphics Designer

Zusätzlich zu den Schaltflächen des Global Script verfügt der Aktionseditor im Graphics Designer über folgende Schaltflächen:

-  Einblenden des Deklarationsbereiches (<Strg+E>)
-  Ausblenden des Deklarationsbereiches (<Strg+A>)

Siehe auch

Der Global Script Editor (Seite 29)

1.8.5 So löschen Sie Aktionen oder Prozeduren

Einleitung

Wenn Sie eine Aktion, Prozedur oder ein Modul in einem der Script-Editoren löschen, werden der Code und die entsprechende Datei im Projektverzeichnis gelöscht.

Achten Sie darauf, nur Prozeduren zu löschen, die in keiner anderen Prozedur oder Aktion mehr verwendet werden. Ruft eine Aktion eine nicht mehr vorhandene Prozedur auf, wird die Aktion in Runtime an der fehlerhaften Stelle abgebrochen. Eine nicht vorhandene Referenz im Code kann von der Syntaxprüfung nicht erkannt werden.

Hinweis

Prozeduren können Sie nur innerhalb eines Modules löschen, indem Sie den Code löschen, nicht im Navigationsfenster des Editors.

Vorgehensweise

1. Öffnen Sie Global Script.
2. Markieren Sie die Aktion oder das Modul, das Sie löschen möchten, im Navigationsfenster.
3. Wählen Sie aus dem Kontextmenü den Befehl "Löschen".
4. Wenn Sie eine Prozedur löschen möchten: Öffnen Sie das entsprechende Modul und löschen Sie den entsprechenden Code im Editierfenster.

Siehe auch

Aktionen (Seite 19)

Module und Prozeduren (Seite 16)

Der Global Script Editor (Seite 29)

1.9 Prozeduren erstellen und bearbeiten

1.9.1 Prozeduren erstellen und bearbeiten

Einleitung

Sie können in WinCC mit VBS Projekt- und Standard-Prozeduren programmieren:

- Projekt-Prozeduren sind nur im aktuellen Projekt aufrufbar. Da Prozeduren im Projektverzeichnis abgelegt sind, werden Sie beim Kopieren eines Projektes automatisch mit kopiert.
- Standard-Prozeduren sind rechnerweit in allen Projekten aufrufbar. Wenn Sie ein Projekt auf einen anderen Rechner kopieren, müssen Sie Standard-Prozeduren manuell in das entsprechende Verzeichnis auf dem Zielrechner kopieren.

Die kopierten Prozeduren stehen in Runtime direkt zur Verfügung. Im Editor werden Sie nach einer Aktualisierung der Ansicht sichtbar.

Neben den von Ihnen programmierten Prozeduren können Sie allgemeine VBS-Funktionen nutzen (z.B. Abs, Array,... Year). Diese allgemeinen VBS-Funktionen können Sie in Ihrem Code über den Kontextmenübefehl "Funktionsliste" aufrufen.

Weiterhin stellt Ihnen WinCC die gängigsten Anweisungen als Codevorlagen zur Verfügung (z.B. If...Then, When...While). Sie können die Codevorlagen aus dem Navigationsfenster auf der Registerkarte Codevorlagen per drag&drop direkt in Ihren Code ziehen.

Wenn Sie einen Codevorlage in Ihren Code einfügen, beachten Sie, dass z.B. Bedingungen in den Vorlagen mit "_XYZ_" gekennzeichnet ist. Diese Platzhalter müssen Sie mit den entsprechenden Angaben ergänzen.

Verwendung von Prozeduren

Prozeduren verwenden Sie um Code, den Sie an mehreren Stellen Ihrer Projektierung verwenden möchten, zentral zu erstellen und zu pflegen. Sie schreiben und speichern den Code in einer Prozedur und rufen diese Prozedur mit den aktuellen Parametern in Aktionen oder anderen Prozeduren auf, anstatt wiederholt denselben Code einzugeben.

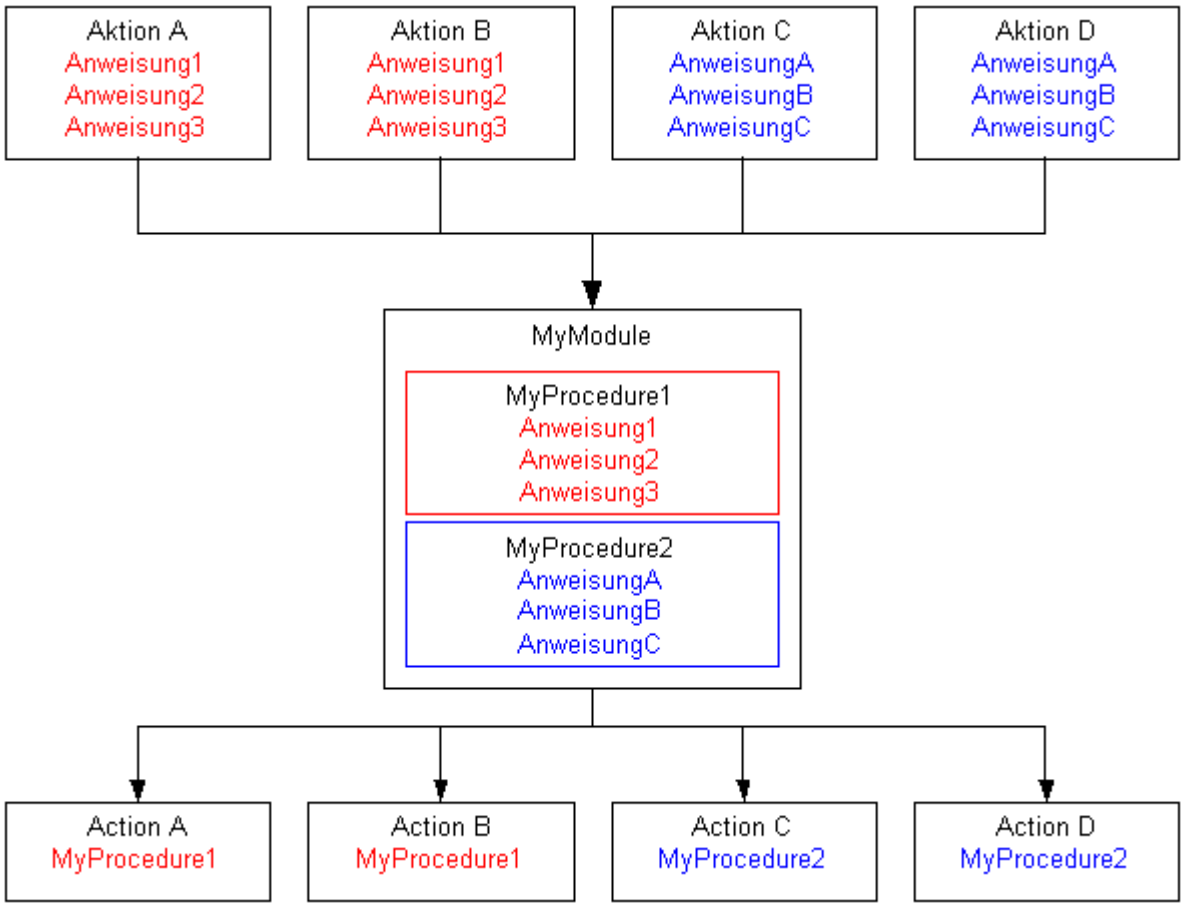
Prozeduren erstellen Sie für sich wiederholende Funktionalitäten, z.B.

- Berechnungen mit unterschiedlichen Ausgangswerten (Prozedur mit Rückgabewert)
- Prüfen von Variablenwerten (Prozedur mit Rückgabewert)
- Ausführende Tätigkeiten (Prozedur ohne Rückgabewert)

Damit sind folgende Vorteile verbunden:

- Der Code wird nur einmal programmiert.
- Änderungen werden nur an einer Stelle gemacht, nämlich in der Prozedur und nicht in jeder Aktion.
- Der Aktionscode wird kürzer und bleibt übersichtlicher.

Zusammengehörige Prozeduren speichern Sie in WinCC immer in Modulen.



Prozeduren werden beim Ausführen der aufrufenden Aktion in Runtime geladen.

Wird eine Prozedur (Modul) geändert, die in einem Bild verwendet wird, so wird die Änderung beim nächsten Laden des Bildes übernommen. Ein aktuell dargestelltes Bild arbeitet also erst mit der geänderten Prozedur, nachdem das Bild erneut geladen wurde.

Nach der Änderung eines Projektmoduls müssen Sie zusätzlich zum Speichern im VBS-Editor das zugehörige Prozessbild im Graphics Designer öffnen und speichern. Erst dann wird die

Änderung in Runtime übernommen. Mit dem Speichern des Bilds werden die Informationen über die benötigten Projektmodule in die Bilddatei übernommen.

Hinweis

Prozeduren können in Aktionen im Global Script und im Graphics Designer verwendet werden.

Wenn Sie eine im Global Script definierte globale Variable in einer Aktion im Graphics Designer verwenden wollen, beachten Sie folgendes:

Damit auf die Variable zugegriffen werden kann, müssen Sie immer die Prozedur aufrufen, in dem die Variable definiert ist.

Wenn Sie eine globale Variable in bildunabhängigen Aktionen im Global Script verwenden wollen, beachten Sie folgendes:

Damit auf die Variable zugegriffen werden kann, muss in mindestens einer globalen Aktion mindestens eine Prozedur des Moduls aufgerufen werden, in dem die Variable definiert wurde.

Abgrenzung Prozedur - Aktion

Globale Prozeduren, die im gesamten Projekt gültig sind, können Sie nur in Global Script erstellen. Im Graphics Designer können Sie nur bildspezifische Prozeduren erstellen und globale Prozeduren in Aktionen aufrufen. Bildspezifische Prozeduren im Graphics Designer definieren Sie im Deklarationsbereich einer Aktion.

Eine Prozedur wird ohne eine Aktion nicht ausgeführt.

Dateinamen und Prozedurnamen

Den Prozedurnamen geben Sie in der ersten Zeile des Prozedurcodes an. Unter diesem Namen wird die Prozedur im Navigationsfenster angezeigt und in Aktionen aufgerufen. Prozeduren haben keinen eigenen Dateinamen, sondern werden in einem Modul gespeichert.

Den Modulnamen vergeben Sie im Navigationsfenster des Editors. Mit dem Befehl "Speichern unter" können Sie ein Modul unter einem anderen Namen im Projektverzeichnis ablegen.

Da Prozeduren in Global Script projektweit gültig sind, müssen Prozedurnamen immer eindeutig sein. Modulnamen können innerhalb eines Projektes doppelt verwendet werden, wenn Sie z.B. in verschiedenen Unterverzeichnissen oder getrennt im Standard- und Projekt-Verzeichnis abgelegt sind.

Darstellung von Prozeduren und Modulen


Wenn Sie ein Modul speichern, das mindestens eine syntaktisch fehlerhafte Prozedur enthält, wird es im Navigationsfenster mit nebenstehendem Symbol angezeigt.

Hinweis

Enthält ein Modul eine syntaktisch fehlerhafte Prozedur, kann das Modul nicht mehr geladen werden. Es lässt sich keine Prozedur aus dem Modul mehr aufrufen.


Wenn Sie ein Modul speichern, das nur syntaktisch korrekte Prozeduren enthält, wird es im Navigationsfenster mit nebenstehendem Symbol angezeigt.

Prozeduren und Module

Prozeduren werden durch ihre Zuordnung zu Standard- oder Projektmodulen klassifiziert als Standard- oder Projektprozedur. Standard- und Projektmodule finden Sie im Navigationsfenster von Global Script auf den entsprechenden Registerkarten.

Nutzen Sie Module, um Prozeduren zu sinnvollen Funktions-Gruppen zusammenzufassen. Beachten Sie bei der Strukturierung Ihrer Module und Prozeduren folgendes:

- Je mehr Module beim Aufruf eines Bildes geladen werden müssen, desto schlechter ist die Performance in Runtime.
- Je größer ein Modul ist, also je mehr Prozeduren es enthält, desto länger ist die Ladezeit für das Modul.

Gliedern Sie Ihre Module also sinnvoll, z. B. ein Modul mit Prozeduren für ein bestimmtes Anlagenteil/Bild.

Siehe auch

So verwenden Sie Prozeduren und Aktionen mehrfach (Seite 22)

So benennen Sie eine Prozedur oder ein Modul um (Seite 54)

So speichern Sie eine Prozedur (Seite 52)

So schützen Sie ein Modul mit einem Passwort (Seite 51)

So fügen Sie modulbegleitende Informationen hinzu (Seite 49)

So verwenden Sie Standard- und Projektprozeduren (Seite 48)

So schreiben Sie den Prozedurcode (Seite 45)

So legen Sie eine neue Prozedur an (Seite 43)

Beispiele zu VBScript (Seite 796)

Module und Prozeduren (Seite 16)

1.9.2 So legen Sie eine neue Prozedur an

Einleitung

Sie können mit Global Script in WinCC Standard- und Projekt-Prozeduren programmieren.

Sie entscheiden mit der Zuordnung zu einem Projekt- oder Standardmodul über die Art der Prozedur. Die Vorgehensweise zum Erstellen von Standard- oder Projektprozeduren ist identisch.

Wenn Sie eine neue Prozedur anlegen, vergibt WinCC automatisch einen Standardnamen "procedure#", wobei # eine fortlaufende Nummer ist. Wenn Sie die Prozedur im Editierfenster bearbeiten, geben Sie der Prozedur einen sprechenden Namen, über den Sie die Prozedur später in Aktionen aufrufen. Der neue Name wird beim Speichern der Prozedur im Navigationsfenster übernommen.

Hinweis

Prozedurnamen müssen in einem Projekt eindeutig sein. Wenn eine Prozedur mit gleichem Namen bereits vorhanden ist, wird das Modul als syntaktisch fehlerhaft gekennzeichnet. Modulnamen können doppelt verwendet werden, wenn die Module in unterschiedlichen Verzeichnissen abgelegt sind.

Sie können globale Prozeduren (projektweit gültig) nur in Global Script programmieren. Im Graphics Designer können Sie Prozeduren über Aktionen aufrufen und bildbezogene Prozeduren im Deklarationsbereich einer Aktion erstellen. In einer globalen Aktion im Global Script können Sie auch direkt im Code Prozeduren erstellen, die dann nur für diese Aktion gültig sind.

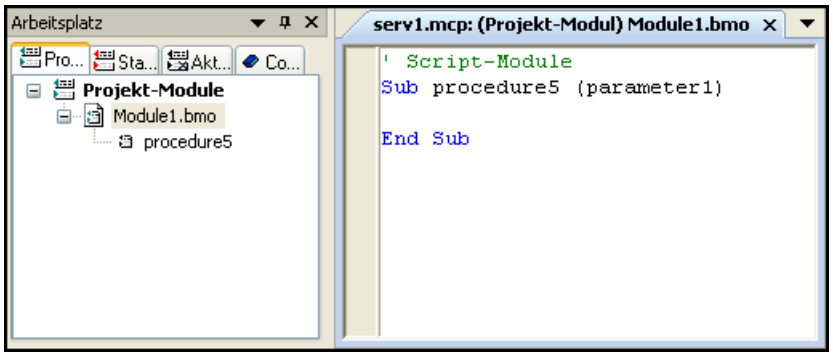
Um Prozeduren aufzurufen, programmieren Sie immer eine Aktion.

Vorgehensweise

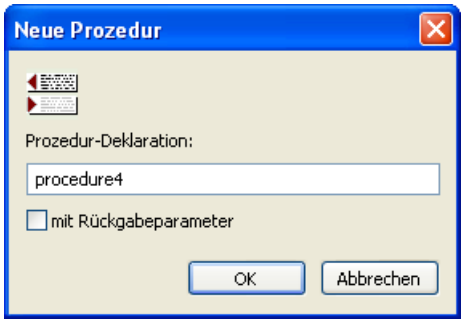
Die folgende Vorgehensweise beschreibt das Erstellen einer neuen Prozedur in Global Script:

1. Öffnen Sie Global Script.
2. Wählen Sie im Navigationsfenster die Registerkarte Standardmodule oder Projektmodule aus, je nachdem ob Sie eine Standardprozedur oder eine Projektprozedur anlegen möchten.
3. Öffnen Sie ein vorhandenes Modul oder legen Sie über den Menübefehl "Datei" > "Neu" > "Projektmodul" bzw. "Datei" > "Neu" > "Standardmodul" ein neues Modul an.

4. Wenn Sie ein neues Modul erstellt haben, wird die Struktur einer Prozedur ohne Rückgabewert im Editierfenster bereits eingetragen:



5. Tragen Sie den Prozedurnamen direkt in den Code ein: Sub "Prozedurname".
6. Wenn Sie eine Prozedur in ein bestehendes Modul einfügen möchten: Markieren Sie das Modul im Navigationsfenster und wählen Sie den Kontextmenübefehl "Neue Prozedur einfügen".
   Der Dialog "Neue Prozedur" erscheint:



7. Geben Sie einen Prozedurnamen an und wählen Sie, ob die Prozedur einen Rückgabeparameter haben soll. Die Definition einer Variable für den Rückgabewert wird dann bereits in den Code mit eingetragen (Dim RetVal).
8. Bestätigen Sie Ihre Eingaben mit OK.

Hinweis

Sie können eine neue Prozedur natürlich auch direkt im Modul eintragen. Beginnen Sie eine Prozedur ohne Rückgabewert immer mit der Anweisung "Sub <Prozedurname>" und schließen Sie sie mit "End Sub" ab. Eine Funktion mit Rückgabewert beginnen Sie mit der Anweisung "Function <Prozedurname>" und schließen sie mit "End Function" ab. Beim Speichern des Moduls wird die neue Prozedur im Navigationsfenster angezeigt.

Siehe auch

Prozeduren erstellen und bearbeiten (Seite 39)

So verwenden Sie Prozeduren und Aktionen mehrfach (Seite 22)

So benennen Sie eine Prozedur oder ein Modul um (Seite 54)

So speichern Sie eine Prozedur (Seite 52)

So schützen Sie ein Modul mit einem Passwort (Seite 51)

So fügen Sie modulbegleitende Informationen hinzu (Seite 49)

So verwenden Sie Standard- und Projektprozeduren (Seite 48)

So schreiben Sie den Prozedurcode (Seite 45)

Module und Prozeduren (Seite 16)

1.9.3 So schreiben Sie den Prozedurcode

Einleitung

Sie schreiben den Prozedurcode im Editierfenster von Global Script. Jede Prozedur kann in ihrem Code andere Prozeduren über deren Prozedurnamen aufrufen.

Sie können Prozeduren mit oder ohne Rückgabewert erstellen. Nutzen Sie den Rückgabewert, um z.B. Auskunft über die erfolgreiche Ausführung der Prozedur zu erhalten.

Wenn Sie eine Prozedur in einem Bild ändern, wird die Änderung erst beim nächsten Laden des Bildes übernommen.

Funktionen in Global Script

Global Script unterstützt Sie beim Erstellen von Prozedurcode mit folgenden Funktionen:

Intellisense und Syntaxhervorhebung

Während der Eingabe erscheinen kontextabhängige Listen, in denen die an der aktuellen Codestelle möglichen Eigenschaften, Methoden und Objekte angeboten werden. Wenn Sie ein Element der Liste einfügen, wird automatisch die erforderliche Syntax mit angegeben.

Hinweis

Vollständige Intellisense für alle Objekte ist im Graphics Designer nur nutzbar, wenn über den Objektnamen auf die Auflistung zugegriffen und das Ergebnis einer Variablen zugewiesen wird. Andernfalls werden Ihnen nur die Standardeigenschaften angeboten.

Beispiel einer vollständigen Intellisense:

```
Dim Variable
Set Variable = ScreenItems ("Kreis1")

Variable.<Intellisense-Auswahl>
```

Wenn bei der Adressierung Bildfenstergrenzen überschritten werden, werden ebenfalls nur die Standardeigenschaften angeboten, da das Bild des Bildfensters nicht geladen ist.

Allgemeine VBS-Funktionen

Über den Befehl "Funktionsliste" des Kontextmenüs im Editierfenster können Sie sich eine Liste mit allgemeinen VBS-Funktionen anzeigen lassen.

Auflistungen von Objekten, Eigenschaften und Methoden

Über das Kontextmenü im Editierfenster können Sie mit dem Befehl "Objektliste" im Graphics Designer eine Liste der möglichen Objekte anzeigen lassen. Im Global Script erhalten Sie in dieser Liste nur das Objekt "HMIRuntime", da auf die Objekte des Graphics Designer nicht direkt zugegriffen werden kann.

Über den Befehl "Eigenschaften/Methoden" des Kontextmenüs erhalten Sie eine Auflistung der möglichen Eigenschaften und Methoden.

Die gleichen Listen erhalten Sie je nach Kontext im Skript mit der Tastenkombination <Strg +Leertaste>.

Codevorlagen

Im Navigationsfenster des Editors auf der Registerkarte "Codevorlagen" finden Sie eine Auswahl häufig verwendeter Anweisungen, z.B. für Schleifen und bedingte Anweisungen. Sie können diese Vorlagen per "drag&drop" in den Prozedurcode einfügen.

Wenn Sie eine Codevorlage in Ihren Code einfügen, müssen Sie die Platzhalter "_XYZ_" in den Vorlagen durch die entsprechenden Angaben ersetzen.

Auswahldialoge

Wenn Sie WinCC-Variablen oder WinCC-Objekte im Code verwenden, stehen Ihnen folgende Auswahldialoge zur Verfügung:

- Öffnet einen Auswahldialog für Variablen und gibt den selektierten Variablennamen als Rückgabewert.
- Öffnet einen Auswahldialog für Variablen und gibt den Variablennamen mit zugehöriger Referenz zurück.

- Öffnet einen Bild/Objektbrowser in dem man ein Bild/Objekt auswählen kann, dessen Name als Rückgabewert geliefert wird.
- Öffnet einen Auswahldialog für Bilder und gibt den Bildnamen ggf. mit Serverpräfix zurück.

Syntaxprüfung

Global Script unterstützt Sie mit einer Syntaxprüfung, die Sie nach dem Erstellen des Codes durchführen. Syntaktische Fehler im Code werden im Ausgabefenster des Editors angezeigt. Sie erreichen die fehlerhafte Stelle im Code durch einen Doppelklick auf den Fehler im Ausgabefenster.

Hinweis

Die Syntaxprüfung kann nur syntaktische Fehler im Code erkennen. Programmierfehler, wie z.B. fehlende Referenzen werden erst in Runtime sichtbar. Prüfen Sie daher Ihre Skripte immer auch in der Runtime-Umgebung.

Ändern einer Prozedur

Wenn Sie eine Prozedur während Runtime ändern, wird die Änderung zu folgenden Zeitpunkten aktiv:

- Prozeduren, die von Aktionen oder anderen Prozeduren in Bildern aufgerufen werden, werden nach einem Bildwechsel aktiv.
- Prozeduren in Global Script sind direkt beim nächsten Aufruf aktiv.

Vorgehensweise

1. Öffnen Sie Global Script.
2. Öffnen Sie das Modul, in dem Sie eine Prozedur bearbeiten möchten.
3. Durch Doppelklick auf die Prozedur im Navigationsfenster springt der Cursor an den Anfang der gewünschten Prozedur.
4. Bearbeiten Sie die Prozedur. Wenn Sie eine Prozedur mit Rückgabeparameter erstellen, z.B. um wiederkehrende Berechnungen oder Überprüfungen zu programmieren, geben Sie am Ende der Prozedur den Rückgabewert mit "procedurename = RetVal" an.

Siehe auch

So schreiben Sie den Prozedurcode (Seite 45)
So verwenden Sie Prozeduren und Aktionen mehrfach (Seite 22)
So benennen Sie eine Prozedur oder ein Modul um (Seite 54)
So speichern Sie eine Prozedur (Seite 52)
So schützen Sie ein Modul mit einem Passwort (Seite 51)
So fügen Sie modulbegleitende Informationen hinzu (Seite 49)
So legen Sie eine neue Prozedur an (Seite 43)
Module und Prozeduren (Seite 16)
Prozeduren erstellen und bearbeiten (Seite 39)

1.9.4 So verwenden Sie Standard- und Projektprozeduren

Einleitung

Eine Prozedur fügen Sie in den aktuellen Code einfach über drag&drop aus dem Navigationsfenster oder über das Kontextmenü in den aktuellen Code ein.

Projektprozeduren können Sie nur innerhalb des aktuellen Projektes verwenden, Standardprozeduren stehen Ihnen projektübergreifend in allen auf dem PC befindlichen Projekten zur Verfügung.

Sie können Prozeduren, die Sie einmal erstellt haben auch in anderen Projekten oder auf anderen Rechnern verwenden. Kopieren Sie dazu die Module, welche die Prozeduren enthalten, in die entsprechenden Projekt- bzw. Standardverzeichnisse.

Verwendung von Prozeduren im Graphics Designer und in Global Script

Sie können Prozeduren, die Sie in Global Script definiert haben, in Aktionen in Global Script und im Graphics Designer aufrufen. Beim Ausführen der Aktion in Runtime wird immer das ganze Modul geladen, in dem die Prozedur enthalten ist.

Wenn Sie eine globale Variable verwenden möchten, die in einer Prozedur in Global Script definiert ist, beachten Sie folgendes:

Im Graphics Designer müssen Sie immer die Prozedur aufrufen, in dem die Variable definiert ist, damit die Variable verwendet werden kann. Ohne den Prozeduraufruf wird das entsprechende Modul nicht geladen, und auf die Variable kann nicht zugegriffen werden.

In bildunabhängigen Aktionen im Global Script müssen Sie in mindestens einer globalen Aktion immer mindestens eine Prozedur aus dem Modul aufrufen, in dem die Variable definiert ist.

Hinweis

Im allgemeinen Deklarartionsteil von Bildern wird nicht geprüft, ob ein Prozedur- oder Funktionsname bereits vergeben ist. Ein Name könnte so mehrfach vorkommen, und es ist nicht definiert, welche Funktion ausgeführt wird. Dies ist das Standardverhalten der MS Scripting Engine.

Vorgehensweise

1. Öffnen Sie Prozedur oder die Aktion, in die Sie die Prozedur einfügen möchten.
2. Ziehen Sie die einzufügende Prozedur aus dem Navigationsfenster per drag&drop an die richtige Stelle im Code.
   oder
3. Stellen Sie den Cursor an die Stelle im Code, an der Sie die Prozedur einfügen möchten.
4. Markieren Sie die Prozedur im Navigationsfenster mit der Maus.
5. Wählen Sie den Kontextmenübefehl "Prozeduraufruf übertragen".

Siehe auch

Prozeduren erstellen und bearbeiten (Seite 39)

So verwenden Sie Prozeduren und Aktionen mehrfach (Seite 22)

So benennen Sie eine Prozedur oder ein Modul um (Seite 54)

So speichern Sie eine Prozedur (Seite 52)

So schützen Sie ein Modul mit einem Passwort (Seite 51)

So fügen Sie modulbegleitende Informationen hinzu (Seite 49)

So schreiben Sie den Prozedurcode (Seite 45)

So legen Sie eine neue Prozedur an (Seite 43)

Module und Prozeduren (Seite 16)

1.9.5 So fügen Sie modulbegleitende Informationen hinzu

Einleitung

Sie können jedem Modul begleitende Informationen hinzufügen, um bei einer späteren Bearbeitung schnell die Funktionalität des Moduls bzw. der in ihm enthaltenen Prozeduren erkennen zu können. Projektieren Sie mit mehreren Bearbeitern an einem Projekt, sollten Sie für Ihre Kollegen modulbegleitende Informationen hinterlegen.

Wenn Sie eine neues Modul anlegen, wird automatisch und nicht änderbar das Erstellungsdatum in den modulbegleitende Informationen eingetragen. Zusätzlich wird dem Modul die Versionsnummer 1.0 vergeben. Die Versionsnummer können Sie beim bearbeiten eines Modul individuell vergeben. Wenn Sie ein Modul ändern und speichern, wird automatisch und nicht änderbar das aktuelle Änderungsdatum eingetragen.

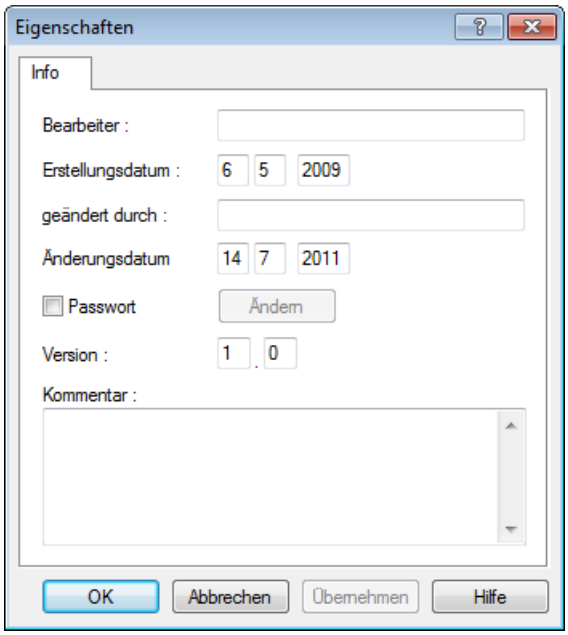
Sie können folgende Informationen zusätzlich hinterlegen:

- "Bearbeiter"
- "Geändert durch"
- "Kommentar" z.B. Funktionalität des Moduls/der in ihm enthaltenen Prozeduren

Weiterhin können Sie hier ein Passwort für das Modul eingeben. Weitere Informationen zur Vergabe von Passworten finden Sie unter "Modul mit einem Passwort schützen".

Vorgehensweise

1. Öffnen Sie Global Script.
2. Markieren Sie das Modul, dem Sie Informationen beistellen möchten, im Navigationsfenster.
3. Betätigen Sie die Schaltfläche "Info/Trigger" in der Symbolleiste oder wählen Sie den Kontextmenübefehl "Info". Der Dialog "Eigenschaften" erscheint.



Geben Sie die gewünschten Informationen ein.

Hinweis

Sie können den Dialog "Info/Trigger" auch aufrufen, wenn Sie eine geöffnete Prozedur im Navigationsfenster markieren. Die Informationen, die Sie in diesem Dialog hinterlegen sind jedoch immer für das gesamte Modul und alle in ihm enthaltenen Prozeduren gültig.

Siehe auch

So verwenden Sie Prozeduren und Aktionen mehrfach (Seite 22)

So benennen Sie eine Prozedur oder ein Modul um (Seite 54)

So speichern Sie eine Prozedur (Seite 52)

So fügen Sie modulbegleitende Informationen hinzu (Seite 49)

So verwenden Sie Standard- und Projektprozeduren (Seite 48)

So schreiben Sie den Prozedurcode (Seite 45)

So legen Sie eine neue Prozedur an (Seite 43)

Module und Prozeduren (Seite 16)

Prozeduren erstellen und bearbeiten (Seite 39)

1.9.6 So schützen Sie ein Modul mit einem Passwort

Einleitung

Sie können ein Modul mit einem Passwort gegen unbefugten Zugriff schützen. Das Passwort ist Teil der modulbegleitenden Informationen.

Hinweis

Wenn Sie ein Modul mit einem Passwort schützen, schützen Sie damit auch immer alle in ihm enthaltenen Prozeduren mit dem Passwort.

Vorgehensweise

1. Öffnen Sie Global Script.
2. Markieren Sie das Modul, das Sie mit einem Passwort schützen möchten, im Navigationsfenster.
3.  Betätigen Sie die Schaltfläche "Info/Trigger" in der Symbolleiste oder wählen Sie den Kontextmenübefehl "Info". Der Dialog "Eigenschaften" erscheint.
4. Aktivieren Sie das Kontrollkästchen "Passwort".
   Der Dialog "Passwort-Eingabe" erscheint.
5. Geben Sie ein Passwort ein und bestätigen Sie es.
6. Bestätigen Sie Ihre Eingaben mit OK.

Ergebnis

Wenn Sie das Modul oder eine in ihm enthaltene Prozedur öffnen, wird zunächst die Eingabe eines Passwortes verlangt.

Passwortschutz deaktivieren

Wenn Sie den Passwortschutz wieder aufheben möchten, deaktivieren Sie das Kontrollkästchen "Passwort" wieder.

Passwort ändern

Wenn Sie das Passwort ändern möchten, rufen Sie den Eigenschaften-Dialog auf und betätigen Sie die Schaltfläche "Ändern". Geben Sie dann Ihr neues Passwort ein.

Hinweis

Wenn Sie das Passwort eines Moduls vergessen, können Sie das Modul nicht mehr bearbeiten.

Hinweis

Sie können den Dialog "Info/Trigger" auch aufrufen, wenn Sie eine Prozedur im Navigationsfenster markiert haben. Die Informationen, die Sie in diesem Dialog hinterlegen sind jedoch immer für das gesamte Modul und alle in ihm enthaltenen Prozeduren gültig.

Siehe auch

So verwenden Sie Prozeduren und Aktionen mehrfach (Seite 22)

So benennen Sie eine Prozedur oder ein Modul um (Seite 54)

So speichern Sie eine Prozedur (Seite 52)

So fügen Sie modulbegleitende Informationen hinzu (Seite 49)

So verwenden Sie Standard- und Projektprozeduren (Seite 48)

So schreiben Sie den Prozedurcode (Seite 45)

So legen Sie eine neue Prozedur an (Seite 43)

Module und Prozeduren (Seite 16)

Prozeduren erstellen und bearbeiten (Seite 39)

1.9.7 So speichern Sie eine Prozedur

Einleitung

Sie speichern nie einzelne Prozeduren, sondern immer das Modul, in dem Sie die Prozedur programmiert haben.

Bevor Sie ein Modul speichern, sollten Sie den Code auf syntaktische Korrektheit überprüfen. Beim Speichern eines Moduls werden die enthaltenen Prozeduren automatisch überprüft, und Sie erhalten bei Syntaxfehlern eine Abfrage, ob Sie das Modul fehlerhaft speichern möchten

oder nicht. So können Sie z.B. Module mit Prozeduren ablegen, die noch nicht fertig programmiert sind. Syntaktisch fehlerhafte Prozeduren laufen in Runtime nicht ab.

Hinweis

Enthält ein Modul eine syntaktisch fehlerhafte Prozedur, kann das Modul nicht mehr geladen werden. Es lässt sich keine Prozedur aus dem Modul mehr aufrufen.

Hinweis

Die Syntaxprüfung kann nur syntaktische Fehler im Code erkennen. Programmierfehler, wie z.B. fehlende Referenzen werden erst in Runtime sichtbar. Prüfen Sie daher Ihre Skripte immer auch in der Runtime-Umgebung und verwenden Sie ggf. einen Debugger, um Fehler zu erkennen und zu beseitigen.

In Runtime werden nur syntaktisch korrekte Module aufgerufen.

Eine Auflistung der möglichen syntaktischen Fehler finden Sie im Anhang unter "Grundlagen von VBScript".

Wenn Sie eine Prozedur vor dem Speichern auf syntaktische Fehler überprüfen, werden Ihnen eventuell vorhandene Fehler im unteren Teil des Editierfensters angezeigt. Durch Doppelklick auf eine Fehlerzeile gelangen Sie direkt an die fehlerhafte Stelle im Code.

Verwenden Sie den Befehl "Speichern unter", wenn Sie das Modul unter einem anderen Namen speichern möchten. Beachten Sie, dass das neue Modul erst nach dem Aktualisieren der Ansicht im Navigationsfenster angezeigt wird.

Voraussetzung

Die zu speichernde Prozedur/das Modul muss im Editierfenster geöffnet sein.

Vorgehensweise

1. Betätigen Sie die Schaltfläche "Syntaxprüfung" in der Symbolleiste.
2. Wenn Syntaxfehler im Ausgabefenster angezeigt werden, doppelklicken Sie auf die Fehlerzeile und korrigieren Sie den Fehler im Code. Wiederholen Sie die Schritte 1 und 2, bis der Code fehlerfrei ist.
3. Speichern Sie das Modul mit der Schaltfläche "Speichern" in der Symbolleiste.

Hinweis

Bild mit geänderter Prozedur müssen Sie im Graphics Designer erneut öffnen und speichern

Zusätzlich zum Speichern im VBS-Editor müssen Sie bei einer Änderung des Projektmoduls das zugehörige Bild im Graphics Designer erneut öffnen und speichern. Dann ist die Änderung in Runtime wirksam. Erst mit dem Speichern des Bilds werden die Informationen über die benötigten Projektmodule in die Bilddatei übernommen.

Siehe auch

Diagnose (Seite 85)

So verwenden Sie Prozeduren und Aktionen mehrfach (Seite 22)

So speichern Sie eine Prozedur (Seite 52)

So schützen Sie ein Modul mit einem Passwort (Seite 51)

So fügen Sie modulbegleitende Informationen hinzu (Seite 49)

So verwenden Sie Standard- und Projektprozeduren (Seite 48)

So schreiben Sie den Prozedurcode (Seite 45)

So legen Sie eine neue Prozedur an (Seite 43)

Module und Prozeduren (Seite 16)

Prozeduren erstellen und bearbeiten (Seite 39)

1.9.8 So benennen Sie eine Prozedur oder ein Modul um

Einleitung

In folgenden Fällen benennen Sie Prozeduren oder Module um:

- Wenn Sie den Standardnamen (procedure# bzw. Modul#), der beim Anlegen eines neuen Moduls/einer neuen Prozedur automatisch vergeben wird, in einen sprechenden Namen ändern.
- Wenn Sie ein Modul oder eine Prozedur kopieren, um z.B. aus einem bereits bestehenden Modul ein neues Modul mit ähnlichem Inhalt zu erstellen.
  Beachten Sie dabei, dass Prozedurnamen innerhalb eines Projektes immer eindeutig sein müssen. Doppelt vorhandene Prozedurnamen werden bei der Syntaxprüfung als Fehler ausgegeben.
  Im Gegensatz zu Prozedurnamen können Sie für Module gleiche Namen verwenden, wenn die betroffenen Module in unterschiedlichen Verzeichnissen abgelegt sind.

Hinweis

Der Modulname ist immer identisch mit dem Dateinamen des Moduls im WinCC-Dateisystem. Ändern Sie einen Modulnamen z.B. im Windows-Explorer, wird der neue Modulname im Navigationsfenster von Global Script übernommen.

Vorgehensweise

Prozedur umbenennen

1. Öffnen Sie die Prozedur, die Sie umbenennen möchten.
2. Geben Sie den neuen Namen in der Kopfzeile der Prozedur ein.
3. Speichern Sie die Prozedur, damit der Name in das Navigationsfenster übernommen wird. Prozedurnamen sind immer eindeutig und dürfen nicht mehrfach verwendet werden.

Modul umbenennen

1. Schließen Sie das Modul, das Sie umbenennen möchten.
2. Markieren Sie das Modul im Navigationsfenster und wählen Sie den Kontextmenübefehl "Umbenennen".
3. Geben Sie den neuen Namen im Navigationsfenster ein. Modulnamen sind auf Verzeichnisebene immer eindeutig und dürfen nicht mehrfach verwendet werden.

Siehe auch

So verwenden Sie Prozeduren und Aktionen mehrfach (Seite 22)

So speichern Sie eine Prozedur (Seite 52)

So schützen Sie ein Modul mit einem Passwort (Seite 51)

So fügen Sie modulbegleitende Informationen hinzu (Seite 49)

So verwenden Sie Standard- und Projektprozeduren (Seite 48)

So schreiben Sie den Prozedurcode (Seite 45)

So legen Sie eine neue Prozedur an (Seite 43)

Module und Prozeduren (Seite 16)

Prozeduren erstellen und bearbeiten (Seite 39)

1.10 Aktionen erstellen und bearbeiten

1.10.1 Aktionen erstellen und bearbeiten

Einleitung

In VBS in WinCC gibt es im Gegensatz zu C keine Unterscheidung zwischen lokalen (projektweit gültigen) und globalen (rechnerweit gültigen) Aktionen. Eine projektierte Aktion gilt immer global.

Die kopierten Aktionen stehen in Runtime nach einem Neustart oder dem Öffnen und Speichern der Aktion zur Verfügung. Im Editor werden Sie nach einer Aktualisierung der Ansicht sichtbar.

Mit VBS-Aktionen können Sie in Runtime Grafikobjekte und Objekteigenschaften dynamisieren oder bildunabhängige Aktionen durchführen.

Hinweis

Beachten Sie, dass die Objektnamenlänge von Objekten, die Sie im Graphics Designer dynamisieren, auf ca. 200 Zeichen beschränkt ist, in den Skript-Dateien jedes Sonderzeichen, das Sie in einem Objektnamen verwenden, in fünf Zeichen umgesetzt wird. Hinter einem führenden X wird das Sonderzeichen in vierstelligem Hexadezimalcode dargestellt. Ist der Name eines dynamisierten Objektes zu lang, erscheint eine entsprechende Fehlermeldung. Weitere Informationen erhalten Sie in dieser Hilfe unter "Aufbau von VBScript-Dateien".

Hinweis

Wenn Sie mit einer VBS-Aktion eine Objekteigenschaft über den Rückgabewert eines Skriptes dynamisieren, wird der Wert der Objekteigenschaft nur dann geschrieben, wenn er sich im Vergleich zum letzten Skript-Durchlauf geändert hat. Dabei wird nicht berücksichtigt, ob der Wert von einer anderen Stelle verändert wurde.

Deshalb dürfen Eigenschaften, welche mit einer VBS-Aktion über den Rückgabewert dynamisiert wurden, nicht von anderer Stelle (z.B. anderen C- oder VBS-Scripten) verändert werden.

Wenn Sie dies nicht beachten, dann können falsche Werte angezeigt werden.

Verwendung von Aktionen

Sie können Aktionen folgendermaßen verwenden:

An Grafikobjekten im Graphics Designer

Als Dynamisierung einer Eigenschaft (Aktion mit Rückgabewert), z.B.:

```
Function BackColor_Trigger(ByVal Item)

'VBS143
```

```
    BackColor_Trigger = RGB(125,0,0)
End Function
```

Getriggert durch ein Ereignis an einem Objekt (Aktion ohne Rückgabewert), z.B.:

```
Sub OnClick(ByVal Item)
'VBS144
    Item.BackColor = RGB(255,0,0)
End Sub
```

Bildunabhängig in Global Script

Als zyklische Aktion, z.B. das Hochzählen einer Variablen:

```
Option Explicit
Function action
'VBS145
    Dim objTag1
    Dim lngValue
    Set objTag1 = HMIRuntime.Tags("Tag1")
    lngValue = objTag1.Read
    objTag1.Write lngValue + 1
    action = CLng(objTag1.value)
End Function
```

Ausführung von Aktionen

Sie können einer Aktion mehrere Trigger zuweisen. Die Aktion wird immer ausgeführt, wenn eines der Triggerereignisse eintritt. Beachten Sie aber folgende Besonderheiten:

- Aktionen in Global Script können nicht gleichzeitig ausgeführt werden. Die zuletzt angestoßene Aktion wird in einer Warteschleife gehalten, bis die laufende Aktion beendet ist.
- Im Graphics Designer können zyklische und variablengetriggerte Aktionen nicht gleichzeitig ausgeführt werden. Behindert die Ausführung einer variablengetriggerten Aktion eine zyklische Aktion, wird die zyklische Aktion erst wieder ausgeführt, wenn die variablengetriggerte Aktion beendet ist. Die zyklische Aktion wird während der Nicht-Ausführung nicht in einer Warteschleife gehalten. Ist die laufende Aktion beendet, wird die zyklische Aktion wieder mit dem normalen Zyklus ausgeführt.
- Im Graphics Designer können keine ereignisgesteuerten Aktionen gleichzeitig ausgeführt werden.

Die genannten Aktions-Typen behindern sich gegenseitig nicht in ihrer Ausführung: Die Ausführung von Aktionen im Global Script hat keinen Einfluss auf Aktionen im Graphics Designer. Ebenso hat im Graphics Designer die Ausführung von zyklischen oder

variablengetriggerten Aktionen keinen Einfluss auf die Ausführung von ereignisgesteuerten Aktionen.

Hinweis

Aktionen in Bildern, die eine Minute nach Abwahl des Bildes noch laufen, werden vom System beendet. Dies wird in einem Logfileeintrag ausgegeben.

Auffinden von Aktionen

Sie können sich alle in einem Bild verwendeten Aktionen über die Eigenschaften des Bildes anzeigen lassen. Markieren Sie dazu das Bild im WinCCExplorer und wählen Sie den Kontextmenübefehl "Eigenschaften". Durch Doppelklick auf einen Eintrag erhalten Sie detaillierte Informationen zur Art der Dynamisierung.

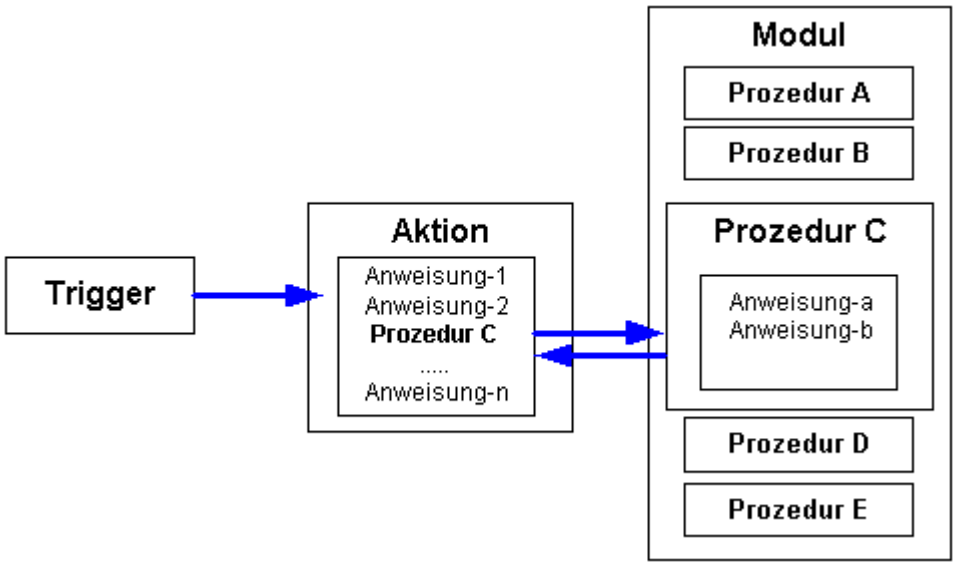
Sie können sich außerdem alle in Aktionen verwendeten Variablen und Bilder über die WinCC CrossReference anzeigen lassen. Mit der CrossReference können Sie auch Variablenanbindungen von Aktionen des Graphics Designer bequem umverdrahten.

Hinweis

Verwenden Sie für die Adressierung von Bildern und Variablen in Ihrem Code die Standardformulierungen HMIRuntime.BaseScreenName = "Screenname" und HMIRuntime.Tags("Tagname"), damit sichergestellt ist, dass Bilder und Variablen von der CrossReference erfasst werden.

Abgrenzung Aktion - Prozedur

In Aktionen können Sie Anweisungen programmieren und Prozeduren aufrufen. In Prozeduren programmieren Sie Code, der an mehreren Stellen Ihrer Projektierung verwendet wird. Aktionen besitzen im Gegensatz zu Prozeduren immer einen Trigger.



Erstellen und Bearbeiten von Aktionen

Aktionen können Sie in Global Script und im Graphics Designer projektieren. In Global Script projektieren Sie globale Aktionen, die unabhängig vom geöffneten Bild ausgeführt werden sollen. Im Graphics Designer projektieren Sie Aktionen an Grafikobjekte, die ausgeführt werden, wenn das Bild in Runtime geöffnet wird oder wenn der projektierte Trigger eintritt.

Die Skript-Editoren in WinCC bieten Ihnen die Möglichkeit, Ihre Skripte auf syntaktische Korrektheit zu überprüfen, ohne sie auszuführen. Fehler im Skript werden Ihnen im Ausgabefenster unterhalb des Editierfensters angezeigt. Mit einem Doppelklick auf die Fehlerzeile gelangen Sie direkt in die entsprechende Stelle im Code.

Hinweis

Die Syntaxprüfung kann nur Objekte überprüfen, die dem System zum Ausführungszeitpunkt bekannt sind. Variablen und Objekte, die Sie in Ihren Skripten ansprechen, müssen in WinCC also angelegt sein.

In Runtime werden nur syntaktisch korrekte Aktionen ausgeführt.

Die Automatisierungsobjekte "PDLRuntime" und "WinCC-Runtime-Projekt" können in VBS-Aktionen nicht verwendet werden.

Darstellung von Aktionen

 Wenn Sie eine syntaktisch fehlerhafte Aktion speichern, wird sie im Navigationsfenster von Global Script mit nebenstehendem Symbol angezeigt.

 Wenn Sie eine syntaktisch korrekte Aktion ohne Trigger speichern, wird sie im Navigationsfenster von Global Script mit nebenstehendem Symbol angezeigt.

 Wenn Sie eine syntaktisch korrekte Aktion mit Trigger speichern, wird sie im Navigationsfenster von Global Script mit nebenstehendem Symbol angezeigt.

Hinweis

Im Graphics Designer können Sie nur syntaktisch korrekte Aktionen übernehmen. Falls Sie eine Aktion mit Fehlern dennoch behalten und verlassen wollen, kommentieren Sie zuvor alles aus.

Systemverhalten, wenn in Runtime Aktionen verändert, gelöscht und gespeichert werden

Wenn eine lokale Aktion während Runtime gespeichert wird, dann werden auf dem Rechner, zu dem die lokale Aktion gehört, alle lokalen und globalen Aktionen des Rechners zurückgesetzt.

Wird eine globale Aktion während Runtime gespeichert, dann werden alle lokalen und globalen Aktionen des gesamten Projekts und damit auf allen Rechnern zurückgesetzt.

Das Zurücksetzen hat zur Folge, dass z. B. die Variablen und Zeiten, die als Trigger in den Aktionen verwendet werden, neu initialisiert werden und dadurch die Aktion ausgelöst wird.

Statische Variablen, die in den zurückgesetzten Aktionen verwendet werden, werden neu initialisiert.

Siehe auch

Aufbau von VBScript-Dateien (Seite 99)
Aktions- und Prozedurnamen im Debugger (Seite 101)
So benennen Sie eine Aktion um (Seite 82)
So speichern Sie eine Aktion (Seite 68)
So schützen Sie eine Aktion mit einem Passwort (Seite 67)
So fügen Sie aktionsbegleitende Informationen hinzu (Seite 65)
So bearbeiten Sie eine Aktion (Seite 61)
So legen Sie eine neue Aktion an (Seite 60)
Trigger (Seite 70)
Aktionen (Seite 19)

1.10.2 So legen Sie eine neue Aktion an

Einleitung

Beim Anlegen einer neuen Aktion schlägt Ihnen der Editor automatisch einen Dateinamen vor (Aktion#.bac), den Sie nachträglich ändern können.

Sie können Aktionen in Global Script und im Graphics Designer projektieren:

- In Global Script projektieren Sie Aktionen, die bildunabhängig in Runtime ausgeführt werden soll. Sie öffnen Global Script über den WinCCExplorer.
- Im Graphics Designer projektieren Sie eine neue Aktion an die Eigenschaft eines Grafikobjektes, in dem Sie auf der Registerkarte Eigenschaften in der Spalte "Dynamik" die rechte Maustaste betätigen und den Eintrag VBS-Aktion wählen. Auf die gleiche Art projektieren Sie eine Aktion an ein Ereignis auf der Registerkarte Ereignis.
  In beiden Fällen öffnet sich der Aktionseditor des Graphics Designer.

Hinweis

Die genaue Vorgehensweise, wie Sie Aktionen mit Grafikobjekten verknüpfen, finden Sie unter dem WinCC-Hilfethema "Dynamisierung".

Vorgehensweise

1. Öffnen Sie Global Script.
2. Aktivieren Sie die Registerkarte Aktionen im Navigationsfenster.
3.  Betätigen Sie nebenstehende Schaltfläche in der Symbolleiste oder wählen Sie den Menübefehl "Datei" > "Neu" > "Aktion".
   Eine neue Aktion wird im Editierfenster geöffnet. Die Aktion erscheint im Navigationsfenster, wenn Sie sie gespeichert haben.

Hinweis

Beim Anlegen einer neuen Aktion wird die Anweisung "Option explicit" automatisch und nicht löschbar in den Deklarationsbereich eingetragen. Die Anweisung ist erforderlich um Fehler durch falsche Schreibweise von Variablen ohne Deklaration zu vermeiden.

Die Anweisung erfordert, dass Sie Variablen immer mit der Anweisung "Dim" in Ihrem Code definieren.

Verwenden Sie die Anweisung "Option explicit" nicht in Ihrem Code, da es sonst zu Lauzeitfehlern kommen kann.

Siehe auch

So benennen Sie eine Aktion um (Seite 82)

So speichern Sie eine Aktion (Seite 68)

So schützen Sie eine Aktion mit einem Passwort (Seite 67)

So fügen Sie aktionsbegleitende Informationen hinzu (Seite 65)

So bearbeiten Sie eine Aktion (Seite 61)

Trigger (Seite 70)

Aktionen erstellen und bearbeiten (Seite 56)

Aktionen (Seite 19)

1.10.3 So bearbeiten Sie eine Aktion

Einleitung

Eine Aktion bearbeiten Sie wie eine Prozedur im Editierfenster des Editors bzw. im Aktionseditor des Graphics Designer.

Damit eine Aktion in Runtime ausgeführt wird, benötigt Sie einen Trigger. Aktionen, die durch ein Ereignis im Graphics Designer ausgelöst werden, müssen Sie keinen Trigger zuweisen.

Wenn Sie eine Aktion während Runtime ändern, wird die Änderung bei erneutem Laden des Bildes (bei Aktionen im Graphics Designer) oder beim nächsten Aufruf der Aktion (bei Aktionen in Global Script) übernommen.

Hinweis

Eine Änderung im Code kann in Runtime nicht übernommen werden, während gerade eine andere Aktion ausgeführt wird.

Sie können einen Prozeduraufruf in die Aktion einfügen, in dem Sie die Prozedur aus dem Navigationsfenster des Editors per "drag&drop" an die entsprechende Stelle des Codes in das Editierfenster ziehen. C-Skripte können Sie in VBS-Aktionen nicht aufrufen.

Deklarationsbereich in Aktionen

Wenn Sie Aktionen im Graphics Designer erstellen, können Sie mit der Schaltfläche den Deklarationsbereich der Aktion einblenden. Beim Anlegen einer neuen Aktion wird die Anweisung "Option explicit" automatisch und nicht löschbar in den Deklarationsbereich eingetragen. Die Anweisung ist erforderlich um Fehler durch falsche Schreibweise von Variablen ohne Deklaration zu vermeiden.

Die Anweisung erfordert, dass Sie Variablen immer mit der Anweisung "Dim" in Ihrem Code definieren.

Verwenden Sie die Anweisung "Option explicit" nicht in Ihrem Code, da es sonst zu Laufzeitfehlern kommen kann.

Im Deklarationsbereich können Sie zusätzlich allgemeine Festlegungen treffen, die Sie global für das aktuelle Bild verwenden möchten, z.B.:

- Variablendefinitionen
- Prozeduren, die Sie nur in diesem Bild verwenden möchten

In den Deklarationsbereichen der Aktionen können in den Bereichen "Ereignis" und "Eigenschaften" eines Objektes voneinander unabhängige globale Variablen definiert werden. Zwischen gleichnamigen globalen Variablen in diesen beiden Bereichen besteht keine Verbindung.

Hinweis

Definieren Sie Prozeduren im Deklarationsbereich immer syntaktisch korrekt mit "Sub" - "End Sub". Erstellen Sie keinen direkt ausführbaren Code im Deklarationsbereich, da dies zu Laufzeitfehlern führt.

Wenn Sie globale Variablen im Deklarationsbereich von Aktionen im Graphics Designer verwenden, beachten Sie, dass ereignisgetriggerte und zyklische/variablengetriggerte Aktionen in Runtime getrennt abgearbeitet werden. In Runtime erfolgt kein Abgleich globaler Variablen zwischen diesen beiden Runtime-Systemen. Wenn eine Synchronisation der Variablen erforderlich ist, projektieren Sie diese über das DataSet-Objekt oder interne WinCC-Variablen.

Beachten Sie bei Definitionen im Deklarationsbereich den Aufbau der Script-Dateien, wie unter "Aufbau von VBScript-Dateien" beschrieben.

Funktionen zum Bearbeiten von Aktionen

Die Skript-Editoren unterstützen Sie beim Erstellen von Aktionscode mit folgenden Funktionen:

Intellisense und Syntaxhervorhebung

Während der Eingabe erscheinen kontextabhängige Listen, in denen Ihnen die an der aktuellen Codestelle möglichen Eigenschaften, Methoden und Objekte angeboten werden.

Wenn Sie ein Element der Liste einfügen, wird Ihnen automatisch die erforderliche Syntax mit angegeben.

Hinweis

Vollständige Intellisense für alle Objekte ist im Graphics Designer nur nutzbar, wenn über den Objektnamen auf die Auflistung zugegriffen und das Ergebnis einer Variablen zugewiesen wird. Sonst werden Ihnen nur die Standardeigenschaften angeboten.

Bsp. für vollständige Intellisense:

Dim Variable
Set Variable = ScreenItems ("Kreis1")

Variable.<Intellisense>

Wenn bei der Adressierung Bildfenstergrenzen überschritten werden, werden ebenfalls nur die Standardeigenschaften angeboten, da das Bild des Bildfensters nicht geladen ist.

Allgemeine VBS-Funktionen

Über den Befehl "Funktionsliste" des Kontextmenüs im Editierfenster können Sie sich eine Liste mit allgemeinen VBS-Funktionen anzeigen lassen.

Auflistungen von Objekten, Eigenschaften und Methoden

Über das Kontextmenü im Editierfenster können Sie mit dem Befehl "Objektliste" im Graphics Designer eine Liste der möglichen Objekte anzeigen lassen. Im Global Script erhalten Sie in dieser Liste nur das Objekt "HMIRuntime", da auf die Objekte des Graphics Designer nicht direkt zugegriffen werden kann.

Über den Befehl "Eigenschaften/Methoden" des Kontextmenüs erhalten Sie eine Auflistung der möglichen Eigenschaften und Methoden.

Die gleichen Listen erhalten Sie je nach Kontext im Skript mit der Tastenkombination <Strg +Leertaste>.

Codevorlagen

Im Navigationsfenster des Editors auf der Registerkarte "Codevorlagen" finden Sie eine Auswahl häufig verwendeter Anweisungen, z.B. für Schleifen und bedingte Anweisungen. Sie können diese Vorlagen per "drag&drop" in den Prozedurcode einfügen.

Wenn Sie eine Codevorlage in Ihren Code einfügen, müssen Sie die Platzhalter "_XYZ_" in den Vorlagen durch die entsprechenden Angaben ersetzen.

Auswahldialoge

Wenn Sie WinCC-Variablen oder WinCC-Objekte im Code verwenden, stehen Ihnen folgende Auswahldialoge zur Verfügung:

- Öffnet einen Auswahldialog für Variablen und gibt den selektierten Variablennamen als Rückgabewert.
- Öffnet einen Auswahldialog für Variablen und gibt den Variablennamen mit zugehöriger Referenz zurück.

- Öffnet einen Bild/Objektbrowser in dem man ein Bild/Objekt auswählen kann, dessen Name als Rückgabewert geliefert wird.
- Öffnet einen Auswahldialog für Bilder und gibt den Bildnamen ggf. mit Serverpräfix zurück.

Syntaxprüfung

Global Script unterstützt Sie durch eine Syntaxprüfung, die Sie nach dem Erstellen des Codes durchführen können. Syntaktische Fehler im Code werden Ihnen im Ausgabefenster des Editors angezeigt. Sie erreichen die fehlerhafte Stelle im Code direkt durch einen Doppelklick auf den Fehler im Ausgabefenster.

Hinweis

Die Syntaxprüfung kann nur syntaktische Fehler im Code erkennen. Programmierfehler, wie z.B. fehlende Referenzen werden erst in Runtime sichtbar. Prüfen Sie daher Ihre Skripte immer auch in der Runtime-Umgebung und verwenden Sie ggf. einen Debugger, um Fehler zu erkennen und zu beseitigen. Wie Sie Skripte mit einem Debugger testen, finden Sie in dieser Dokumentation unter dem Thema "Diagnose" > "Testen mit dem Debugger".

Vorgehensweise

1. Öffnen Sie Global Script.
2. Doppelklicken Sie die Aktion auf der Registerkarte Aktionen im Navigationsfenster.
3. Bearbeiten Sie die Aktion.

Siehe auch

Verwendung von globalen Variablen in VBS (Seite 26)

Testen mit dem Debugger (Seite 92)

Aufbau von VBScript-Dateien (Seite 99)

So benennen Sie eine Aktion um (Seite 82)

So speichern Sie eine Aktion (Seite 68)

So schützen Sie eine Aktion mit einem Passwort (Seite 67)

So bearbeiten Sie eine Aktion (Seite 61)

So legen Sie eine neue Aktion an (Seite 60)

Trigger (Seite 70)

Aktionen erstellen und bearbeiten (Seite 56)

Aktionen (Seite 19)

1.10.4 So fügen Sie aktionsbegleitende Informationen hinzu

Einleitung

Sie können jeder Aktion in Global Script begleitende Informationen hinzufügen, um bei einer späteren Bearbeitung schnell die Funktion der Aktion erkennen zu können. Projektieren Sie mit mehreren Bearbeitern an einem Projekt, sollten Sie für Ihre Kollegen aktionsbegleitende Informationen hinterlegen.

Wenn Sie eine neue Aktion anlegen, wird automatisch und nicht änderbar das Erstellungsdatum in den aktionsbegleitenden Informationen eingetragen. Zusätzlich wird der Aktion die Versionsnummer 1.0 vergeben. Die Versionsnummer können Sie beim bearbeiten einer Aktion individuell vergeben. Wenn Sie eine Aktion ändern und speichern, wird automatisch und nicht änderbar das aktuelle Änderungsdatum eingetragen.

Sie können folgende zusätzliche Informationen hinterlegen:

- "Bearbeiter"
- "Geändert durch"
- "Kommentar" z.B. Funktionalität der Aktion

Zusätzlich können Sie hier ein Passwort für die Aktion eingeben. Weitere Informationen zur Vergabe von Passworten finden Sie unter "Aktion mit einem Passwort schützen".

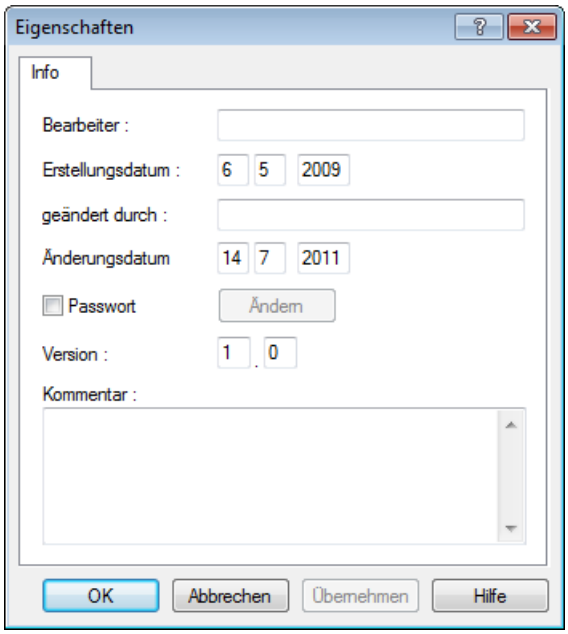
Hinweis

Sie können nur für Aktionen im Global Script zusätzliche Informationen bereitstellen, nicht für Aktionen im Graphics Designer.

Vorgehensweise

1. Öffnen Sie Global Script.
2. Öffnen Sie die Aktion, der Sie Informationen beistellen möchten.

3. Betätigen Sie die Schaltfläche "Info/Trigger" in der Symbolleiste oder wählen Sie den Kontextmenübefehl "Info/Trigger". Der Dialog "Eigenschaften" erscheint.



4. Geben Sie die gewünschten Informationen ein.

Siehe auch

So benennen Sie eine Aktion um (Seite 82)

So speichern Sie eine Aktion (Seite 68)

So schützen Sie eine Aktion mit einem Passwort (Seite 67)

So bearbeiten Sie eine Aktion (Seite 61)

So legen Sie eine neue Aktion an (Seite 60)

Trigger (Seite 70)

Aktionen erstellen und bearbeiten (Seite 56)

Aktionen (Seite 19)

1.10.5 So schützen Sie eine Aktion mit einem Passwort

Einleitung

Sie können eine Aktion in Global Script mit einem Passwort gegen unbefugten Zugriff schützen. Das Passwort ist Teil der aktionsbegleitenden Informationen.

Hinweis

Sie können nur Aktionen im Global Script mit einem Passwort schützen, keine Aktionen im Graphics Designer.

Vorgehensweise

1. Öffnen Sie Global Script.
2. Öffnen Sie die Aktion, die Sie mit einem Passwort schützen möchten.
3. Betätigen Sie die Schaltfläche "Info/Trigger" in der Symbolleiste oder wählen Sie den Kontextmenübefehl "Info/Trigger". Der Dialog "Eigenschaften" erscheint.
4. Aktivieren Sie das Kontrollkästchen "Passwort".
5. Betätigen Sie die Schaltfläche "Ändern". Das Fenster "Passwort-Eingabe" erscheint.
6. Geben Sie ein Passwort ein und bestätigen Sie es.
7. Bestätigen Sie Ihre Eingaben mit OK.

Ergebnis

Wird die Aktion geöffnet, wird zunächst die Eingabe eines Passwortes verlangt.

Passwortschutz deaktivieren

Wenn Sie den Passwortschutz wieder aufheben möchten, deaktivieren Sie das Kontrollkästchen "Passwort" wieder.

Passwort ändern

Wenn Sie das Passwort ändern möchten, rufen Sie den Eigenschaften-Dialog auf und betätigen Sie die Schaltfläche "Ändern". Geben Sie dann Ihr neues Passwort ein.

Hinweis

Wenn Sie das Passwort einer Aktion vergessen, können Sie die Aktion nicht mehr bearbeiten.

Siehe auch

So benennen Sie eine Aktion um (Seite 82)
So speichern Sie eine Aktion (Seite 68)
So fügen Sie aktionsbegleitende Informationen hinzu (Seite 65)
So bearbeiten Sie eine Aktion (Seite 61)
So legen Sie eine neue Aktion an (Seite 60)
Trigger (Seite 70)
Aktionen erstellen und bearbeiten (Seite 56)
Aktionen (Seite 19)

1.10.6 So speichern Sie eine Aktion

Einleitung

Bevor Sie eine Aktion in Runtime ablaufen lassen können, muss Sie gespeichert werden. Sie speichern die Aktion wie jede andere Windows-Datei auch mit dem Befehl "Datei" > "Speichern" oder dem entsprechenden Symbol.

Hinweis

Aktionen im Graphics Designer werden automatisch beim Schließen des Aktionseditors mit dem Bild übernommen. Im Graphics Designer können Sie nur syntaktisch korrekte Funktionen übernehmen. Falls Sie eine Aktion mit Fehlern dennoch behalten und verlassen wollen, kommentieren Sie zuvor alles aus.

Eine Auflistung der möglichen syntaktischen Fehler finden Sie im Anhang unter "Grundlagen von VBScript".

Wenn Sie eine Aktion unter einem anderen Namen speichern möchten, z.B. um die Aktion als Grundlage für eine andere Aktion zu verwenden, verwenden Sie den Befehl "Speichern unter".

Beachten Sie beim Verwenden von "Speichern unter", dass nur der Dateiname geändert wird, nicht der Aktionsname.

Vor dem Speichern

Bevor Sie eine Aktion speichern, sollten Sie den Code auf syntaktische Korrektheit prüfen. Im Ausgabefenster von Global Script werden Ihnen die Syntaxfehler im Code angegeben. Mit einem Doppelklick auf eine Fehlerzeile gelangen Sie an die entsprechende Stelle im Code.

Hinweis

Die Syntaxprüfung kann nur syntaktische Fehler im Code erkennen. Programmierfehler, wie z.B. fehlende Referenzen werden erst in Runtime sichtbar. Prüfen Sie daher Ihre Skripte immer auch in der Runtime-Umgebung und verwenden Sie ggf. einen Debugger, um Fehler zu erkennen und zu beseitigen.

Wenn Sie Aktionen ohne vorherige Syntaxprüfung speichern möchten, weist Sie der Editor darauf hin, dass Sie eine syntaktisch fehlerhafte Aktion speichern und diese nicht in Runtime zum Ablauf kommen wird.

Syntaktisch fehlerhafte Aktionen werden im Navigationsfenster mit nebenstehendem Symbol angezeigt.

Vorgehensweise

1. Betätigen Sie die Schaltfläche "Syntaxprüfung" in der Symbolleiste.
2. Wenn Fehler im unteren Teil des Editierfensters angezeigt werden, doppelklicken Sie auf die Fehlerzeile und korrigieren Sie den Fehler im Code. Wiederholen Sie Schritte 1 und 2, bis der Code fehlerfrei ist.
3. Speichern Sie die Aktion mit der Schaltfläche "Speichern" in der Symbolleiste.

Siehe auch

Aktionen (Seite 19)

So benennen Sie eine Aktion um (Seite 82)

So schützen Sie eine Aktion mit einem Passwort (Seite 67)

So fügen Sie aktionsbegleitende Informationen hinzu (Seite 65)

So bearbeiten Sie eine Aktion (Seite 61)

So legen Sie eine neue Aktion an (Seite 60)

Trigger (Seite 70)

Aktionen erstellen und bearbeiten (Seite 56)

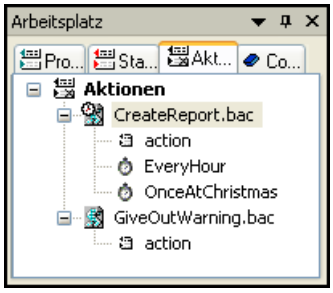
1.10.7 Trigger

1.10.7.1 Trigger

Definition und Verwendung

Trigger werden gebraucht, um Aktionen in Runtime auszuführen. Dazu wird ein Trigger mit einer Aktion verbunden und bildet somit das auslösende Ereignis für den Aufruf der Aktion. Aktionen ohne Trigger werden nicht ausgeführt.

Die an einer Aktion definierten Trigger werden Ihnen im Navigationsfenster von Global Script angezeigt.



Triggerarten

Folgende Trigger werden angeboten:

Azyklische Trigger

Sie bestehen aus der Angabe des Datums und der Uhrzeit. Die Aktion, die mit solch einem Trigger verbunden ist, wird einmal am angegebenen Datum zur angegebenen Uhrzeit ausgeführt.

Zyklische Trigger

Sie bestehen aus der Angabe eines Zeitintervalls und dessen Beginn. Folgende zyklische Trigger stehen zur Verfügung:

- Standardzyklus. Der Beginn des ersten Zeitintervalls fällt mit dem Start von Runtime zusammen. Die Länge des Intervalls wird durch den Zyklus bestimmt.
- Stündlich. Der Beginn des Zeitintervalls wird mit Minute und Sekunde festgelegt. Die Länge des Intervalls beträgt eine Stunde.
- Täglich. Der Beginn des Zeitintervalls wird mit der Uhrzeit (Stunde, Minute und Sekunde) festgelegt. Die Länge des Intervalls beträgt ein Tag.
- Wöchentlich. Der Beginn des Zeitintervalls wird mit Wochentag (Montag, Dienstag, ...) und Uhrzeit festgelegt. Die Länge des Intervalls beträgt eine Woche.

- Monatlich. Der Beginn des Zeitintervalls wird mit Tag und Uhrzeit festgelegt. Die Länge des Intervalls beträgt einen Monat.
- Jährlich. Der Beginn des Zeitintervalls wird mit Tag, Monat und Uhrzeit festgelegt. Die Länge des Intervalls beträgt ein Jahr.

Sie verwenden zeitgesteuerte Trigger für Aktionen im Global Script und für Aktionen zur Dynamisierung von Grafikobjekten.

Variablentrigger

Sie bestehen aus der Angabe einer oder mehrerer Variablen. Die Aktion, die mit solch einem Trigger verbunden ist, wird jedes Mal ausgeführt, wenn eine Änderung des Werts einer dieser Variablen festgestellt wurde.

Wie die Variablenwerte abgefragt werden, ist für jede Variable individuell einstellbar. Sie können wählen zwischen folgenden Modi:

- Zyklische Abfrage des Variablenwertes: Sie geben einen Standardzyklus an. In den gewählten Intervallen (z.B. alle 2 Sekunden) wird der Wert der Variablen abgefragt. Die Aktion wird ausgelöst, wenn vom System eine Änderung des Variablenwerts festgestellt wurde.
  Je nachdem, wie groß der Zyklus gewählt wird, kann es vorkommen, dass sich die Variable ändert, dies vom System aber nicht wahrgenommen wird.
  Wenn Sie z.B. einen Zyklus von 5 Minuten einstellen, und sich der Wert der Variablen in diesen 5 Minuten mehrfach ändert, wird nur der Wert wahrgenommen, der bei der nächsten Abfrage aktuell ist. Die Wertänderungen zwischen den beiden Abfragen werden nicht wahrgenommen.
- Bei Änderung des Variablenwertes: Jede Änderung des Variablenwertes wird vom System wahrgenommen. Die Aktion wird jedes Mal ausgeführt, wenn sich ihr Wert ändert.

Sie können Variablentrigger für Aktionen im Global Script verwenden und für Aktionen zur Dynamisierung von Grafikobjekten.

Ereignis-Trigger

Wenn Sie eine Aktion an ein Ereignis an einem Grafikobjekt projektieren, wird die Aktion ereignisgesteuert ausgelöst, z.B. bei Mausklick oder bei Änderung der Hintergrundfarbe durch eine andere Aktion.

Animationszyklus

Ab WinCC V7.0 steht Ihnen für die Dynamisierung von Objekten mit VBS die Triggerart "Animationszyklus" zur Verfügung. Der Animationszyklus erlaubt Ihnen in Runtime das Einschalten und Ausschalten von Aktionen und das Ändern der Zeit, in der der Trigger zur Ausführung kommt.

Weitere Informationen erhalten Sie im Kapitel "Animationstrigger".

Auswirkung von Triggern auf Aktionen

Ist die Aktion mit nur einem Trigger verbunden, dann wird die Aktion ausgeführt, sobald das auslösende Ereignis eintrifft.

Eine Aktion kann aber auch mit mehreren Triggern verbunden werden, beispielsweise mit einem zyklischen Trigger und mit einem Variablentrigger. Die Aktion wird immer dann ausgeführt, wenn eins dieser auslösenden Ereignisse eintrifft. Treffen zwei Ereignisse

gleichzeitig ein, dann wird die Aktion zweimal hintereinander ausgeführt. Ändern sich zwei im Trigger enthaltene Variablen zur selben Zeit, so wird die Aktion nur einmal durchlaufen.

Abarbeitung von Aktionen im Graphics Designer

Im Graphics Designer gelten folgende Regeln für die Abarbeitung von Aktionen:

- Es kann keine ereignisgetriggerte Aktion ausgeführt werden, solange eine andere ereignisgetriggerte Aktion läuft.
- Es kann keine zyklische/variablengetriggerte Aktion ausgeführt werden, solange eine andere zyklische/variablengetriggerte Aktion läuft.
- Die beiden Aktionstypen beeinflussen sich nicht untereinander: Eine ereignisgesteuerte Aktion kann auch ausgeführt werden, wenn gerade eine zyklische Aktion läuft.
- Werden Aktionen durch andere Aktionen in der Ausführung blockiert (z.B. eine zyklische Aktion durch eine variablengetriggerte Aktion), wird jede blockierte Aktion zum nächsten möglichen Durchführungszeitpunkt einmalig ausgeführt. Zyklische Aktionen laufen nach der einmaligen Ausführung in ihrem normalen Zyklus weiter.

Abarbeitung von Aktionen im Global Script

Bildunabhängige Aktionen aus Global Script werden in Runtime bei Auslösung nacheinander ausgeführt. Wird eine Aktion angestoßen, während gerade eine andere Aktion läuft, wird die zweite Aktion in einer Warteschleife gehalten, bis die Ausführung möglich ist.

Aktionen in Global Script und im Graphics Designer beeinflussen sich gegenseitig nicht.

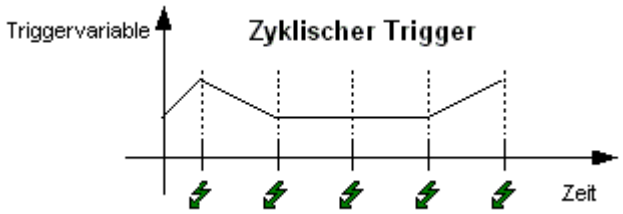
Hinweis

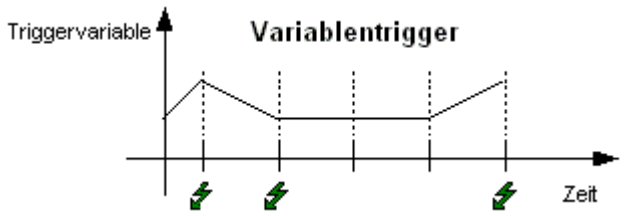
Soll die Aktion nicht bei jedem Ereignis ausgeführt werden, dann kann in der Aktion eine Bedingung formuliert werden, die abhängig vom Ergebnis die weitere Ausführung der Aktion steuert.

Hinweise zum Projektieren von Triggern

Systembedingt kann nicht garantiert werden, dass eine Aktion mit zyklischem Trigger genau zu den angegebenen Zeiten ausgeführt wird. Soll dies gewährleistet sein, dann ist die Aufgabe (z.B. Überprüfungen) im AG zu realisieren.

Der Variablentrigger sollte dem zyklischen Trigger vorgezogen werden: Bei zyklischen Aktionen wird die Aktion immer ausgeführt, z.B. alle 20 Sekunden. Der Variablentrigger führt die Aktion nur aus, wenn eine Wertänderung der Variablen bei zyklischer Abfrage festgestellt wurde. Dies reduziert die Systemlast und erhöht die Performance.





Wenn Sie einen Variablentrigger verwenden, projektieren Sie den Zyklus "Bei Änderung" so sparsam wie möglich. Dieser Abfragezyklus veranlasst, dass die Variable bei jeder Änderung eine Aktion triggert. Dies führt zu einer hohen Systembelastung.

Umverdrahten von Variablentriggern

Mit der CrossReference von WinCC können Sie alle Verwendungsstellen von Variablen auch in VBS-Aktionen schnell finden. Variablentrigger aus Aktionen im Graphics Designer können Sie mit CrossReference "umverdrahten", also an allen oder an ausgewählten Stellen durch andere Variablen ersetzen.

Hinweis

Sie können Variablen auch direkt im Graphics Designer umverdrahten, indem Sie das Grafikobjekt markieren und aus dem Kontextmenü den Befehl "Umverdrahten..." wählen.

Verwenden Sie für die Adressierung von Bildern und Variablen in Ihrem Code die Standardformulierungen
`HMIRuntime.BaseScreenName = "Screenname"` und
`HMIRuntime.Tags("Tagname")`, damit sichergestellt ist, dass Bilder und Variablen von der CrossReference erfasst werden.

Weiterführende Hinweise erhalten Sie in der WinCC-Dokumentation zu CrossReference.

Siehe auch

Aktionen (Seite 19)

So löschen Sie einen Trigger (Seite 80)

So ändern Sie einen Trigger (Seite 79)

So fügen Sie einen Trigger vom Typ "Variable" hinzu (Seite 77)

So fügen Sie einen Trigger vom Typ "Timer" hinzu (Seite 75)

Aktionen erstellen und bearbeiten (Seite 56)

1.10.7.2 Animationstrigger

Einleitung

Ab WinCC V7.0 steht für die Dynamisierung von Objekten mit VBS die Triggerart "Animationszyklus" zur Verfügung. Der Animationszyklus erlaubt in Runtime das Einschalten

und Ausschalten von Aktionen und das Ändern der Zeit, in der der Trigger zur Ausführung kommt.

Animationszyklen

Name	Zyklus	Name	Zyklus
CycleTime125ms	125 ms	CycleUser1	Anwenderzyklus 1
CycleTime250ms	250 ms	CycleUser2	Anwenderzyklus 2
CycleTime500ms	500 ms	CycleUser3	Anwenderzyklus 3
CycleTime1s	1 s	CycleUser4	Anwenderzyklus 4
CycleTime2s	2 s	CycleUser5	Anwenderzyklus 5
CycleTime5s	5 s	CyclePicture	Bildzyklus
CycleTime10s	10 s	CycleWindow	Fensterzyklus
CycleTime1min	1 min		
CycleTime5min	5 min		
CycleTime10min	10 min		
CycleTime1h	1 h		

Den Trigger nutzen Sie, indem sie eine Aktion schreiben und die Triggerart "Animationszyklus" verwenden. Mit den Methoden "ActivateDynamic" und "DeactivateDynamic" kann diese Aktion in Runtime aktiviert bzw. deaktiviert werden. Die Methoden sind in der VBS-Referenz im WinCC Information System beschrieben. Die korrekte Syntax der Methoden weicht von der Beschreibung in der VBS-Referenz ab und wird in den beiden folgenden Beispielen dargestellt.

Beispiel

Mit einer Aktion an der festgelegten Eigenschaft "Position X" (Left) wird das Rechteck um 5 Pixel nach rechts verschoben. Als Trigger wählen Sie in der Aktion das Ereignis "Animationszyklus".

Geben Sie in der Eigenschaft "Left" als Aktion folgendes ein:

```
item.Left = item.Left + 5
```

Mit den folgenden Methoden können sie die Aktion an der Eigenschaft "Position X" einschalten bzw. ausschalten.

Mit der Methode "ActivateDynamic" wird der Trigger in Runtime eingeschaltet:

```
Dim obj
Set obj = ScreenItems.Item("Rectangle1")
obj.ActivateDynamic "Left", "CycleTime1s"
```

Mit der Methode "DeactivateDynamic" wird der Trigger in Runtime ausgeschaltet:

```
Dim obj
Set obj = ScreenItems.Item("Rectangle1")
obj.DeactivateDynamic "Left"
```

Hinweis

Die WinCC-Variablen bleiben angefordert, auch wenn der Trigger ausgeschaltet wird.

Siehe auch

ActivateDynamic-Methode (Seite 687)

1.10.7.3 So fügen Sie einen Trigger vom Typ "Timer" hinzu

Einleitung

Trigger vom Typ "Timer" führen eine Aktion beim Eintreten eines bestimmten Zeitpunktes aus. Trigger vom Typ "Timer" können zyklische oder azyklische Trigger sein.

- Azyklische Trigger: Lösen eine Aktion einmalig bei Eintreten des projektierten Zeitpunktes aus.
- Zyklische Trigger: Lösen eine Aktion in regelmäßigen Zeitintervallen aus. Sie projektieren den Zeitintervall und den Startzeitpunkt des Triggers. Wenn Sie als zyklischen Trigger einen Standardzyklus wählen, ist der Startzeitpunkt immer der Start von Runtime. Als Standardzyklen können Sie auch anwenderspezifische Zyklen wählen.

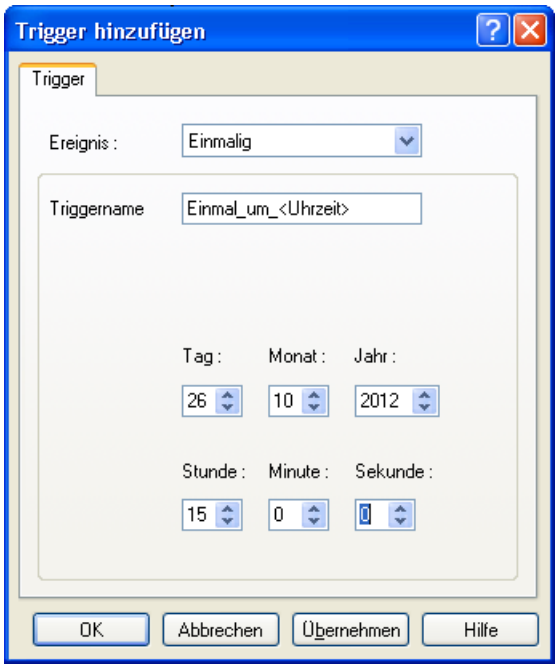
  Hinweis

  Zyklische Trigger gewährleisten eine hohe Aktualisierungsrate Ihres Systems, fordern aber eine hohe Systembelastung. Wählen Sie zyklische Trigger nur für Aktionen, bei denen die Aktualisierung sehr wichtig ist.
  Bei einer sehr hohen Systembelastung kann es vorkommen, dass Aktionen nicht mehr ausgeführt werden können.

Trigger vom Typ "Timer" verwenden Sie zum Dynamisieren von Eigenschaften im Graphics Designer und zum Ausführen von globalen Aktionen.

Vorgehensweise

1. Öffnen Sie die Aktion.
2. Betätigen Sie die Schaltfläche "Info/Trigger" in der Symbolleiste oder wählen Sie den Kontextmenübefehl "Info/Trigger". Der Dialog "Eigenschaften" erscheint.
3. Wählen Sie die Registerkarte "Trigger".
4. Wählen Sie als Trigger "Timer", und markieren Sie die Triggerart, die Sie erstellen möchten: zyklisch oder azyklisch.
5. Betätigen Sie die Schaltfläche "Hinzufügen". Der Dialog "Trigger hinzufügen" erscheint.
6. Wenn Sie die Triggerart "azyklisch" gewählt haben: Geben Sie einen sprechenden Triggernamen ein und stellen sie den Zeitpunkt ein, an dem die Aktion ausgeführt werden soll.



7. Wenn Sie die Triggerart "zyklisch" gewählt haben: Geben Sie einen sprechenden Triggernamen ein, und stellen Sie den Startzeitpunkt ein, an dem die Aktion das erste Mal ausgeführt werden soll. Geben Sie einen Zyklus ein, in dem die Aktion wiederholt werden soll.
   Bestätigen Sie Ihre Eingabe mit OK.

Hinweis

Sie können einer Aktion mehrere Trigger zuweisen. Die Aktion wird immer ausgeführt, wenn eines der Triggerereignisse eintritt.

Siehe auch

So löschen Sie einen Trigger (Seite 80)

So fügen Sie einen Trigger vom Typ "Variable" hinzu (Seite 77)

Trigger (Seite 70)

Aktionen erstellen und bearbeiten (Seite 56)

Aktionen (Seite 19)

1.10.7.4 So fügen Sie einen Trigger vom Typ "Variable" hinzu

Einleitung

Trigger vom Typ "Variable" führen eine Aktion bei Änderung eines Variablenwertes aus. Als Triggervariable können Sie jede in WinCC angelegte interne oder externe Variable verwenden.

Aktionen mit Variablentrigger können Sie zu folgenden Zeitpunkten ausführen lassen:

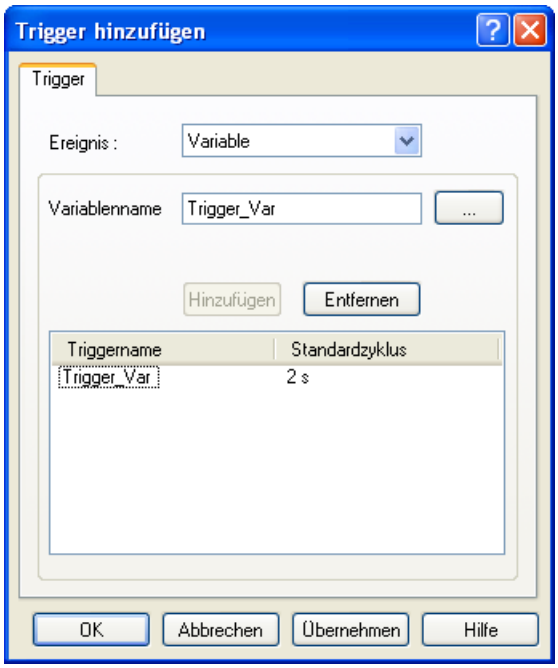
- Bei Änderung der Variable: Die Aktion wird bei jeder Wertänderung der Variable durchgeführt. Da diese Einstellung zu einer sehr hohen Systembelastung führt, sollten Sie diese Aktualisierungszeit möglichst sparsam einsetzen.
- Abfrage des Variablenstatus nach Standardzyklen (inklusive Anwenderzyklen): Sie definieren einen Zyklus, in dessen Intervallen der Wert der Variable abgefragt wird. Die Aktion wird nur ausgeführt, wenn sich der Wert der Variable bei Abfrage geändert hat. Wenn der Abfragezyklus groß gewählt wird, kann es vorkommen, dass sich der Wert der Variable ändert, ohne dass das vom System wahrgenommen wird. Die Aktion wird in diesem Fall nicht ausgeführt.

Wenn Sie eine Aktion mit mehreren Variablen verknüpfen, wird die Aktion ausgeführt, wenn eine der Variablen ihren Wert ändert.

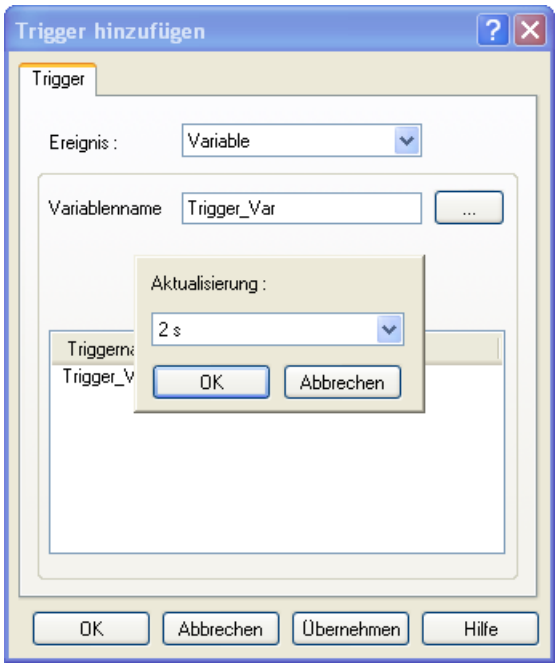
Vorgehensweise

1. Öffnen Sie die Aktion.
2. Betätigen Sie die Schaltfläche "Info/Trigger" in der Symbolleiste oder wählen Sie den Kontextmenübefehl "Info/Trigger". Der Dialog "Eigenschaften" erscheint.
3. Wählen Sie die Registerkarte "Trigger".
4. Wählen Sie als Trigger "Variable".
5. Betätigen Sie die Schaltfläche "Hinzufügen". Der Dialog "Trigger hinzufügen" erscheint.

6. Geben Sie den Namen der Variable ein, die Sie als Trigger verwenden möchten oder betätigen Sie die Schaltfläche neben dem Feld "Variablenname", um eine Variable aus dem Variablenauswahldialog auszuwählen.



7. Mit Doppelklick auf das Feld "Standardzyklus" öffnen Sie den Auswahldialog für den Aktualisierungszyklus der Variable:



Wählen Sie einen Zyklus aus und bestätigen Sie Ihre Eingabe mit OK.

Siehe auch

So löschen Sie einen Trigger (Seite 80)

So fügen Sie einen Trigger vom Typ "Variable" hinzu (Seite 77)

So fügen Sie einen Trigger vom Typ "Timer" hinzu (Seite 75)

Trigger (Seite 70)

Aktionen erstellen und bearbeiten (Seite 56)

Aktionen (Seite 19)

1.10.7.5 So ändern Sie einen Trigger

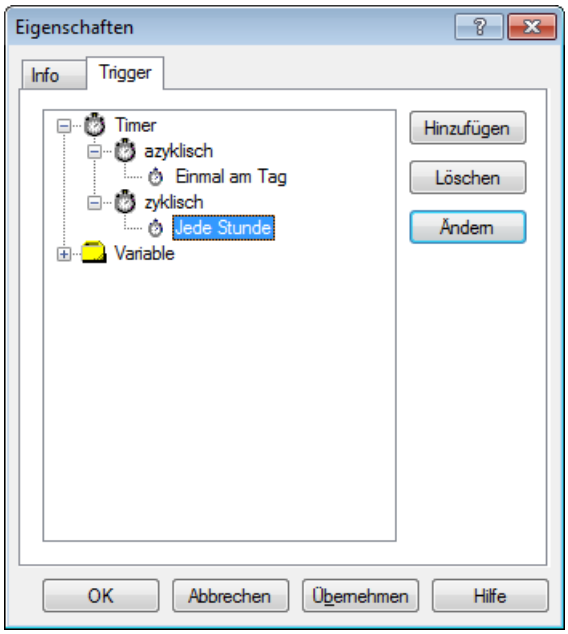
Einleitung

Sie können einmal definierte Trigger jederzeit ändern, auch während Runtime läuft.

Vorgehensweise

1. Öffnen Sie die Aktion, deren Trigger Sie ändern möchten.
2. Betätigen Sie die Schaltfläche "Info/Trigger" in der Symbolleiste
   oder wählen Sie den Kontextmenübefehl "Info/Trigger". Der Dialog "Eigenschaften" erscheint.
   Alternativ können Sie den Dialog aufrufen, ohne die Aktion zu öffnen, indem Sie den Trigger im Navigationsfenster doppelklicken.
3. Wählen Sie die Registerkarte "Trigger".

4. Wählen Sie den Trigger aus, den Sie ändern möchten, und betätigen Sie die Schaltfläche "Ändern".



5. Ändern Sie den Trigger und bestätigen Sie Ihre Eingaben mit OK.

Siehe auch

Trigger (Seite 70)

So löschen Sie einen Trigger (Seite 80)

So fügen Sie einen Trigger vom Typ "Variable" hinzu (Seite 77)

So fügen Sie einen Trigger vom Typ "Timer" hinzu (Seite 75)

Aktionen erstellen und bearbeiten (Seite 56)

Aktionen (Seite 19)

1.10.7.6 So löschen Sie einen Trigger

Einleitung

Festgelegte Trigger können jederzeit wieder gelöscht werden. Sie können Trigger auch löschen, während Runtime läuft.

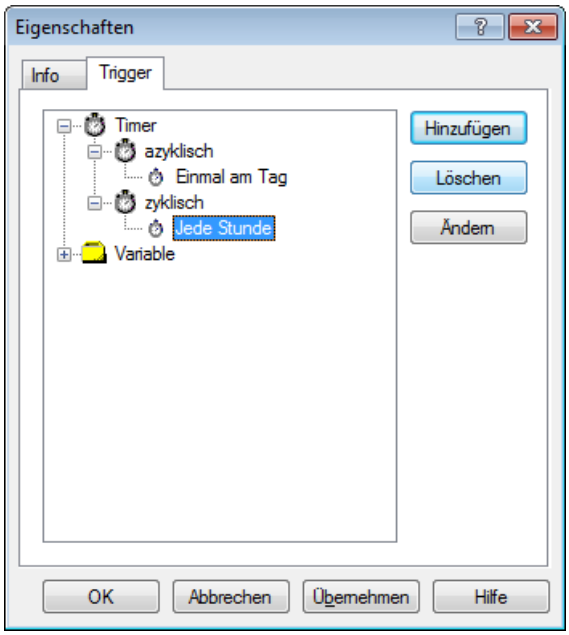
Wird in Runtime ein Trigger gelöscht, dann wirkt sich das erst nach Speichern der Aktion aus.

Hinweis

Eine Aktion ohne Trigger wird in Runtime nicht ausgeführt. Alle Aktionen, die den gelöschten Trigger verwendet haben, werden nicht mehr ausgeführt.

Vorgehensweise

1. Öffnen Sie den Global Script Editor oder den Aktionseditor im Graphics Designer.
2. Öffnen Sie die Aktion.
3. Betätigen Sie die Schaltfläche "Info/Trigger" in der Symbolleiste oder wählen Sie den Kontextmenübefehl "Info/Trigger". Der Dialog "Eigenschaften" erscheint.
4. Wählen Sie die Registerkarte "Trigger".
5. Wählen Sie den Trigger aus, den Sie löschen möchten, und betätigen Sie die Schaltfläche "Löschen".



6. Der Trigger wird sofort gelöscht.

Hinweis

Sie können Trigger auch direkt im Navigationsfenster von Global Script über den Kontextmenübefehl "Löschen" löschen.

Siehe auch

Aktionen (Seite 19)

So ändern Sie einen Trigger (Seite 79)

So fügen Sie einen Trigger vom Typ "Variable" hinzu (Seite 77)

So fügen Sie einen Trigger vom Typ "Timer" hinzu (Seite 75)

Trigger (Seite 70)

Aktionen erstellen und bearbeiten (Seite 56)

1.10.8 So benennen Sie eine Aktion um

Einleitung

Sie können eine Aktion in Global Script umbenennen. Beim Umbenennen einer Aktion werden der Aktionsname und der Dateiname geändert.

Die Aktion, die Sie umbenennen möchten darf nicht im Editierfenster geöffnet sein.

Vorgehensweise

1. Öffnen Sie Global Script.
2. Markieren Sie die Aktion, die Sie umbenennen möchten im Navigationsfenster des Editors.
3. Wählen Sie aus dem Kontextmenü den Befehl "Umbenennen".
4. Geben Sie einen neuen Namen für die Aktion mit der Endung *.bac ein.

Siehe auch

So schützen Sie eine Aktion mit einem Passwort (Seite 67)

So speichern Sie eine Aktion (Seite 68)

So fügen Sie aktionsbegleitende Informationen hinzu (Seite 65)

So bearbeiten Sie eine Aktion (Seite 61)

So legen Sie eine neue Aktion an (Seite 60)

Trigger (Seite 70)

Aktionen erstellen und bearbeiten (Seite 56)

Aktionen (Seite 19)

1.11 So aktivieren Sie globale Aktionen in Runtime

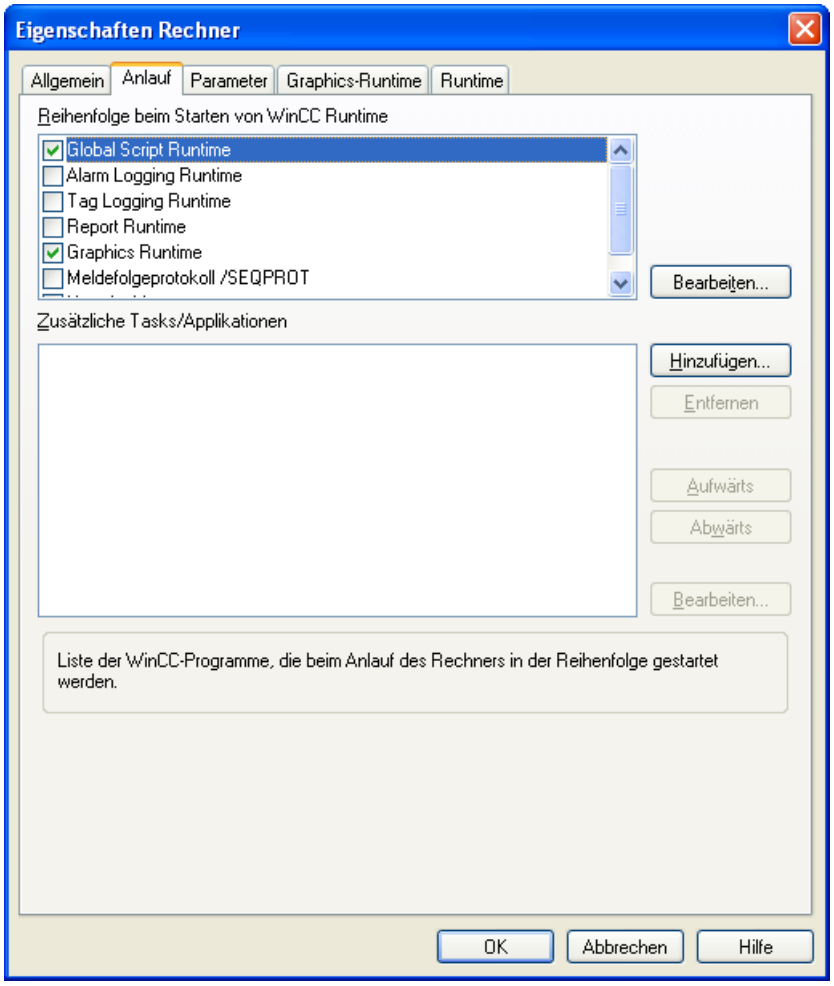
Einleitung

Im Global Script definierte Skripte werden in Runtime immer dann ausgeführt, wenn der projektierte Trigger eintritt. Skripte im grafischen Runtime-System werden ausgeführt, wenn das Bild aufgerufen wird und dann das projektierte Ereignis bzw. der Trigger eintritt.

Damit die bildunabhängigen, globalen Aktionen des Global Script ausgeführt werden, muss der Global Script Editor in die Anlaufliste des Runtime-Rechners aufgenommen werden.

Vorgehensweise

1. Wählen Sie im Kontextmenü des Rechners im WinCCExplorer den Befehl "Eigenschaften". Es öffnet sich der Dialog "Eigenschaften Rechner".
2. Wählen Sie die Registerkarte "Anlauf".
3. Aktivieren Sie Global Script Runtime.



4. Bestätigen Sie Ihre Eingabe mit OK.

Siehe auch

Aktionen erstellen und bearbeiten (Seite 56)
Prozeduren erstellen und bearbeiten (Seite 39)
Die VBScript-Editoren (Seite 28)
Einsatz von Visual Basic Script in WinCC (Seite 12)

1.12 Diagnose

1.12.1 Diagnose

Einleitung

Wenn Sie Ihre Skripte in Runtime ausführen und testen, können Sie mit Hilfe der Diagnosefenster schnell eine Analyse ausgeben.

Diagnose-Werkzeuge

WinCC stellt Ihnen Werkzeuge zur Verfügung, um das Laufzeitverhalten in Aktionen zu analysieren:

- Die Applikationsfenster GSC-Runtime und GSC-Diagnose
- Verwenden eines Debuggers

GSC-Runtime und GCS-Diagnose

Sie verwenden die Applikationsfenster GSC-Runtime und GSC-Diagnose, indem Sie sie in ein Prozessbild einfügen. Dies kann ein eigens zu Diagnosezwecken entworfenes Prozessbild sein, das in Runtime aufgerufen wird.

Mit diesen Applikationsfenstern werden unterschiedliche Strategien verfolgt:

GSC-Runtime gibt Auskunft über das dynamische Verhalten aller (Global Script-) Aktionen, ermöglicht den individuellen Start sowie das An- und Abmelden jeder einzelnen Aktion und bietet den Einsprung in den Global Script Editor, während Runtime aktiv ist.

GSC-Diagnose gibt die in den Aktionen enthaltenen Trace-Methoden in der zeitlichen Reihenfolge ihres Aufrufs aus. Das gilt auch für Trace-Anweisungen in Prozeduren, die in Aktionen aufgerufen werden. Durch gezielten Einsatz von Trace-Anweisungen, beispielsweise zur Ausgabe von Variablenwerten, lässt sich so der Ablauf von Aktionen und den darin aufgerufenen Prozeduren verfolgen. Die Trace-Anweisung geben Sie in der Form "HMIRuntime.Trace(<Ausgabe>)" an.

Im GSC-Diagnosefenster werden Trace-Ausgaben von C und VBS ausgegeben.

Hinweis

Laufzeitfehler in VBS werden nicht angezeigt

Einige Skript-Fehler werden weder über Trace ausgegeben noch über den Fehlerdialog angezeigt. Verwenden Sie den Microsoft Script Debugger.

Debugger

Zum Testen Ihrer Skripte in Runtime können Sie neben den Diagnosefenstern auch einen Debugger verwenden. Die Verwendung des Microsoft Script Debugger wird im Kapitel "Testen mit dem Debugger" beschrieben.

Den Microsoft Script Debugger finden Sie im Download-Center von Microsoft unter der folgenden URL:

- http://www.microsoft.com/downloads/Search.aspx?displaylang=en

Suchen Sie im Feld "Search" nach "Script Debugger" und wählen Sie den benötigten Download.

Siehe auch

Testen mit dem Debugger (Seite 92)

GSC-Runtime (Seite 89)

GSC-Diagnose (Seite 86)

Microsoft Download Center (http://www.microsoft.com/downloads/Search.aspx?displaylang=en)

1.12.2 GSC-Diagnose

1.12.2.1 GSC-Diagnose

Einleitung

GSC-Diagnose gibt die in den Aktionen enthaltenen Trace-Anweisungen in der zeitlichen Reihenfolge ihres Aufrufs im Diagnosefenster aus. Das gilt auch für Trace-Anweisungen in Prozeduren, die in Aktionen aufgerufen werden. Durch gezielten Einsatz von Trace-Anweisungen, beispielsweise zur Ausgabe von Variablenwerten, lässt sich so der Ablauf von Aktionen und den darin aufgerufenen Prozeduren verfolgen.

Anwendung

Damit Sie GSC-Diagnose verwenden können, fügen Sie ein Applikationsfenster vom Typ GSC-Diagnose in ein Prozessbild ein. Mit den Attributen von GSC-Diagnose können Sie das Erscheinungsbild des GSC-Diagnosefensters steuern.

Bei einem Bildwechsel wird der Inhalt des GSC-Diagnosefensters gelöscht.

Hinweis

Meldungen im Fenster "GSC-Diagnose" werden auch ausgegeben, wenn der Debugger aktiviert ist.

Siehe auch

Symbolleiste von GSC-Diagnose (Seite 88)

Attribute von GSC-Diagnose (Seite 88)

So fügen Sie das GSC-Diagnosefenster in ein Bild ein (Seite 87)

1.12.2.2 So fügen Sie das GSC-Diagnosefenster in ein Bild ein

Einleitung

Damit Sie GSC-Diagnose verwenden können, fügen Sie GSC-Diagnose in ein Prozessbild ein. Dieses Prozessbild kann ein vorhandenes Bild oder auch ein Bild sein, das eigens Diagnosezwecken dient. GSC-Diagnose lässt sich als Applikation nicht direkt in das Prozessbild einfügen, sondern wird als Anwendung in ein Applikationsfenster eingefügt. Das Applikationsfenster selbst ist dabei Bestandteil des Prozessbilds.

Voraussetzung

Der Graphics Designer ist gestartet und das Prozessbild ist geöffnet.

Vorgehensweise

1. Fügen Sie aus der Objektpalette "Smart-Objekte" das "Applikationsfenster" in Ihr Bild ein.
2. Wählen Sie aus dem Dialog "Fensterinhalt" den Eintrag "Global Script" und bestätigen Sie mit "OK".
3. Wählen Sie aus dem Dialog "Vorlage" den Eintrag "GSC-Diagnose".
4. Bestätigen Sie mit OK, um das Diagnosefenster einzufügen.

Siehe auch

Symbolleiste von GSC-Diagnose (Seite 88)

Attribute von GSC-Diagnose (Seite 88)

GSC-Diagnose (Seite 86)

1.12.2.3 Attribute von GSC-Diagnose

Übersicht

GSC-Diagnose besitzt Attribute, die das Erscheinungsbild des GSC-Diagnosefensters in Runtime beeinflussen. Dies sind die Geometrieattribute und insbesondere folgende Attribute:

- Anzeige: Mit diesem Attribut legen Sie fest, ob das Fenster sichtbar oder unsichtbar ist. Das Attribut ist mit dem Namen Visible dynamisierbar
- Größe veränderbar: Mit diesem Attribut bestimmen Sie, ob die Größe des Fensters in Runtime verändert werden kann
- Verschiebbar: Mit diesem Attribut legen Sie fest, ob das Fenster in Runtime verschiebbar ist
- Rahmen: Mit diesem Attribut bestimmen Sie, ob das Fenster einen Rahmen erhält. Hat das Fenster einen Rahmen, dann ist es in Höhe und Breite in Runtime veränderbar
- Titel: Hiermit legen Sie fest, ob das Fenster eine Titelleiste besitzt
- Maximierbar: Mit diesem Attribut bestimmen Sie, ob die Titelleiste die Schaltfläche zum Maximieren des Fensters enthält
- Schließbar: Mit diesem Attribut bestimmen Sie, ob die Titelleiste die Schaltfläche zum Schließen des Fensters enthält
- Vordergrund: Hiermit legen Sie fest, ob das Fenster immer im Vordergrund ist

Siehe auch

Symbolleiste von GSC-Diagnose (Seite 88)

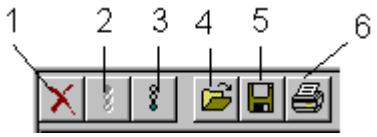
So fügen Sie das GSC-Diagnosefenster in ein Bild ein (Seite 87)

GSC-Diagnose (Seite 86)

1.12.2.4 Symbolleiste von GSC-Diagnose

Übersicht

Die Symbolleiste von GSC-Diagnose erlaubt die Steuerung der Ausgabe im Diagnosefenster sowie das Speichern, Drucken und Öffnen des Fensterinhalts:



1: Löscht den Inhalt des Diagnosefensters

2: Stoppt die Aktualisierung des Fensters

3: Aktiviert die Aktualisierung des Fensters

4: Öffnet eine Textdatei im Fenster

5: Speichert den Fensterinhalt in einer Textdatei

6: Druckt den Fensterinhalt

Siehe auch

Attribute von GSC-Diagnose (Seite 88)

So fügen Sie das GSC-Diagnosefenster in ein Bild ein (Seite 87)

GSC-Diagnose (Seite 86)

1.12.3 GSC-Runtime

1.12.3.1 GSC-Runtime

Einleitung

GSC-Runtime ist ein Fenster, welches das dynamische Verhalten aller Global Script-Aktionen in Runtime anzeigt. Darüber hinaus ermöglicht GSC-Runtime, Einfluss auf die Ausführung jeder einzelnen Aktion zu nehmen und bietet den Einsprung in den Global Script Editor, während Runtime aktiv ist.

Aktionen

Im GSC-Runtime Fenster werden C-Aktionen und VBS-Aktionen unterschieden:

-  Symbolisiert eine C-Aktion
-  Symbolisiert eine VBS-Aktion

Folgende Informationen werden ausgegeben:

- Aktionsname: Der Name der Aktion
- ID: Die Aktions ID. Sie wird systemintern verwendet. GSC-Runtime liefert zur Aktions ID den zugehörigen Aktionsnamen. Die Verbindung zwischen ID und Aktionsname ist nur solange gültig, bis Runtime beendet wird oder bei aktivem Runtime eine Aktion gespeichert wird.
- Status: Gibt Auskunft über den momentanen Status der Aktion. Die möglichen Stati entnehmen Sie bitte unten stehender Tabelle
- Aktivierungsabstand: Die Zeit in der Form Stunde:Minute:Sekunde, die zwischen zwei Aufrufen der Aktion verstrichen ist
- Rückgabe: Der Rückgabewert der Aktion
- gestartet am: Datum und Uhrzeit des aktuellen Starts der Aktion
- Nächster Start: Datum und Uhrzeit des nächsten Starts der Aktion
- Fehlermeldung: enthält den Fehlertext im Falle eines Fehlers

Aktions-Status

Die möglichen Stati einer Aktion sind:

- Aktion wurde angemeldet
- Aktion wurde abgemeldet
- Aktion wurde gestoppt
- Aktion läuft
- Fehler beim Anmelden der Aktion!
- Fehler beim Ausführen der Aktion!

Kontextmenü

Im Kontextmenü zu jeder Aktion finden Sie folgende Funktionen:

- abmelden: Die betreffende Aktion wird nach Ende der aktuellen Ausführung nicht mehr ausgeführt
- anmelden: Die betreffende Aktion wird mit dem nächsten Trigger wieder ausgeführt
- starten: Die betreffende Aktion wird einmal ausgeführt
- bearbeiten: Die betreffende Aktion wird im Global Script Editor zur Bearbeitung geöffnet. Runtime bleibt dabei aktiv. Wird die bearbeitete Aktion übersetzt (falls notwendig) und gespeichert, dann werden die Änderungen sofort vom Runtime System übernommen
  Die Möglichkeit, das Kontextmenü zu öffnen, können Sie für jede Aktion durch die Vergabe einer Berechtigung steuern.
  Damit Sie GSC-Runtime verwenden können, ist ein Applikationsfenster vom Typ GSC-Runtime in ein Prozessbild einzufügen. Mit den Attributen von GSC-Runtime können Sie das Erscheinungsbild des GSC-Runtime-Fensters steuern.

Hinweis

Die Aktualisierung des GSC-Runtime-Fensters erhöht die Systembelastung. Die Systembelastung ist abhängig davon, wie viele Aktionen im Fenster sichtbar sind. Die Systembelastung kann reduziert werden, wenn das Fenster in seiner Höhe verkleinert wird, sodass weniger Zeilen sichtbar sind.

Siehe auch

So fügen Sie das GSC-Runtime-Fenster in ein Bild ein (Seite 91)

Attribute von GSC-Runtime (Seite 91)

1.12.3.2 So fügen Sie das GSC-Runtime-Fenster in ein Bild ein

Einleitung

Damit Sie GSC-Runtime verwenden können, fügen Sie GSC-Runtime in ein Prozessbild ein. Dieses Prozessbild kann ein vorhandenes Bild oder auch ein Bild sein, das eigens Diagnosezwecken dient. GSC-Runtime lässt sich nicht direkt in das Prozessbild einfügen, sondern wird als Anwendung in ein Applikationsfenster eingefügt. Das Applikationsfenster selbst ist dabei Bestandteil des Prozessbilds.

Voraussetzung

Der Graphics Designer ist gestartet und das Prozessbild ist geöffnet.

Vorgehensweise

1. Fügen Sie aus der Objektpalette "Smart-Objekte" das "Applikationsfenster" in Ihr Bild ein.
2. Wählen Sie aus dem Dialog "Fensterinhalt" den Eintrag "Global Script" und bestätigen Sie mit "OK".
3. Wählen Sie aus dem Dialog "Vorlage" den Eintrag "GSC-Runtime".
4. Bestätigen Sie mit OK, um das Diagnosefenster einzufügen.

Siehe auch

GSC-Runtime (Seite 89)

Attribute von GSC-Runtime (Seite 91)

1.12.3.3 Attribute von GSC-Runtime

Übersicht

GSC-Runtime besitzt Attribute, die das Erscheinungsbild des GSC-Runtime-Fensters in Runtime beeinflussen. Dies sind die Geometrieattribute und insbesondere folgende Attribute:

- Anzeige: Mit diesem Attribut legen Sie fest, ob das Fenster sichtbar oder unsichtbar ist. Das Attribut ist mit dem Namen Visible dynamisierbar
- Größe veränderbar: Mit diesem Attribut bestimmen Sie, ob die Größe des Fensters in Runtime verändert werden kann
- Verschiebbar: Mit diesem Attribut legen Sie fest, ob das Fenster in Runtime verschiebbar ist
- Rahmen: Mit diesem Attribut bestimmen Sie, ob das Fenster einen Rahmen erhält. Hat das Fenster einen Rahmen, dann ist es in Höhe und Breite in Runtime veränderbar
- Titel: Hiermit legen Sie fest, ob das Fenster eine Titelleiste besitzt

- Maximierbar: Mit diesem Attribut bestimmen Sie, ob die Titelleiste die Schaltfläche zum Maximieren des Fensters enthält
- Schließbar: Mit diesem Attribut bestimmen Sie, ob die Titelleiste die Schaltfläche zum Schließen des Fensters enthält
- Vordergrund: Hiermit legen Sie fest, ob das Fenster immer im Vordergrund ist

Siehe auch

GSC-Runtime (Seite 89)

So fügen Sie das GSC-Runtime-Fenster in ein Bild ein (Seite 91)

1.12.4 Testen mit dem Debugger

1.12.4.1 Testen mit dem Debugger

Übersicht

Um Ihre VB-Skripte in Runtime zu testen, können Sie einen Debugger verwenden, z. B.:

- Microsoft Script Debugger
- Debugger "InterDev" (im Installationsumfang von Developer Studio enthalten)
- Microsoft Script Editor (MSE) Debugger (im Lieferumfang von Microsoft Office enthalten)

Im Folgenden wird ausschließlich der Umgang mit dem Microsoft Script Debugger beschrieben.

Download des Microsoft Script Debugger

Den Microsoft Script Debugger finden Sie im Download-Center von Microsoft unter der folgenden URL:

- http://www.microsoft.com/downloads/Search.aspx?displaylang=en833a6a92-961e-4ce1-9069-528d22605127

Suchen Sie im Feld "Search" nach "Script Debugger" und wählen Sie den benötigten Download.

Hinweise zum MSE Debugger

Wenn Sie den MSE Debugger verwenden, ändern Sie die folgenden Einstellungen, damit die laufenden Prozesse angezeigt werden:

1. Wählen Sie im Fenster "Prozesse" die Schaltfläche "Eigenschaften".
2. Aktivieren Sie im Dialog "Debugger-Eigenschaften" die Option "Just-In-Time-Debuggen".

3. Starten Sie den Rechner neu.
4. Deaktivieren Sie im MS Internet Explorer die Option "Disable script debugging", damit der Internet Explorer den Debugging-Vorgang von WinCC nicht verhindert.

Siehe auch

Grundlagen des Debuggens (Seite 95)

So führen Sie Skriptbefehle aus (Seite 110)

So ermitteln und ändern Sie Variablen- oder Eigenschaftenwerte (Seite 109)

So setzen Sie Lesezeichen im Skript (Seite 108)

So löschen Sie Haltepunkte (Seite 107)

So setzen Sie Haltepunkte (Seite 105)

So arbeiten Sie Skripte schrittweise ab (Seite 104)

So wählen Sie ein Skript zur Bearbeitung aus (Seite 103)

Aktions- und Prozedurnamen im Debugger (Seite 101)

Aufbau von VBScript-Dateien (Seite 99)

Komponenten des Microsoft Script Debuggers (Seite 97)

So aktivieren Sie den Debugger (Seite 93)

Diagnose (Seite 85)

Microsoft Download Center (http://www.microsoft.com/downloads/Search.aspx?displaylang=en)

1.12.4.2 So aktivieren Sie den Debugger

Prinzip

Sie haben mehrere Möglichkeiten, den Debugger zu aktivieren:

- Automatischer Aufruf des Debuggers bei Auftreten eines Fehlers in Runtime.
- Einblenden einer Fehlerbox in Runtime, über die der Debugger aufgerufen werden kann.
- Starten des Debuggers aus dem Startmenü und öffnen eines laufenden Runtime-Skriptes.

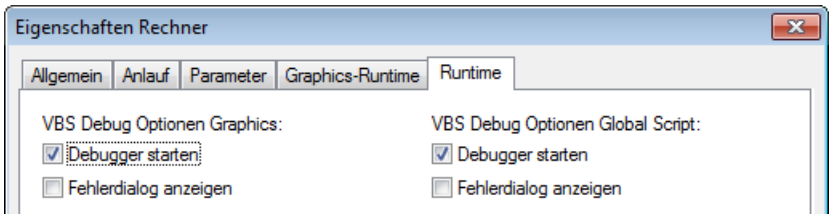
Voraussetzung

Der Microsoft Script Debugger muss auf dem Projektierungsrechner installiert sein.

Vorgehensweise

Die folgende Vorgehensweise beschreibt die ersten beiden Punkte, das Aktivieren des Debuggers in WinCC.

1. Wählen Sie im Kontextmenü des Rechners im WinCC Explorer den Befehl "Eigenschaften". Der Dialog "Eigenschaften Rechner" erscheint.
2. Wählen Sie die Registerkarte Runtime.
3. Aktivieren Sie die gewünschte Debug-Optionen. Sie können das Debug-Verhalten für Aktionen im Global Script und im Graphics Designer unabhängig voneinander einstellen:



4. Wählen Sie "Debugger starten", wenn bei Auftreten eines Fehlers in Runtime direkt der Debugger gestartet werden soll.
5. Wählen Sie "Fehlerdialog anzeigen", wenn nicht direkt der Debugger gestartet werden, sondern eine Fehlerbox mit Informationen zum Fehler erscheinen soll. Von der Fehlerbox aus können Sie über eine Schaltfläche den Debugger starten.
6. Bestätigen Sie Ihre Eingabe mit OK.

Debugger starten und ein laufendes Skript öffnen

Sie können den Debugger auch nachträglich starten und mit dem aktuell laufenden System verbinden. Stellen Sie dazu im Debugger eine Verbindung mit den jeweiligen Prozessen "pdlrt.exe" für das Graphische Runtime-System und "gscrt.exe" für das Globale Runtime-System her. Wie Sie ein laufendes Skript im Debugger öffnen, finden Sie unter dem Thema "Skript auswählen".

Beenden des Debuggers

Sie können den Debugger beenden, ohne dass WinCC-Runtime beendet wird.

Siehe auch

So setzen Sie Lesezeichen im Skript (Seite 108)

So führen Sie Skriptbefehle aus (Seite 110)

So ermitteln und ändern Sie Variablen- oder Eigenschaftenwerte (Seite 109)

So löschen Sie Haltepunkte (Seite 107)

So setzen Sie Haltepunkte (Seite 105)

So arbeiten Sie Skripte schrittweise ab (Seite 104)

So wählen Sie ein Skript zur Bearbeitung aus (Seite 103)

Aktions- und Prozedurnamen im Debugger (Seite 101)

Aufbau von VBScript-Dateien (Seite 99)

Komponenten des Microsoft Script Debuggers (Seite 97)

Grundlagen des Debuggens (Seite 95)

Testen mit dem Debugger (Seite 92)

Diagnose (Seite 85)

1.12.4.3 Grundlagen des Debuggens

Einführung

Zum Debuggen Ihrer VBScripte können Sie den Microsoft Script Debugger verwenden. Der Microsoft Script Debugger erlaubt Ihnen:

- Den Quellcode des Skripts, das Sie debuggen, zu sichten
- Die schrittweise Abarbeitung der Skripte zu kontrollieren
- Variablen und Eigenschaftenwerte anzuschauen und zu ändern
- Den Skript-Ablauf zu sehen und zu kontrollieren

Hinweis

Beachten Sie, das im Debugger Ihr Code schreibgeschützt dargestellt wird. Sie können den Code nicht direkt im Debugger ändern, sondern nur die erforderlichen Änderungen testen.

Fehlertypen

Folgende Fehlertypen werden beim Debuggen unterschieden:

Syntaxfehler

Syntaxfehler entstehen, wenn Sie z.B. ein Schlüsselwort falsch schreiben oder eine Klammer nicht schließen. Wenn Sie die Syntaxprüfung von WinCC einsetzen, können Sie Syntaxfehler bereits vor dem Test Ihrer Skripte in Runtime ausschließen. Im Graphics Designer können

prinzipiell nur syntaktisch korrekte Skripte gespeichert werden. Die Syntaxprüfung in WinCC prüft außerdem:

- Ob die Prozedurnamen im Global Script eindeutig sind
- Ob in Aktionsmodulen in Global Script nur eine Prozedur enthalten ist
- Ob im Aktionsteil im Graphics Designer nur eine Prozedur enthalten ist

Durch die Syntaxprüfung in WinCC wird das Skript geparst, ohne es auszuführen. Unmittelbar vor der Ausführung in Runtime wird das Skript nochmals geparst. Es werden alle Skriptteile geparst, auch diejenigen, die z.B. erst nach Eintreten einer bestimmten Aktion zu einem späteren Zeitpunkt ausgeführt werden.

Sind Syntaxfehler in Ihrem Skript enthalten, wird das Skript in Runtime nicht ausgeführt.

Laufzeitfehler

Ein Laufzeitfehler tritt auf, wenn eine ungültige/fehlerhafte Aktion ausgeführt werden soll, z.B. weil eine Variable nicht definiert ist. Um Laufzeitfehler abzufangen, können Sie in VBScript die Anweisung "On Error Resume Next" verwenden. Die Anweisung bewirkt, dass nach einem Laufzeitfehler die Folgeanweisung ausführt wird. In der Folgezeile kann dann der Fehlercode über das Err-Objekt überprüft werden. Um die Behandlung von Laufzeitfehlern im Skript wieder abzuschalten, verwenden Sie die Anweisung "On Error Goto 0".

Logische Fehler

Besonders hilfreich ist der Debugger zum Bereinigen von logischen Fehlern. Ein logischer Fehler tritt auf, wenn nicht das von Ihnen erwartete Ergebnis eintritt, weil z.B. eine Bedingung falsch überprüft wird. Um logische Fehler zu bereinigen, gehen Sie Ihr Skript Schritt für Schritt durch, um den nicht funktionierenden Teil des Skriptes zu identifizieren.

Grundlegendes Vorgehen

Wenn ein Fehler aufgetreten ist, und der Debugger geöffnet ist, erscheint das Skript schreibgeschützt in einem Fenster. Sie können durch das Skript-Dokument navigieren, Haltepunkte setzen, das Skript erneut in Runtime ausführen, und die Skripte schrittweise abarbeiten.

Die wichtigsten Schritte zum erfolgreichen Debuggen Ihrer Skripte finden Sie unter "Skripte schrittweise abarbeiten".

Sie können den Quellcode Ihrer Skripte nicht direkt im Debugger editieren. Wenn Sie einen Fehler gefunden haben, können Sie den Fehler im Original Skript in WinCC korrigieren, z.B. das Bild erneut laden und es im Debugger aktualisieren.

Hinweis

Tipps und Tricks zum Debuggen, häufig auftretende Fehlerquellen und andere Hinweise finden Sie in der Online-Hilfe des Microsoft Skript Debuggers.

Bildwechsel beim Debuggen

Wenn Sie während des Debuggens einen Bildwechsel durchführen, bleibt das Skriptdokument des "alten" Bildes geöffnet, ist aber nicht mehr gültig. Es werden ggf. ungültige Fehler angezeigt, da die aufgerufenen Objekte nach dem Bildwechsel nicht mehr vorhanden sind.

Siehe auch

Testen mit dem Debugger (Seite 92)

So führen Sie Skriptbefehle aus (Seite 110)

So ermitteln und ändern Sie Variablen- oder Eigenschaftenwerte (Seite 109)

So setzen Sie Lesezeichen im Skript (Seite 108)

So löschen Sie Haltepunkte (Seite 107)

So setzen Sie Haltepunkte (Seite 105)

So arbeiten Sie Skripte schrittweise ab (Seite 104)

So wählen Sie ein Skript zur Bearbeitung aus (Seite 103)

Aktions- und Prozedurnamen im Debugger (Seite 101)

Aufbau von VBScript-Dateien (Seite 99)

Komponenten des Microsoft Script Debuggers (Seite 97)

So aktivieren Sie den Debugger (Seite 93)

Diagnose (Seite 85)

1.12.4.4 Komponenten des Microsoft Script Debuggers

Einführung

Der Microsoft Script Debugger bietet Ihnen mehrere Komponenten, die beim Debuggen hilfreich sind:

Fenster "Command Window"

Das "Command Window" rufen Sie über den Menübefehl "View" > "Command Window" auf.

Während ein Skript in Runtime läuft, können Sie das "Command Window" des Debuggers verwenden, um z.B. Werte von Variablen und Eigenschaften des laufenden Skriptes zu ermitteln und zu ändern. Änderungen, die Sie im "Command Window" durchführen, wirken sich direkt auf das laufende Skript aus, so dass Sie geplante Änderungen direkt testen können.

Folgende Aktionen können Sie im "Command Window" durchführen:

- Befehle eingeben: Sie können direkt Befehle in VBScript eingeben und ausführen
- Variablenwerte ändern: Im "Command Window" können Sie Variablenwerte ermitteln und direkt ändern. Dies betrifft sowohl Variablen im aktuellen Skript als auch globale Variablen.
- Eigenschaften ändern: Sie können die Eigenschaften von allen Objekten des aktuellen Skript-Kontextes sowohl lesen als auch schreiben.

Sie können das "Command Window" immer verwenden, wenn ein Skript an einem Haltepunkt angelangt ist, oder Sie von einem Haltepunkt zu weiteren Befehlen gegangen sind.

Hinweis

Beachten Sie, dass sich die Änderungen, die Sie im "Command Window" durchführen nicht auf den Quellcode Ihres Skriptes auswirken, sondern nur im Debugger zu Testzwecken dienen.

Fenster "Running Documents"

Das Fenster "Running Documents" rufen Sie über den Menübefehl "View" > "Running Documents" auf.

In diesem Fenster sehen Sie alle aktuell in WinCC-Runtime laufenden Skripte, getrennt nach Skripten aus Global Script ("Global Script Runtime") und Skripten des grafischen Runtime-Systems ("PDLRT"). Sie sehen jeweils alle laufenden Aktionen und Module des Global Script Runtime. Im Grafischen Runtime System sind die Skripte getrennt nach triggergesteuerten Aktionen (Bildname_trigger) und eventgesteuerten Aktionen (Bildname_events).

Fenster "Call Stack"

Das Fenster "Call Stack" rufen Sie über den Menübefehl "View" > "Call Stack" auf.

In diesem Fenster sehen Sie eine Liste aller laufenden Aktionen und aufgerufenen Prozeduren. Wenn z.B. eine Prozedur aufgerufen wird, wird der Name der "Call Stack"-Liste hinzugefügt. Ist die Prozedur abgeschlossen, wird der Name wieder aus der Liste entfernt. Sie können eine Prozedur aus der Liste auswählen, um an die entsprechende Stelle im Skript-Dokument zu springen, an der die Prozedur aufgerufen wurde.

Siehe auch

So führen Sie Skriptbefehle aus (Seite 110)

So ermitteln und ändern Sie Variablen- oder Eigenschaftenwerte (Seite 109)

So setzen Sie Lesezeichen im Skript (Seite 108)

So löschen Sie Haltepunkte (Seite 107)

So setzen Sie Haltepunkte (Seite 105)

So arbeiten Sie Skripte schrittweise ab (Seite 104)

So wählen Sie ein Skript zur Bearbeitung aus (Seite 103)

Aktions- und Prozedurnamen im Debugger (Seite 101)

Aufbau von VBScript-Dateien (Seite 99)

Grundlagen des Debuggens (Seite 95)

So aktivieren Sie den Debugger (Seite 93)

Testen mit dem Debugger (Seite 92)

Diagnose (Seite 85)

1.12.4.5 Aufbau von VBScript-Dateien

Prinzip

Um die gleichzeitige Abarbeitung von zyklischen und ereignisgesteuerten Skripten im Grafischen Runtime-System nicht zu behindern, sind ereignisgesteuerte Aktionen und zyklische/variablengetriggerte Aktionen bei der Abarbeitung streng getrennt. So kann z.B. eine zyklische Aktion nicht die Ausführung einer Aktion behindern, die beim Drücken einer Schaltfläche ausgeführt werden soll.

Um dies zu gewährleisten, werden beim Speichern eines Bildes die ereignisgesteuerten Aktionen und die zyklischen- sowie die variablengetriggerten Aktionen in getrennten Skript-Dateien abgelegt. Haben Sie in Aktionen im Graphics Designer einen bildglobalen Teil definiert, wird dieser in beide Skript-Dateien kopiert. Ebenso werden die Module, die in den Aktionen verwendet werden, in beide Skript-Dateien kopiert.

Wenn eine Variable aus einem Modul verwendet werden soll, muss das entsprechende Modul aufgerufen werden. Das Modul wird sonst nicht in die Skript-Datei kopiert und es wird ein Fehler erzeugt.

Hinweis

Da beide Skript-Dateien getrennt behandelt werden, haben sie keinen gemeinsamen Datenbereich. Es erfolgt daher kein Abgleich globaler Variablen zwischen den beiden Skript-Dateien. Ist eine Synchronisation erforderlich, müssen Sie diese über das DataSet-Objekt oder interne WinCC-Variablen realisieren.

Aufbau der Skript-Dateien

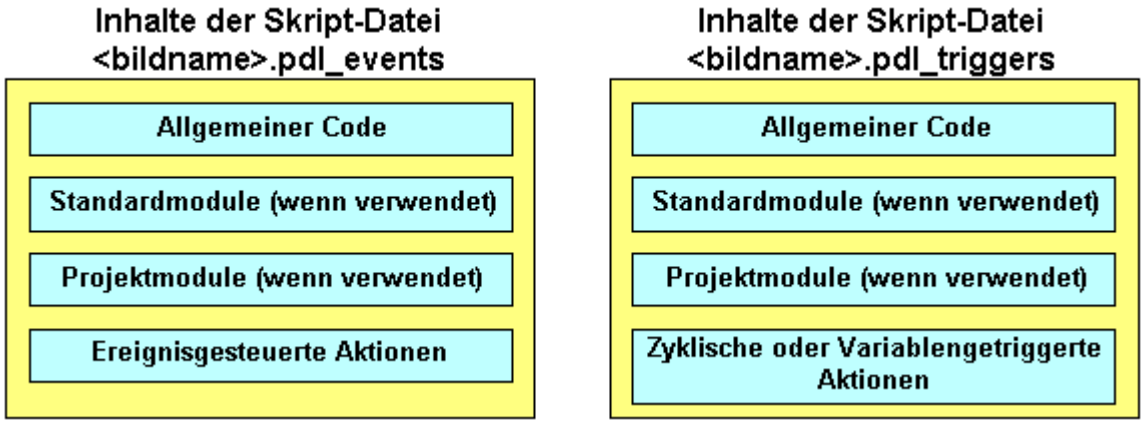
Wenn Sie Skripte mit dem Debugger debuggen, öffnen sich immer die Skript-Dateien der unterschiedlichen Runtime-Systeme.

Für das Grafische Runtime-System bedeutet dies, dass Sie pro Bild zwei Skript-Dateien erhalten:

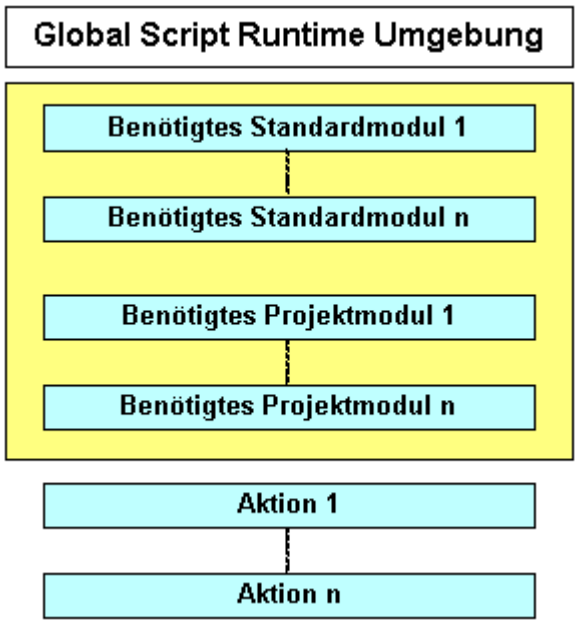
- <Bildname>.pdl_events: Enthält die ereignisgesteuerten Aktionen
- <Bildname>.pdl_triggers: Enthält die zyklischen und variablengesteuerten Aktionen.

Im Folgenden sehen Sie, wie die Skript-Dateien aufgebaut sind:

Grafisches Runtime System



Global Script Runtime



Hinweis

Beachten Sie, dass die Aktionen und Prozeduren des Grafischen Runtime-Systems nicht mit den Aktionsnamen in der Skript-Datei angezeigt werden, unter der Sie sie in WinCC gespeichert haben. Die Namenskonventionen für Aktionen und Prozeduren in den Skript-Dateien finden Sie unter "Aktions- und Prozedurnamen im Debugger".

Siehe auch

So aktivieren Sie den Debugger (Seite 93)
So führen Sie Skriptbefehle aus (Seite 110)
So ermitteln und ändern Sie Variablen- oder Eigenschaftenwerte (Seite 109)
So setzen Sie Lesezeichen im Skript (Seite 108)
So löschen Sie Haltepunkte (Seite 107)
So setzen Sie Haltepunkte (Seite 105)
So arbeiten Sie Skripte schrittweise ab (Seite 104)
So wählen Sie ein Skript zur Bearbeitung aus (Seite 103)
Aktions- und Prozedurnamen im Debugger (Seite 101)
Komponenten des Microsoft Script Debuggers (Seite 97)
Grundlagen des Debuggens (Seite 95)
Testen mit dem Debugger (Seite 92)
Diagnose (Seite 85)

1.12.4.6 Aktions- und Prozedurnamen im Debugger

Aktions- und Prozedurnamen im Debugger

Die Namen von Prozeduren und Aktionen in den Skript-Dateien im Debugger unterscheiden sich von den Namen, unter denen Sie Ihre Skripte in WinCC gespeichert haben.

Die Aktions- und Prozedurnamen in den Skript-Dateien werden nach folgenden Regeln zusammengesetzt:

Aktionstyp	Name in der Skript-Datei
Zyklische oder variablengetriggerte Aktionen an einer Eigenschaft	ObjektName_Eigenschaftsname_Trigger
Maus-Ereignisse	ObjektName_OnClick ObjektName_OnLButtonDown ObjektName_OnLButtonUp ObjektName_OnRButtonDown ObjektName_OnRButtonUp
Tastatur-Ereignisse	ObjektName_OnKeyDown ObjektName_OnKeyUp
Objekt-Ereignisse	ObjektName_OnObjectChanged ObjektName_OnSetFocus

Aktionstyp	Name in der Skript-Datei
Ereignisse an Eigenschaften	ObjektName_PropertyName_OnPropertyChanged ObjektName_PropertyName_OnPropertyStateChanged
Bildereignisse	Document_OnOpen Document_OnClosed

Zugelassene Länge der Aktionsnamen

Die Namen der Aktionen in den Skript-Dateien sind auf 255 Zeichen beschränkt. Jedes Sonderzeichen, das Sie in einem Objektnamen verwenden, wird in fünf Zeichen umgesetzt. Hinter einem führenden X wird das Sonderzeichen in vierstelligem Headezimalcode dargestellt. Wenn Sie z.B. auf Mausklick eine Aktion an eine Schaltfläche mit dem Namen "DrückMich" projektieren, erscheint das Skript in der Skript-Datei als "DrXFFFCckMich_OnClick".

Wird der zusammengesetzte Objektname zu lang, wird bei der Syntaxprüfung in WinCC eine Fehlermeldung ausgegeben. Aufgrund dieser Einschränkung können Sie Ihre Grafikobjekte während der Projektierung nicht beliebig lang wählen.

Hinweis

Wenn Sie den Namen eines Objektes in Runtime ermitteln möchten, drücken Sie <Strg+Alt +Shift> und fahren mit der Maus über das entsprechende Objekt. Der Bildname und der Objektname werden dann in einem Tooltip angezeigt.

Siehe auch

So führen Sie Skriptbefehle aus (Seite 110)

So ermitteln und ändern Sie Variablen- oder Eigenschaftenwerte (Seite 109)

So setzen Sie Lesezeichen im Skript (Seite 108)

So löschen Sie Haltepunkte (Seite 107)

So setzen Sie Haltepunkte (Seite 105)

So arbeiten Sie Skripte schrittweise ab (Seite 104)

So wählen Sie ein Skript zur Bearbeitung aus (Seite 103)

Aufbau von VBScript-Dateien (Seite 99)

Komponenten des Microsoft Script Debuggers (Seite 97)

Grundlagen des Debuggens (Seite 95)

So aktivieren Sie den Debugger (Seite 93)

Testen mit dem Debugger (Seite 92)

Diagnose (Seite 85)

1.12.4.7 So wählen Sie ein Skript zur Bearbeitung aus

Einführung

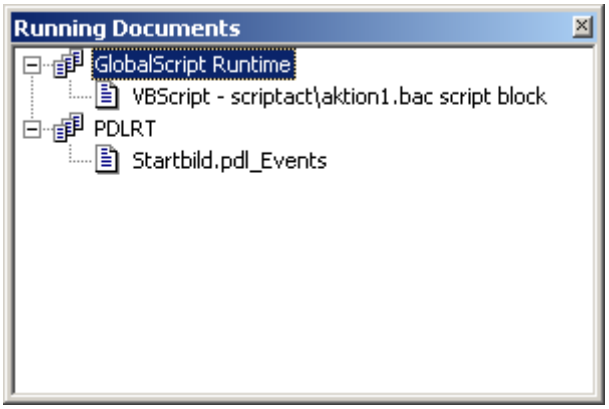
Wenn Sie statt der automatischen Aktivierung in WinCC den Microsoft Script Debugger aus den Windows-Startmenü aufrufen, während Runtime läuft, können Sie ein laufendes Skript zur Bearbeitung auswählen.

Voraussetzung

Runtime ist aktiviert, das Bild, das Sie debuggen möchten, ist aktiv.

Vorgehensweise

1. Starten Sie den Debugger aus dem Windows-Startmenü ("Start" > "Programme" > "Zubehör" > "Microsoft Script Debugger").
2. Aktivieren Sie in der Menüleiste den Befehl "View" > "Running Dokuments".
   Das Fenster "Running Documents" wird geöffnet. In diesem Fenster sehen Sie alle aktuell in WinCC-Runtime laufenden Skripte, getrennt nach Skripten aus Global Script ("Global Script Runtime") und Skripten des grafischen Runtime-Systems ("PDLRT"):



   Im Beispiel oben sind ist das "Startbild.pdl" aktiv, und es sind im Startbild nur ereignisgesteuerte Skripte vorhanden.
3. Doppelklicken Sie im Fenster "Running Documents" das Skript-Dokument, das Sie debuggen möchten.
   Das Skript-Dokument wird schreibgeschützt im Fenster des Debuggers geöffnet.

Siehe auch

So führen Sie Skriptbefehle aus (Seite 110)

So ermitteln und ändern Sie Variablen- oder Eigenschaftenwerte (Seite 109)

So setzen Sie Lesezeichen im Skript (Seite 108)

So löschen Sie Haltepunkte (Seite 107)

So setzen Sie Haltepunkte (Seite 105)

So arbeiten Sie Skripte schrittweise ab (Seite 104)

Aktions- und Prozedurnamen im Debugger (Seite 101)

Aufbau von VBScript-Dateien (Seite 99)

Komponenten des Microsoft Script Debuggers (Seite 97)

Grundlagen des Debuggens (Seite 95)

So aktivieren Sie den Debugger (Seite 93)

Testen mit dem Debugger (Seite 92)

Diagnose (Seite 85)

1.12.4.8 So arbeiten Sie Skripte schrittweise ab

Einführung

Sie können den Microsoft Script Debugger verwenden, um Ihre Skripte Schritt für Schritt abzuarbeiten und so z.B. logische Fehler systematisch einzugrenzen. So können Sie die Auswirkungen jeder einzelnen Skriptzeile in Runtime testen.

Prinzipielle Vorgehensweise

1. Aktivieren Sie das zu debuggende Dokument in Runtime.
2. Starten Sie den Debugger manuell aus dem Startmenü und öffnen die gewünschte Skript-Datei, oder aktivieren Sie den Debugger in WinCC. Bei der Aktivierung in WinCC öffnet sich der Debugger automatisch, wenn ein fehlerhaftes Skript ausgeführt wird.
3. Setzen Sie in der Skript-Datei einen Haltepunkt. Haltepunkte setzen Sie typischerweise vor Codezeilen, in denen Sie Fehler vermuten.
4. Wechseln Sie zu WinCC Runtime und lösen Sie eine Aktion aus, die das Skript zum Ablaufen bringt.
   Der Debugger hält am ersten Haltepunkt und markiert die aktuelle Zeile.

5. Um schrittweise durch das Skript-Dokument zu gehen, wählen Sie einen der folgenden Menübefehle:
   "Debug" > "Step Into": Geht zur nächsten Codezeile. Wenn das Skript in dieser Zeile eine Prozedur aufruft, springen Sie mit dem Befehl "Step Into" zu dieser Prozedur. Sie können die gerufenen Prozedur dann schrittweise abarbeiten.

   "Debug" > "Step Over": Überspringt die gerufene Prozedur. Die Prozedur wird ausgeführt, aber der Debugger führt Sie nicht durch die einzelnen Zeilen der Prozedur. Stattdessen führt er Sie zur nächsten Zeile des aktuellen Skriptes, nachdem die Prozedur ausgeführt wurde.
6. Um das schrittweise Abarbeiten einer Prozedur abzubrechen, wählen Sie den Menübefehl "Debug" > "Step Out". Der Debugger springt dann zur nächsten Aktion.
7. Gehen Sie schrittweise bis zum Ende des Dokumentes, oder wählen Sie den Menübefehl "Debug" > "Run", um das Skript in Runtime erneut zu starten.

Siehe auch

Grundlagen des Debuggens (Seite 95)

So führen Sie Skriptbefehle aus (Seite 110)

So ermitteln und ändern Sie Variablen- oder Eigenschaftenwerte (Seite 109)

So setzen Sie Lesezeichen im Skript (Seite 108)

So löschen Sie Haltepunkte (Seite 107)

So setzen Sie Haltepunkte (Seite 105)

So wählen Sie ein Skript zur Bearbeitung aus (Seite 103)

Aktions- und Prozedurnamen im Debugger (Seite 101)

Aufbau von VBScript-Dateien (Seite 99)

Komponenten des Microsoft Script Debuggers (Seite 97)

So aktivieren Sie den Debugger (Seite 93)

Testen mit dem Debugger (Seite 92)

Diagnose (Seite 85)

1.12.4.9 So setzen Sie Haltepunkte

Einführung

Sie können in einem Skript Haltepunkte setzen, um an bestimmten Stellen die Abarbeitung des Skriptes zu stoppen und den Debugger zu starten. Setzen Sie z.B. einen Haltepunkt vor eine Zeile, in der Sie einen Fehler im Skript vermuten.

Sie können:

- Haltepunkte an bestimmten Zeilen setzen, um logische Fehler in Ihrem Skript Schritt für Schritt einzugrenzen.
- Einen Haltepunkt setzen und den Debugger aufrufen, bevor die nächste Zeile des Skriptes abgearbeitet wird. Dieses Vorgehen verwenden Sie z.B. für Ereignisse wie "Bildwechsel".

Wenn eine Skript-Datei im Debugger aktualisiert wird, gehen alle Haltepunkte verloren.

Setzen Sie in einer der Skript-Dateien "<Bildname>.pdl_trigger" bzw. "<Bildname>.pdl_event einen Haltepunkt, werden in Runtime alle triggergesteuerten bzw. alle eventgesteuerten Prozeduren angehalten.

Voraussetzung

Runtime ist aktiviert, das zu debuggende Bild ist aktiv.

Vorgehensweise

Einen Haltepunkt setzen

1. Starten Sie den Debugger und wählen Sie das Skript aus. Wenn Sie die automatische Aktivierung des Debuggers in WinCC gewählt haben, wird der Debugger aufgerufen, sobald ein fehlerhaftes Skript ausgeführt wird.
2. Setzen Sie den Cursor in die Aktion, in der Sie einen Haltepunkt setzen möchten.
3. Wählen Sie aus dem Menü "Debug" den Eintrag "Toggle Breakpoint" oder das Symbol  aus der Symbolleiste.
   Die nächste ausführbare Zeile wird mit einem roten Punkt markiert.
4. Wechseln Sie in WinCC Runtime und führen Sie die Aktion aus, die Sie debuggen möchten. Der Debugger stoppt am ersten Haltepunkt, den er im Skript findet. Die aktuelle Zeile wird gelb hinterlegt dargestellt. Sie können nun das Skript schrittweise durchgehen.

Siehe auch

So löschen Sie Haltepunkte (Seite 107)

So führen Sie Skriptbefehle aus (Seite 110)

So ermitteln und ändern Sie Variablen- oder Eigenschaftenwerte (Seite 109)

So setzen Sie Lesezeichen im Skript (Seite 108)

So setzen Sie Haltepunkte (Seite 105)

So wählen Sie ein Skript zur Bearbeitung aus (Seite 103)

Aktions- und Prozedurnamen im Debugger (Seite 101)

Aufbau von VBScript-Dateien (Seite 99)

Komponenten des Microsoft Script Debuggers (Seite 97)

Grundlagen des Debuggens (Seite 95)

So aktivieren Sie den Debugger (Seite 93)

Testen mit dem Debugger (Seite 92)

Diagnose (Seite 85)

1.12.4.10 So löschen Sie Haltepunkte

Einführung

Wenn Sie einen Fehler erfolgreich beseitigt haben, können Sie einzelne oder auch alle Haltepunkte in einem Skript wieder löschen.

Vorgehensweise

1. Setzen Sie den Cursor in die Zeile, dessen Haltepunkt Sie löschen möchten.
2. Wählen Sie aus dem Menü "Debug" den Eintrag "Toggle Breakpoint" oder das Symbol  aus der Symbolleiste.
   Die Zeile wird wieder ohne Markierung angezeigt.
3. Um alle Haltepunkte in einem Skript zu löschen, wählen Sie aus dem Menü "Debug" den Eintrag "Clear all Breakpoints" oder das Symbol  aus der Symbolleiste.

Siehe auch

So führen Sie Skriptbefehle aus (Seite 110)
So ermitteln und ändern Sie Variablen- oder Eigenschaftenwerte (Seite 109)
So setzen Sie Lesezeichen im Skript (Seite 108)
So setzen Sie Haltepunkte (Seite 105)
So wählen Sie ein Skript zur Bearbeitung aus (Seite 103)
Aktions- und Prozedurnamen im Debugger (Seite 101)
Aufbau von VBScript-Dateien (Seite 99)
Komponenten des Microsoft Script Debuggers (Seite 97)
Grundlagen des Debuggens (Seite 95)
So aktivieren Sie den Debugger (Seite 93)
Testen mit dem Debugger (Seite 92)
Diagnose (Seite 85)

1.12.4.11 So setzen Sie Lesezeichen im Skript

Einführung

Während des Debuggens können Sie Lesezeichen an Codezeilen setzen, um eine Zeile später leichter wiederzufinden.

Lesezeichen setzen oder löschen

Setzen Sie dazu den Mauszeiger in die Zeile, in der Sie ein Lesezeichen setzen möchten, und drücken Sie <Strg+F2> um ein Lesezeichen zu setzen oder zu löschen.

Zum nächsten Lesezeichen springen

Mit <F2> gelangen Sie zum nächsten Lesezeichen im Skript.

Zum vorherigen Lesezeichen springen

Mit <Shift+F2> gelangen Sie zum vorhergehenden Lesezeichen im Skript.

Siehe auch

So führen Sie Skriptbefehle aus (Seite 110)
So ermitteln und ändern Sie Variablen- oder Eigenschaftenwerte (Seite 109)
So löschen Sie Haltepunkte (Seite 107)
So setzen Sie Haltepunkte (Seite 105)
So wählen Sie ein Skript zur Bearbeitung aus (Seite 103)
Aktions- und Prozedurnamen im Debugger (Seite 101)
Aufbau von VBScript-Dateien (Seite 99)
Komponenten des Microsoft Script Debuggers (Seite 97)
Grundlagen des Debuggens (Seite 95)
So aktivieren Sie den Debugger (Seite 93)
Testen mit dem Debugger (Seite 92)
Diagnose (Seite 85)

1.12.4.12 So ermitteln und ändern Sie Variablen- oder Eigenschaftenwerte

Einführung

Während ein Skript in Runtime läuft, können Sie das "Command Window" des Debuggers verwenden, um Werte von Variablen oder Eigenschaften des laufenden Skriptes zu ermitteln und zu ändern. So können Sie z.B. einen Prozesswert für das Skript wieder auf Null setzen, ohne den Prozess anhalten zu müssen.

Hinweis

Wenn Sie den Namen eines WinCC-Objektes in Runtime ermitteln möchten, drücken Sie <Strg+Alt+Shift> und fahren mit der Maus über das entsprechende Objekt. Der Bildname und der Objektname werden dann in einem Tooltip angezeigt.

Voraussetzung

Das Skript läuft in Runtime und der Debugger ist geöffnet.

Vorgehensweise

1. Setzen Sie mindestens einen Haltepunkt im aktuellen Skript.
2. Wechseln Sie zu WinCC Runtime und führen Sie eine Aktion aus, die das Skript zur Ausführung bringt.
   Der Debugger stoppt am ersten Haltepunkt.
3. Aktivieren Sie im Menü "View" den Eintrag "Command Window".
   Das "Command Window" erscheint.

4. Um den Wert einer Variablen oder Eigenschaft zu ermitteln, geben Sie ein "?" ein, gefolgt von einem Leerzeichen sowie den Namen der Variablen oder Eigenschaft, den Sie ermitteln möchten, z.B. "? myTag".
   Drücken Sie <Return>, um den Befehl auszuführen.
5. Um den Wert einer Variablen/Eigenschaft zu ändern, weisen Sie Ihr in der VBS-Syntax einen Wert zu.

Siehe auch

Grundlagen des Debuggens (Seite 95)

So führen Sie Skriptbefehle aus (Seite 110)

So setzen Sie Lesezeichen im Skript (Seite 108)

So löschen Sie Haltepunkte (Seite 107)

So setzen Sie Haltepunkte (Seite 105)

So wählen Sie ein Skript zur Bearbeitung aus (Seite 103)

Aktions- und Prozedurnamen im Debugger (Seite 101)

Aufbau von VBScript-Dateien (Seite 99)

Komponenten des Microsoft Script Debuggers (Seite 97)

So aktivieren Sie den Debugger (Seite 93)

Testen mit dem Debugger (Seite 92)

Diagnose (Seite 85)

1.12.4.13 So führen Sie Skriptbefehle aus

Einführung

Während ein Skript in Runtime läuft, können Sie das "Command Window" des Debuggers verwenden, um Skriptbefehle direkt auszuführen und so den Ablauf des laufenden Skripts zu manipulieren. Sie können die Skriptbefehle zum Testen direkt ausführen, ohne den Befehl in einem Skript zu erstellen und dieses ablaufen zu lassen. Sie können z.B.:

- Methoden aufrufen
- Prozeduren aufrufen
- Objekteigenschaften manipulieren

Sie können im "Command Window" prinzipiell alle Befehle ausführen, die Sie auch in einem VB-Skript ausführen können.

Voraussetzung

Das Skript läuft in Runtime und der Debugger ist geöffnet.

Vorgehensweise

1. Setzen Sie mindestens einen Haltepunkt im aktuellen Skript.
2. Wechseln Sie zu WinCC Runtime und führen Sie eine Aktion aus, die das Skript zur Ausführung bringt.
   Der Debugger stoppt am ersten Haltepunkt.
3. Aktivieren Sie im Menü "View" den Eintrag "Command Window".
   Das "Command Window" erscheint.
4. Geben Sie den gewünschten Befehl ein und drücken Sie "ENTER".

Hinweis

Wenn Sie einen fehlerhaften Befehl im Command Window eingeben, wird in Runtime keine Fehlermeldung erzeugt. Stattdessen erscheint im Command Fenster die Meldung "<Script Error>".

Siehe auch

So ermitteln und ändern Sie Variablen- oder Eigenschaftenwerte (Seite 109)

So setzen Sie Lesezeichen im Skript (Seite 108)

So löschen Sie Haltepunkte (Seite 107)

So setzen Sie Haltepunkte (Seite 105)

So wählen Sie ein Skript zur Bearbeitung aus (Seite 103)

Aktions- und Prozedurnamen im Debugger (Seite 101)

Aufbau von VBScript-Dateien (Seite 99)

Komponenten des Microsoft Script Debuggers (Seite 97)

Grundlagen des Debuggens (Seite 95)

So aktivieren Sie den Debugger (Seite 93)

Testen mit dem Debugger (Seite 92)

Diagnose (Seite 85)

1.13 So drucken Sie VB-Skripte

Prinzip

Sie können sowohl die in Global Script als auch die im Graphics Designer projektierten Aktionen und Prozeduren in WinCC dokumentieren.

Die Möglichkeiten der Dokumentation unterscheiden in:

- Rückdokumentation drucken: Im Graphics Designer werden mit der Rückdokumentation des aktuellen Bildes alle projektieren Aktionen gedruckt. Die Rückdokumentation enthält nebeneinander C-Aktionen und VBS-Aktionen, die durch Angabe des Quelltextes (C oder VBScript) unterschieden werden.
- Aktuelles Skript drucken: Die Rückdokumentation im Global Script enthält immer die aktuell geöffnete Prozedur oder Aktion.

Für das Layout der Rückdokumentation werden in WinCC vordefinierte Drucklayouts verwendet. Sie können auch eigene Drucklayouts entwerfen und mit dem Befehl "Projektdokumentation einrichten" auf der Registerkarte Druckauftrag verknüpfen.

Vorgehensweise

1. Öffnen Sie Global Script oder den Graphics Designer, je nachdem, welche Skripte Sie dokumentieren möchten.
2. Konfigurieren Sie ggf. den Druckauftrag mit dem Befehl "Projektdokumentation einrichten".
3. Mit dem Befehl "Projektdokumentation Ansicht" erhalten Sie eine Vorschau auf die Daten, die gedruckt werden.
4. Wählen Sie den Menübefehl "Datei" > "Projektdokumentation drucken", um die Daten auszudrucken.

Siehe auch

Aktionen erstellen und bearbeiten (Seite 56)

Prozeduren erstellen und bearbeiten (Seite 39)

Die VBScript-Editoren (Seite 28)

Einsatz von Visual Basic Script in WinCC (Seite 12)

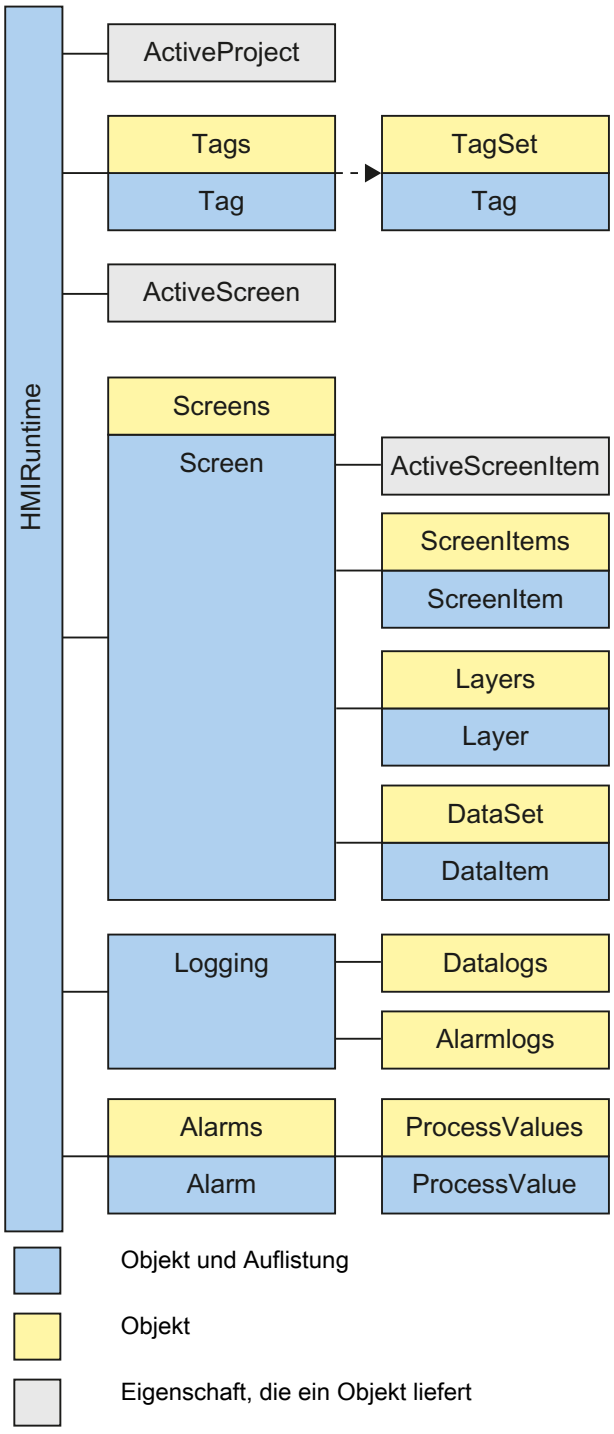
1.14 VBS Referenz

1.14.1 VBS Referenz

Das VBS-Objektmodell in WinCC

Das WinCC Objektmodell des grafischen Runtimesystems erlaubt Ihnen den Zugriff auf Grafikobjekte und Variablen in Runtime.

Wenn Sie mit der Maus auf einen Objektnamen klicken, erhalten Sie eine detaillierte Beschreibung.



Das VBS-Objektmodell in einem Faceplate-Typ

In einem Faceplate-Typ ist das VBS-Objektmodell für WinCC nicht gültig. Es wurde ersetzt durch ein komplett neues Modell.

Das VBS-Objektmodell des Faceplate-Typs erlaubt Ihnen den Zugriff auf die Grafikobjekte und Faceplate-Variablen des Faceplate-Typs in Runtime.

Objekte

Über die Objekte und Auflistungen erhalten Sie Zugriff auf alle Objekte des grafischen Runtime-Systems: Grafikobjekte, Bilder, Ebenen und Variablen.

Eigenschaften

Über die Eigenschaften der einzelnen Objekte können Sie Grafikobjekte und Variablen in Runtime gezielt verändern, z.B. auf Mausklick ein Bedienelement freischalten oder bei Änderung eines Variablenwerts einen Farbumschlag auslösen.

Methoden

Mit den Methoden, die Sie auf die einzelnen Objekte anwenden, können Sie z.B. Variablenwerte zur weiteren Verarbeitung auslesen oder Diagnosemeldungen in Runtime ausgeben.

Siehe auch

ActiveScreen-Eigenschaft (Seite 298)
Objekt-Typen des Objekts ScreenItem (Seite 152)
Methoden (Seite 685)
Eigenschaften (Seite 295)
Objekte und Auflistungen (Seite 117)
AlarmLogs-Objekt (Seite 121)
DataItem-Objekt (Seite 122)
DataLogs-Objekt (Seite 124)
DataSet-Objekt (Auflistung) (Seite 125)
HMIRuntime-Objekt (Seite 127)
Layer-Objekt (Seite 129)
Layers-Objekt (Auflistung) (Seite 130)
ScreenItem-Objekt (Seite 134)
ScreenItems-Objekt (Auflistung) (Seite 138)
Screen-Objekt (Seite 140)
Screens-Objekt (Auflistung) (Seite 143)
Tag-Objekt (Seite 146)
Tags-Objekt (Auflistung) (Seite 149)
TagSet-Objekt (Auflistung) (Seite 151)
ActiveProject-Eigenschaft (Seite 297)
ActiveScreenItem-Eigenschaft (Seite 298)
Logging-Objekt (Seite 131)
Alarm-Objekt (Seite 119)
Alarms-Objekt (Auflistung) (Seite 120)
ProcessValue-Objekt (Seite 132)
ProcessValues-Objekt (Auflistung) (Seite 133)

1.14.2 Objekte und Auflistungen

1.14.2.1 Objekte und Auflistungen

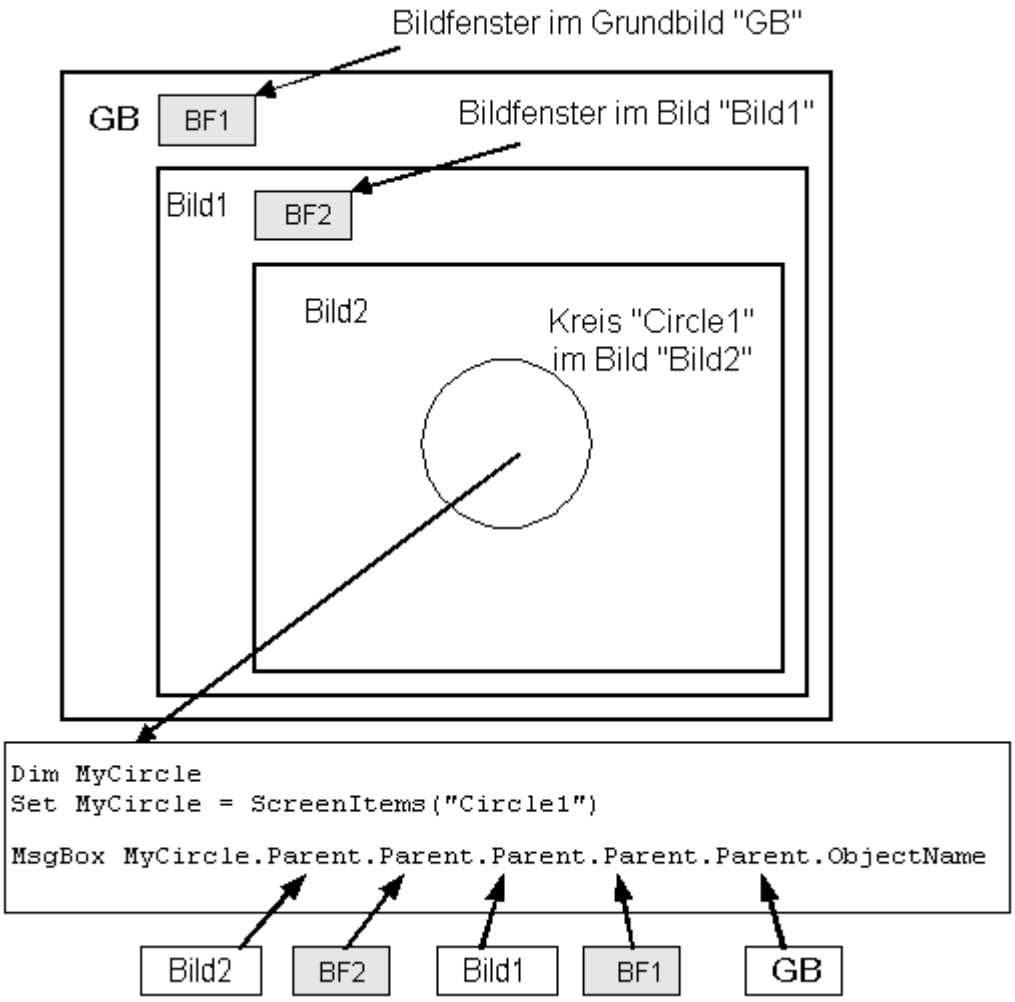
Übersicht

Die Objekte und Auflistungen des WinCC-Objektmodells erlauben Ihnen in Runtime Zugriff auf Grafikobjekte und Variablen.

Navigation im Objektmodell

Sie greifen auf Objekte innerhalb des VBS-Objektmodells hierarchisch zu. Wenn Sie z.B. ein Bildelement innerhalb eines Bildes ansprechen, sprechen Sie ein Bildelement, das in einem Bild liegt, über sein Parent-Objekt (das umgebende Bild) an.

Beispiel



Ausgegeben wird in diesem Beispiel der Grundbildname.

Zugriff auf Grafikobjekte

Sie greifen in WinCC über das übergeordnete Objekt "HMIRuntime" auf die Bilder, Ebenen und Grafikobjekte in Runtime zu. Der Zugriff auf Objekte und Ebenen erfolgt immer über das Bild (Screen), in dem sie enthalten sind.

Zugriff auf Variablen

Sie greifen in WinCC über das übergeordnete Objekt "HMIRuntime" direkt auf Variablen in Runtime zu. Werte von Variablen können Sie auslesen oder neu setzen.

Auflistungen

Die Auflistungen des WinCC-Objektmodells verhalten sich wie die Standard-Collections von VBS. Ausnahme: Die Auflistung "Tags" hat keine Enum-Funktionalität.

Verfügbare Objekte

- Alarm
- Alarms
- AlarmLogs
- DataItem
- DataLogs
- DataSet
- HMIRuntime
- Item
- Layer
- Layers
- Logging
- ProcessValues
- ProcessValue
- Project
- ScreenItem
- ScreenItems
- Screen
- Screens
- Tag

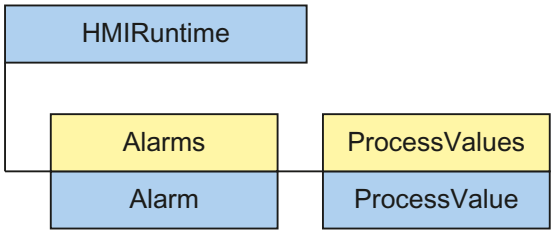
- Tags
- TagSet

Siehe auch

ScreenItems-Objekt (Auflistung) (Seite 138)
TagSet-Objekt (Auflistung) (Seite 151)
Tags-Objekt (Auflistung) (Seite 149)
Tag-Objekt (Seite 146)
Screens-Objekt (Auflistung) (Seite 143)
Screen-Objekt (Seite 140)
ScreenItem-Objekt (Seite 134)
Layers-Objekt (Auflistung) (Seite 130)
Layer-Objekt (Seite 129)
Item-Objekt (Seite 128)
HMIRuntime-Objekt (Seite 127)

1.14.2.2 Alarm-Objekt

Beschreibung



Das Alarm-Objekt wird verwendet um auf die Alarms-Objekt-Auflistung zuzugreifen.

Hinweis

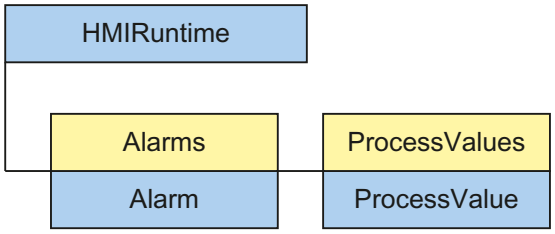
Die Eigenschaften des Alarm-Objekts werden nicht automatisch aktualisiert, wenn sich die Werte der Eigenschaften ändern.

Siehe auch

Alarms-Objekt (Auflistung) (Seite 120)

1.14.2.3 Alarms-Objekt (Auflistung)

Beschreibung



Mit dem Alarm-Objekt können Sie vorhandene Meldungen auslösen.

Verwendung

Sie können über die Auflistung "Alarms":

- Auf eine Meldung in der Auflistung zugreifen (Item-Methode)
- Ein neues Alarm-Objekt erzeugen (Create-Methode)
- Die AlarmID der Meldung auslesen (AlarmID-Eigenschaft)
- Den Status einer Meldung auslesen (State-Eigenschaft)
- Den Zeitstempel der Meldung auslesen (Timestamp-Eigenschaft)
- Eine Instanz des Alarm-Objekts erzeugen (Instance-Eigenschaft)
- Den Namen des Computers, auf dem die Meldung gekommen ist, auslesen (ComputerName-Eigenschaft)
- Den Namen des Benutzers, der die Meldung ausgelöst hat, auslesen oder setzen (UserName-Eigenschaft)
- Die Namen der Prozesswertblöcke lesen oder setzen (ProcessValues-Eigenschaft)
- Den Kommentar der Meldung lesen oder setzen (Comment-Eigenschaft)
- Das Serverpräfix der Meldung lesen oder setzen (Context-Eigenschaft)

Beispiel

Im folgenden Beispiel wird die im Alarm Logging-Editor projektierte Meldung mit der Meldenummer "1" ausgelöst:

```
'VBS360
Dim MyAlarm
Set MyAlarm = HMIRuntime.Alarms(1)
MyAlarm.State = 5 'hmiAlarmStateCome + hmiAlarmStateComment
MyAlarm.Comment = "MyComment"
MyAlarm.UserName = "Hans-Peter"
MyAlarm.ProcessValues(1) = "Process Value 1"
MyAlarm.ProcessValues(4) = "Process Value 4"
```

```
MyAlarm.Create "MyApplication"
```

Siehe auch

TimeStamp-Eigenschaft (Seite 599)

ComputerName-Eigenschaft (Seite 370)

Context-Eigenschaft (Seite 371)

State-Eigenschaft (Seite 564)

AlarmID-Eigenschaft (Seite 302)

Instance-Eigenschaft (Seite 431)

Comment-Eigenschaft (Seite 369)

UserName-Eigenschaft (Seite 652)

ProcessValue-Eigenschaft (Seite 522)

Alarm-Objekt (Seite 119)

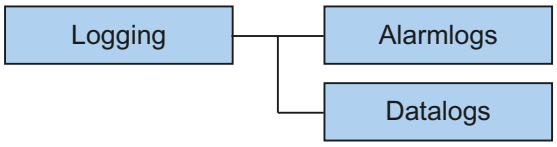
ProcessValues-Objekt (Auflistung) (Seite 133)

Create-Methode (Seite 691)

Item-Methode (Seite 744)

1.14.2.4 AlarmLogs-Objekt

Beschreibung



Mit dem Objekt können ausgelagerte Archivsegmente des Alarm Logging wieder mit der Runtime verbunden oder zuvor eingelagerte Archivsegmente des Alarm Logging wieder gelöscht werden. Dabei werden

- die einzulagernden Archivsegmente in das Common Archiving-Verzeichnis des WinCC-Projektes kopiert oder
- die zuvor eingelagerten Archivsegmente im Common Archiving-Verzeichnis gelöscht.

Über Parameter können Sie steuern, von welchem Ort Archivsegmente eingelagert werden sollen. Es kann auch der Zeitraum festgelegt werden, über den Archivsegmente eingelagert oder gelöscht werden sollen. Die Archivsegmente werden dabei in das Common Archiving-Verzeichnis des Projektes kopiert.

Wenn bei der Operation mit den Archivsegmenten ein Fehler aufgetreten ist, gibt die eingesetzte Methode eine Fehlermeldung zurück. Weitere Informationen finden Sie unter dem Thema "Fehlermeldungen aus dem Bereich Datenbanken".

Verwendung

Zuvor ausgelagerte Archivsegmente des Alarm Logging können mit der Runtime verbunden werden (Methode "Restore").

Zuvor eingelagerte Archivsegmente des Alarm Logging können aus dem Runtime-Projekt gelöscht werden (Methode "Remove")

Beispiel

In dem folgenden Beispiel werden Alarm Logging Archivsegmente eingelagert und der Rückgabewert als Trace ausgegeben.

```
'VBS187
HMIRuntime.Trace "Ret: " & HMIRuntime.Logging.AlarmLogs.Restore("D:
\Folder","2004-09-14","2004-09-20",-1) & vbNewLine
```

Siehe auch

Fehlermeldungen aus dem Bereich Datenbanken (Seite 794)

Restore-Methode (Seite 768)

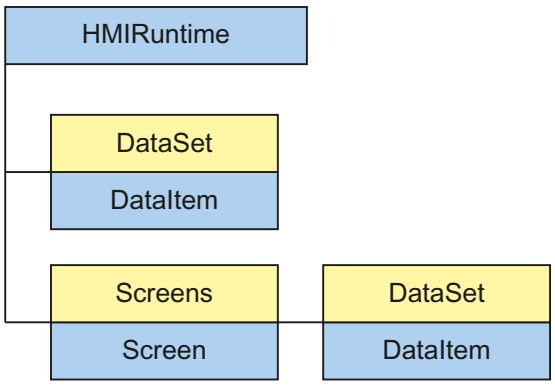
Remove-Methode (Seite 762)

DataLogs-Objekt (Seite 124)

Logging-Objekt (Seite 131)

1.14.2.5 DataItem-Objekt

Beschreibung



Das DataItem-Objekt wird benutzt, um auf die Inhalte der DataSet-Auflistung zuzugreifen. Werte oder Objektreferenzen werden in der Auflistung als DataItem abgelegt.

Der Zugriff erfolgt über den Namen, unter dem der Wert in die Auflistung hinzugefügt wurde. Ein Einzelzugriff über Index ist nicht empfehlenswert, da sich der Index beim Hinzufügen und

Löschen von Werten ändert. Der Index kann verwendet werden, um den kompletten Inhalt der Auflistung auszugeben. Die Ausgabe erfolgt in alphabetischer Reihenfolge.

Hinweis

Bei Objektreferenzen muss sichergestellt sein, dass die Objekte multithreadfähig sind.

Beispiel

Das Beispiel zeigt wie der Wert von 'Motor1' als Trace ausgegeben wird.

```
'VBS163
HMIRuntime.Trace "motor1: " & HMIRuntime.DataSet("motor1").Value & vbNewLine
```

Das folgende Beispiel enumeriert alle DataItem-Objekte der DataSet-Auflistung. Name und Value werden als Trace ausgegeben.

```
'VBS164
Dim data
For Each data In HMIRuntime.DataSet
HMIRuntime.Trace data.Name & ": " & data.Value & vbNewLine
Next
```

Hinweis

Bei Objekten kann Value evtl. nicht direkt ausgegeben werden

Siehe auch

Screen-Objekt (Seite 140)

HMIRuntime-Objekt (Seite 127)

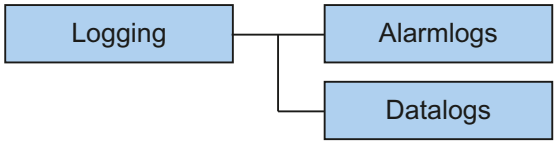
DataSet-Objekt (Auflistung) (Seite 125)

Value-Eigenschaft (Seite 655)

Name-Eigenschaft (Seite 487)

1.14.2.6 DataLogs-Objekt

Beschreibung



Mit dem Objekt können ausgelagerte Archivsegmente des Tag Logging wieder mit der Runtime verbunden oder zuvor eingelagerte Archivsegmente des Tag Logging wieder gelöscht werden. Dabei werden

- die einzulagernden Archivsegmente in das Common Archiving-Verzeichnis des WinCC-Projektes kopiert oder
- die zuvor eingelagerten Archivsegmente im Common Archiving-Verzeichnis gelöscht.

Über Parameter können Sie steuern, von welchem Ort Archivsegmente eingelagert werden sollen. Es kann auch der Zeitraum festgelegt werden, über den Archivsegmente eingelagert oder gelöscht werden sollen. Zusätzlich können Sie den Typ des Archivs einstellen ("Tag Logging Fast", "Tag Logging Slow", "Tag Logging Fast und Tag Logging Slow"). Die Archivsegmente werden dabei in das Common Archiving-Verzeichnis des Projektes kopiert.

Wenn bei der Operation mit den Archivsegmenten ein Fehler aufgetreten ist, gibt die eingesetzte Methode eine Fehlermeldung zurück. Weitere Informationen finden Sie unter dem Thema "Fehlermeldungen aus dem Bereich Datenbanken".

Verwendung

Zuvor ausgelagerte Archivsegmente des Tag Logging können mit der Runtime verbunden werden (Methode "Restore").

Zuvor eingelagerte Archivsegmente des Tag Logging können aus dem Runtime-Projekt gelöscht werden (Methode "Remove")

Beispiel

In dem folgenden Beispiel werden Tag Logging Fast-Archivsegmente eingelagert und der Rückgabewert als Trace ausgegeben.

```
'VBS188
HMIRuntime.Trace "Ret: " & HMIRuntime.Logging.DataLogs.Restore("D:
\Folder","2004-09-14","2004-09-20",-1,1) & vbNewLine
```

Siehe auch

Fehlermeldungen aus dem Bereich Datenbanken (Seite 794)

Restore-Methode (Seite 768)

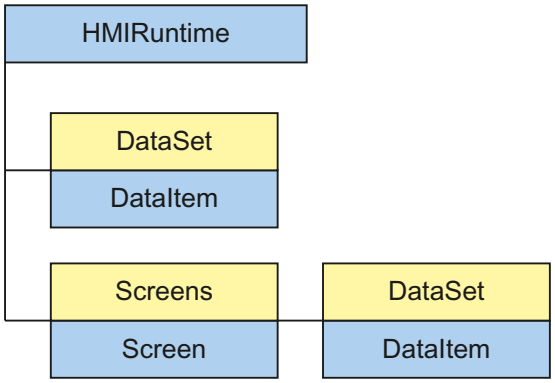
Remove-Methode (Seite 762)

AlarmLogs-Objekt (Seite 121)

Logging-Objekt (Seite 131)

1.14.2.7 DataSet-Objekt (Auflistung)

Beschreibung



Mit dem DataSet-Objekt können über mehrere Aktionen hinweg Daten ausgetauscht werden.

Ein DataSet-Objekt ist global und am Screen-Objekt definiert. Auf die Daten kann aus jeder VBS-Aktion zugegriffen werden.

Das DataSet-Objekt am Screen-Objekt muss entsprechend der Bildhierarchie adressiert werden und besteht solange das Bild angezeigt wird. Das globale Objekt besteht über den gesamten Zeitraum des Runtime.

Der Zugriff erfolgt über das DataItem-Objekt.

Hinweis

Objekte vom Typ Screen, Screens, ScreenItem, ScreenItems, Tag und TagSet können nicht in die DataSet-Auflistung aufgenommen werden.
Das DataSet-Objekt unterstützt keine Klassen.

Verwendung

Sie können über die Auflistung "DataSet":

- Alle Objekte innerhalb der Auflistung ausgeben oder bearbeiten (Enumerieren).
- Die Anzahl der enthaltenen Elemente ausgeben (Eigenschaft "Count").

- Ein bestimmtes Objekt der Auflistung bearbeiten (Methode "Item").
- Ein Objekt zur Auflistung hinzufügen (Methode "Add").
- Ein bestimmtes Objekt aus der Auflistung entfernen (Methode "Remove").
- Alle Objekte aus der Auflistung entfernen (Methode "RemoveAll").

Der Zugriff auf die Elemente der Auflistung erfolgt über :

```
HMIRuntime.DataSet("Itemname")
```

Bei einer bildspezifischen Auflistung erfolgt der Zugriff über :

```
HMIRuntime.Screens("Screenname").DataSet("Itemname")
```

In einem Bild kann man auf das DataSet-Objekt des Bildes zugreifen über :

```
DataSet("Itemname")
```

Wenn beim Zugriff der angegebene Name nicht in der Auflistung existiert, wird VT_Empty zurückgeliefert und eine Exception ausgelöst.

Beispiel

Das Beispiel zeigt, wie man einen Wert in die Auflistung aufnimmt, ihn ausliest und wieder entfernt. Sinnvollerweise geschieht dies in verschiedenen Aktionen.

```
'VBS162
HMIRuntime.DataSet.Add "motor1", 23
HMIRuntime.Trace "motor1: " & HMIRuntime.DataSet("motor1").Value & vbNewLine
HMIRuntime.DataSet.Remove("motor1")
```

Siehe auch

DataItem-Objekt (Seite 122)

RemoveAll-Methode (Seite 766)

Remove-Methode (Seite 762)

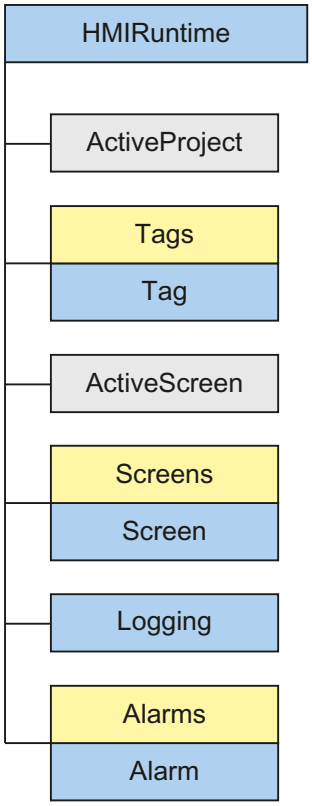
Item-Methode (Seite 744)

Count-Eigenschaft (Seite 372)

Add-Methode (Seite 688)

1.14.2.8 HMIRuntime-Objekt

Beschreibung



Das HMIRuntime-Objekt repräsentiert die grafische Runtime-Umgebung.

Verwendung

Sie können über das Objekt "HMIRuntime" z.B.:

- Die aktuelle Runtime-Sprache lesen oder setzen (Eigenschaft "Language").
- Den Namen des aktuellen Grundbildes lesen oder setzen (Eigenschaft "BaseScreenName").
- Den Pfad des aktiven Runtime-Projektes lesen (Eigenschaft "ActiveProject").
- Auf Variablen zugreifen (Eigenschaft "Tags").
- Auf Variablen einer Auflistung zugreifen (Eigenschaft "DataSet").
- Runtime beenden (Methode "Stop").
- Meldungen in einem Diagnosefenster ausgeben (Methode "Trace").

Beispiel

Folgender Befehl beendet WinCC Runtime:

```
'VBS3
HMIRuntime.Stop
```

Siehe auch

Screens-Objekt (Auflistung) (Seite 143)
TagSet-Objekt (Auflistung) (Seite 151)
Tags-Objekt (Auflistung) (Seite 149)
Logging-Objekt (Seite 131)
DataSet-Objekt (Auflistung) (Seite 125)
Visible-Eigenschaft (Seite 671)
Trace-Methode (Seite 785)
Tags-Eigenschaft (Seite 573)
Stop-Methode (Seite 784)
AlignmentLeft-Eigenschaft (Seite 303)
Logging-Eigenschaft (Seite 462)
Language-Eigenschaft (Seite 435)
DataSet-Eigenschaft (Seite 377)
CurrentContext-Eigenschaft (Seite 373)
BaseScreenName-Eigenschaft (Seite 323)
ActiveProject-Eigenschaft (Seite 297)
ActiveScreen-Eigenschaft (Seite 298)
MenuToolBarConfig-Eigenschaft (Seite 473)
Alarms-Objekt (Auflistung) (Seite 120)

1.14.2.9 Item-Objekt

Beschreibung

Das Objekt "Item" liefert Ihnen eine Referenz auf das aktuelle Objekt.

Verwendung

Sie verwenden das Objekt "Item", um z.B. die Eigenschaften des aktuell im Graphics Designer markierten Objektes anzusprechen.

Beispiel

Im folgenden Beispiel haben Sie z.B. ein Rechteck erstellt. Wenn das Objekt markiert ist, können Sie alle Eigenschaften des aktuellen Objektes ansprechen, hier die Hintergrundfarbe auf Rot setzen:

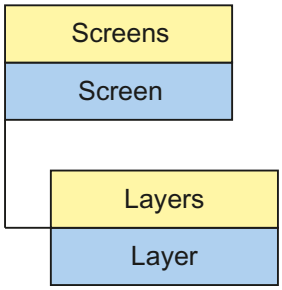
```
'VBS195
Item.BackColor = RGB(255,0,0)
```

Siehe auch

Objekte und Auflistungen (Seite 117)

1.14.2.10 Layer-Objekt

Beschreibung



Das Layer-Objekt wird Ihnen als Ergebnis des Zugriffs auf die Layers-Auflistung zurückgegeben.

Parent-Objekt

Bild, in dem die Bildebene liegt.

Verwendung

Über das Layer-Objekt können Sie abhängig von bestimmten Ereignissen auf die Eigenschaften einer kompletten Ebene zugreifen, z.B. um abhängig von der Bedienberechtigung eine Ebene mit Bedienelementen aus- oder einzublenden.

Sie können über das Objekt "Layer":

- Die Sichtbarkeit einer Ebene ein- oder ausschalten (Eigenschaft "Visible").
- Den Namen einer Ebene auslesen (Eigenschaft "Name").

Hinweis

Die Layer-Eigenschaft, gibt die Ebene aus, in der sich das Objekt befindet. Die Ebene "0" wird als Ebene"0" ausgegeben.

Beim Zugriff werden in VBS die Ebenen von 1 an gezählt. Deshalb muss die Ebene "1" mit layers(2) angesprochen werden.

Beispiel

Im folgenden Beispiel wird die Ebene 1 auf unsichtbar gesetzt:

```
'VBS4
Layers(2).Visible = vbFalse
```

Siehe auch

Layer-Objekt (Seite 129)

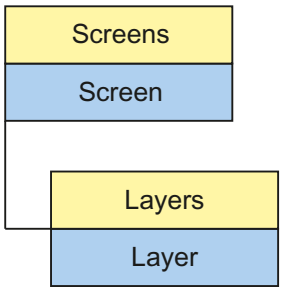
Visible-Eigenschaft (Seite 671)

Parent-Eigenschaft (Seite 506)

Name-Eigenschaft (Seite 487)

1.14.2.11 Layers-Objekt (Auflistung)

Beschreibung



Mit der Layers-Auflistung können Sie auf alle 32 Ebenen des grafischen Runtime-Systems zugreifen.

Parent-Objekt

Bild, in dem die Bildebene liegt.

Verwendung

Sie können über die Auflistung "Layers":

- Alle Ebenen innerhalb der Auflistung bearbeiten (Eigenschaft "_NewEnum").
- Alle in der Auflistung enthaltenen Ebenen zählen (Eigenschaft "Count").
- Eine Ebene aus der Auflistung bearbeiten (Methode "Item").

Die Eigenschaften sind Standard-Eigenschaften und Methoden einer Auflistung und werden in der WinCC-Dokumentation nicht detailliert beschrieben.

Siehe auch

Parent-Eigenschaft (Seite 506)

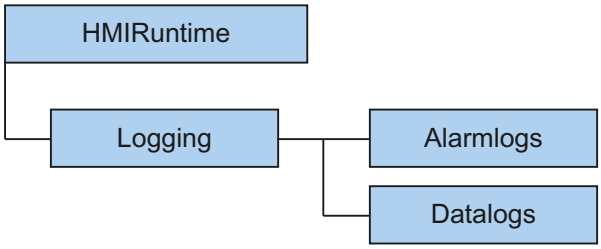
Item-Methode (Seite 744)

Count-Eigenschaft (Seite 372)

Layer-Objekt (Seite 129)

1.14.2.12 Logging-Objekt

Beschreibung



Mit dem Objekt können ausgelagerte Archivsegmente wieder mit der Runtime verbunden oder zuvor eingelagerte Archivsegmente wieder gelöscht werden. Dabei werden

- die einzulagernden Archivsegmente in das Common Archiving-Verzeichnis des WinCC-Projektes kopiert oder
- die zuvor eingelagerten Archivsegmente im Common Archiving-Verzeichnis gelöscht.

Über Parameter können Sie steuern, von welchem Ort Archivsegmente eingelagert werden sollen. Es kann auch der Zeitraum festgelegt werden, über den Archivsegmente eingelagert oder gelöscht werden sollen. Die Archivsegmente werden dabei in das Common Archiving-Verzeichnis des Projektes kopiert.

Wenn bei der Operation mit den Archivsegmenten ein Fehler aufgetreten ist, gibt die eingesetzte Methode eine Fehlermeldung zurück. Weitere Informationen finden Sie unter dem Thema "Fehlermeldungen aus dem Bereich Datenbanken".

Verwendung

Zuvor ausgelagerte Archivsegmente des Alarm Logging und des Tag Logging können mit der Runtime verbunden werden (Methode "Restore").

Zuvor eingelagerte Archivsegmente des Alarm Logging und des Tag Logging können aus dem Runtime-Projekt gelöscht werden (Methode "Remove").

Beispiel

In dem folgenden Beispiel werden Alarm und Tag Logging Archivsegmente eingelagert und der Rückgabewert als Trace ausgegeben.

```
'VBS189
HMIRuntime.Trace "Ret: " & HMIRuntime.Logging.Restore("D:
\Folder","2004-09-14","2004-09-20",-1) & vbNewLine
```

Siehe auch

Fehlermeldungen aus dem Bereich Datenbanken (Seite 794)

DataLogs-Objekt (Seite 124)

AlarmLogs-Objekt (Seite 121)

Restore-Methode (Seite 768)

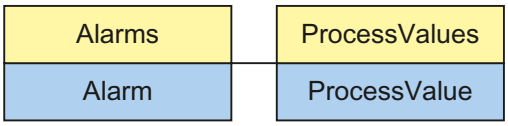
Remove-Methode (Seite 762)

DataLogs-Eigenschaft (Seite 377)

AlarmLogs-Eigenschaft (Seite 302)

1.14.2.13 ProcessValue-Objekt

Beschreibung



Das ProcessValue-Objekt wird verwendet um auf die ProcessValues-Objekt-Auflistung zuzugreifen.

Hinweis

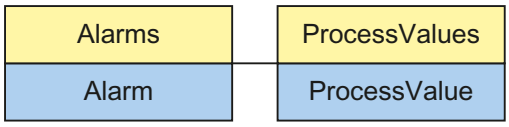
Es werden nur die zehn vordefinierten ProcessValues unterstützt.

Siehe auch

ProcessValues-Objekt (Auflistung) (Seite 133)

1.14.2.14 ProcessValues-Objekt (Auflistung)

Beschreibung



Verwendung

Sie können über die Auflistung "ProcessValues":

- Einen ProcessValue aus der Auflistung bearbeiten (Item-Methode)
- Alle Objekte innerhalb der Auflistung ausgeben oder bearbeiten (_NewEnum-Eigenschaft)
- Alle in der Auflistung enthaltenen ProcessValues zählen (Count-Eigenschaft)
- Die Werte des ProcessValue-Objekts lesen oder setzen (Value-Eigenschaft)

Die Eigenschaften sind Standard-Eigenschaften und Methoden einer Auflistung und werden in der WinCC-Dokumentation nicht detailliert beschrieben.

Siehe auch

Alarms-Objekt (Auflistung) (Seite 120)

ProcessValue-Objekt (Seite 132)

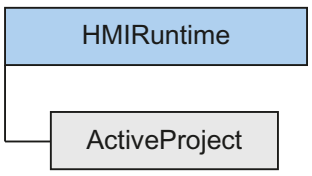
Count-Eigenschaft (Seite 372)

Value-Eigenschaft (Seite 655)

Item-Methode (Seite 744)

1.14.2.15 Project-Objekt

Beschreibung



Mit dem Objekt können Informationen des aktuellen Runtime-Projektes abgefragt werden.

Das Project-Objekt wird als Ergebnis von ActiveProject zurückgegeben.

Verwendung

Sie können über das Objekt "Project" :

- Den Pfad des aktuellen Runtime-Projektes lesen (Eigenschaft "Path").
- Den Namen des aktuellen Runtime-Projektes lesen, ohne Pfadangabe und Datei-Erweiterung (Eigenschaft "Name").

Beispiel

Das folgende Beispiel gibt den Namen und den Pfad des aktuellen Runtime-Projektes als Trace aus:

```
'VBS159
HMIRuntime.Trace "Name: " & HMIRuntime.ActiveProject.Name & vbNewLine
HMIRuntime.Trace "Path: " & HMIRuntime.ActiveProject.Path & vbNewLine
```

Siehe auch

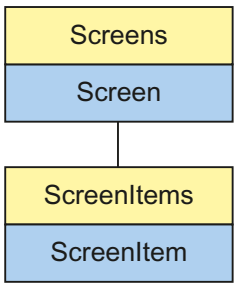
ActiveProject-Eigenschaft (Seite 297)

Name-Eigenschaft (Seite 487)

Path-Eigenschaft (Seite 508)

1.14.2.16 ScreenItem-Objekt

Beschreibung



Das ScreenItem-Objekt wird Ihnen als Ergebnis des Zugriffs auf die ScreenItems-Auflistung zurückgegeben.

Parent-Objekt

Bild, in dem das Bildelement liegt.

Verwendung

Über das ScreenItem-Objekt können Sie abhängig von bestimmten Ereignissen auf die Eigenschaften von Grafikobjekten innerhalb eines Bildes zugreifen.

Sie können über das Objekt "ScreenItem" z.B.:

- Die Sichtbarkeit eines Objektes ein- oder ausschalten (Eigenschaft "Visible").
- Ein Objekt zur Bedienung freigeben oder sperren (Eigenschaft "Enabled").
- Die Breite und Höhe eines Objektes verändern (Eigenschaften "Height" und "Width").
- Die Position eines Objektes verändern (Eigenschaften "Top" und "Left").
- Die Ebene, in der ein Grafikobjekt liegt, auslesen und setzen (Eigenschaft "Layer").
- Den Namen eines Grafikobjektes auslesen oder setzen (Eigenschaft "ObjectName").
- Eine Referenz auf das übergeordnete Bild setzen (Eigenschaft "Parent").

Mit der Methode "Activate" wird der Fokus auf das entsprechende ScreenItem-Objekt gesetzt. Wenn der Fokus nicht gesetzt werden kann, weil z.B. das Objekt nicht bedienbar ist, wird ein Fehler erzeugt. Über eine Fehlerbehandlung (On Error Resume Next) kann der Fehler ausgewertet werden.

Mögliche Ausprägungen von ScreenItem

Das Objekt "ScreenItem" kann folgende Objekt-Typen beinhalten:

Standardobjekte	Smartobjekte	Windowsobjekte	Rohrobjekte	Controls	Weitere
Ellipse	3D-Balken	Button	Doppel-T-Stück	Siemens HMI Symbol Library	Anwender-Objekt
Ellipsenbogen	Applikationsfenster	Check-Box	Polygonrohr	WinCC AlarmControl	Gruppe
Ellipsensegment	Balken	Radio-Box	Rohrbogen	WinCC Digital/ Analog Clock Control	
Kreis	Bildfenster	Rundbutton	T-Stück	WinCC FunctionTrendControl	
Kreisbogen	Control	Slider		WinCC Gauge Control	
Kreissegment	EA-Feld			WinCC OnlineTrendControl	
Linie	Faceplate-Instanz			WinCC OnlineTableControl	
Polygon	Grafik-Objekt			WinCC Push Button Control	
Polygonzug	Kombinationsfeld			WinCC RulerControl	

Standardobjekte	Smartobjekte	Windowsobjekte	Rohrobjekte	Controls	Weitere
Rechteck	Listenfeld			WinCC Slider Control	
Rundrechteck	Mehrzeiliger Text			WinCC UserArchiveControl	
Verbinder	OLE-Objekt				
	Sammelanzeige				
	Textliste				
	Zustandsanzeige				

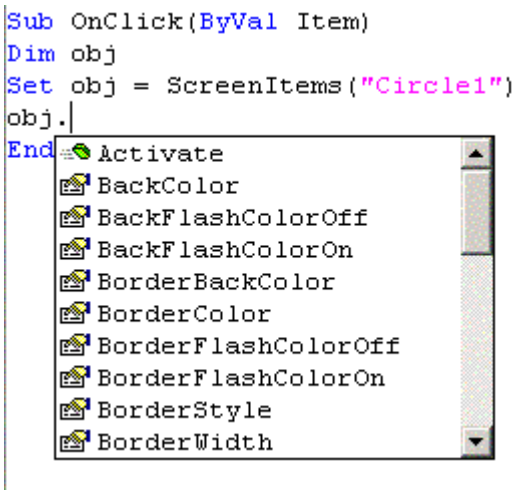
Unter "Objekt-Typen des Objektes ScreenItem" finden Sie detaillierte Beschreibungen zu den einzelnen Objekt-Typen. Über die Eigenschaft "Type" des ScreenItem-Objektes können Sie die Objekt-Typen über ihre VBS-Typkennzeichnung ansprechen.

Objekt-Eigenschaften

Das Objekt "ScreenItem" verfügt je nach Ausprägung über unterschiedliche Eigenschaften. Im Folgenden finden Sie die Eigenschaften, die jeder Objekt-Typ von ScreenItem besitzt:

```
Sub OnClick(ByVal Item)
Dim obj
Set obj = ScreenItems("Circle1").
End Sub
```

Wenn Sie einen bestimmten Objekt-Typ ansprechen, kommen weitere Eigenschaften zu den Standardeigenschaften hinzu:



Die zusätzlichen Eigenschaften finden Sie bei der Beschreibung der einzelnen Objekt-Typen.

Beispiel

Im folgenden Beispiel setzen Sie den Radius eines Kreises in Runtime bei Mausklick auf 2:

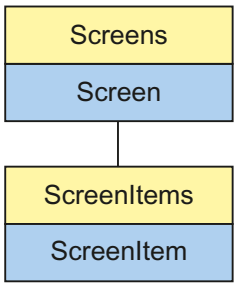
```
Sub OnClick (ByVal Item)
'VBS5
Dim objCircle
Set objCircle= ScreenItems("Circle1")
objCircle.Radius = 2
End Sub
```

Siehe auch

Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
Parent-Eigenschaft (Seite 506)
Left-Eigenschaft (Seite 454)
Layer-Eigenschaft (Seite 437)
Height-Eigenschaft (Seite 421)
Enabled-Eigenschaft (Seite 386)
Activate-Methode (Seite 686)
Beispiel: So lesen Sie Variablenwerte (Seite 803)
Beispiel: So schreiben Sie Variablenwerte (Seite 801)
Eigenschaften (Seite 295)
Objekte und Auflistungen (Seite 117)
Objekt-Typen des Objekts ScreenItem (Seite 152)

1.14.2.17 ScreenItems-Objekt (Auflistung)

Beschreibung



Mit der Auflistung "ScreenItems" können Sie auf ein Objekt im Bild referenzieren.

Parent-Objekt

Bild, in dem das Bildelement liegt.

Verwendung

Sie können über die Auflistung "ScreenItems":

- Alle Objekte innerhalb der Auflistung (also alle Objekte innerhalb eines Bildes) ausgeben oder bearbeiten (Eigenschaft "_NewEnum").
- Die Objekte eines Bildes aufzählen (Eigenschaft "Count").
- Ein bestimmtes Objekt der Auflistung bearbeiten (Methode "Item").

Die Eigenschaften sind Standard-Eigenschaften und Methoden einer Collection und werden in der WinCC-Dokumentation nicht detailliert beschrieben.

Besonderheiten des ScreenItem-Objektes

Wenn Sie ein externes Control (ActiveX-Control oder OLE-Objekt) in WinCC einbetten, kann es vorkommen, dass die Eigenschaften des eingebetteten Controls namensgleich sind mit den allgemeinen Eigenschaften des ScreenItem-Objektes. In diesem Fall haben die ScreenItem-Eigenschaften Vorrang.

Sie können die Eigenschaften des eingebetteten Controls aber über die Eigenschaft "object" ansprechen:

Die Eigenschaft "object" gibt es nur bei ActiveX-Controls und OLE-Objekten.

Beispiel:

```
'Control1 is an embedded ActiveX-Control with property "type"
'VBS196
Dim Control
Set Control=ScreenItems("Control1")
Control.object.type

'Control1 is a WinCC-Control
'VBS197
Dim Control
Set Control=ScreenItems("Control1")
Control.type
```

Beispiel

Im folgenden Beispiel geben Sie die Namen der im aktuellen Bild enthaltenen Objekte in einer Messagebox aus:

```
Sub OnClick(ByVal Item)
'VBS6
  Dim lngAnswer
  Dim lngIndex
  lngIndex = 1
  For lngIndex = 1 To ScreenItems.Count
    lngAnswer = MsgBox(ScreenItems(lngIndex).Objectname, vbOKCancel)
    If vbCancel = lngAnswer Then Exit For
```

```
Next
End Sub
```

Siehe auch

Count-Eigenschaft (Seite 372)

Beispiel: So lesen Sie Variablenwerte (Seite 803)

Beispiel: So schreiben Sie Variablenwerte (Seite 801)

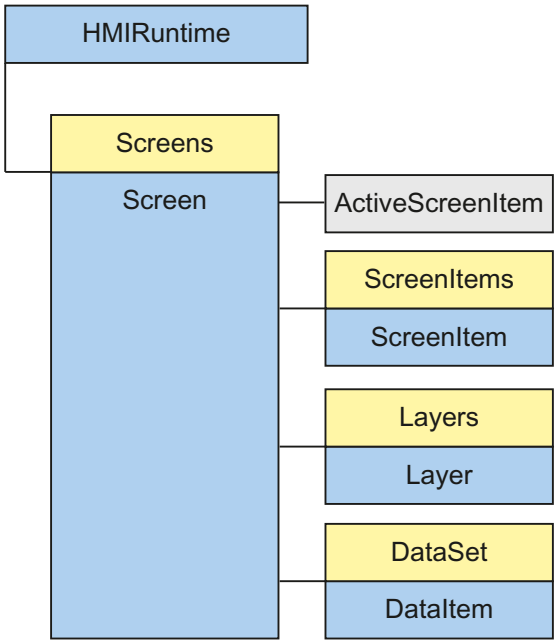
ScreenItem-Objekt (Seite 134)

Parent-Eigenschaft (Seite 506)

Item-Methode (Seite 744)

1.14.2.18 Screen-Objekt

Beschreibung



Das Screen-Objekt wird Ihnen als Ergebnis des Zugriffs auf die Screens-Auflistung zurückgegeben. Sie können alle Eigenschaften und Methoden dieses Objektes in Runtime auch direkt bearbeiten. Das Objekt "Screen" repräsentiert ein WinCC-Bild in Runtime und enthält alle Eigenschaften des Bild-Dokumentes und der Bild-Ansicht.

Des Weiteren enthält das Objekt "Screen":

- Eine Auflistung aller in dem angesprochenen Bild enthaltenen Grafikobjekte, die über das Objekt "ScreenItems" angesprochen werden können.
- Eine Auflistung aller in dem angesprochenen Bild enthaltenen Ebenen, die über das Objekt "Layers" angesprochen werden können.

Parent-Objekt

Bildfenster, in dem das Screen-Objekt eingebettet ist.

Wenn das Screen-Objekt das Grundbild ist, dann ist das Parent-Objekt nicht definiert und auf Null gesetzt.

Verwendung

Sie können über das Objekt "Screen" z.B.:

- Ein Bild zur Bedienung freigeben oder sperren (Eigenschaft "Enabled").
- Die Breite und Höhe eines Bildes bearbeiten (Eigenschaften "Width" und "Heigth").
- Den Zoom eines Bildes bearbeiten (Eigenschaft "Zoom").
- Füllmuster, Hintergrundfarbe und Füllmusterfarbe ändern (Eigenschaften "Fillstyle", "Backcolor" und "Fillcolor").

Hinweis

Wird ein Bildwechsel ausgeführt, werden alle geöffneten Referenzen auf nicht mehr vorhandene Bilder ungültig. Sie können mit diesen Referenzen dann nicht mehr arbeiten.

Beispiel

Im folgenden Beispiel wird die Breite des ersten Bildes in Runtime um 20 Pixel erhöht:

```
'VBS7
Dim objScreen
Set objScreen = HMIRuntime.Screens(1)
MsgBox "Screen width before changing: " & objScreen.Width
objScreen.Width = objScreen.Width + 20
MsgBox "Screen width after changing: " & objScreen.Width
```

Hinweise zu CrossReference

Alle Bilder, die Sie mit der Standardformulierung

```
HMIRuntime.BaseScreenName = "Screenname"
```

adressieren, werden durch die CrossReference von WinCC automatisch erfasst und sind in den Bildeigenschaften aufgeführt.

Sollten Sie Bilder mit anderen Formulierungen in Ihrem Code ansprechen, können Sie diese über folgende Sektion der CrossReference bekanntmachen:

```
' WINCC:SCREENNAME_SECTION_START
Const ScreenNameInAction = "ScreenName"
' WINCC:SCREENNAME_SECTION_END
Sie können diese Sektion beliebig oft in VBS-Aktionen einfügen.
```

Hinweis

Bildnamen sind aus Kompatibilitätsgründen zu zukünftigen Versionen immer ohne die Dateierweiterung ".PDL" zu schreiben.

Siehe auch

ScreenItems-Eigenschaft (Seite 538)

Refresh-Methode (Seite 761)

Activate-Methode (Seite 686)

Zoom-Eigenschaft (Seite 684)

Width-Eigenschaft (Seite 673)

Parent-Eigenschaft (Seite 506)

ObjectSizeDeclutteringMin-Eigenschaft (Seite 492)

ObjectSizeDeclutteringMax-Eigenschaft (Seite 491)

ObjectSizeDeclutteringEnable-Eigenschaft (Seite 491)

ObjectName-Eigenschaft (Seite 490)

Layers-Eigenschaft (Seite 454)

DataSet-Eigenschaft (Seite 377)

LayerDeclutteringEnable-Eigenschaft (Seite 453)

Height-Eigenschaft (Seite 421)

Fillstyle-Eigenschaft (Seite 398)

FillColor-Eigenschaft (Seite 396)

ExtendedZoomingEnable-Eigenschaft (Seite 394)

Enabled-Eigenschaft (Seite 386)

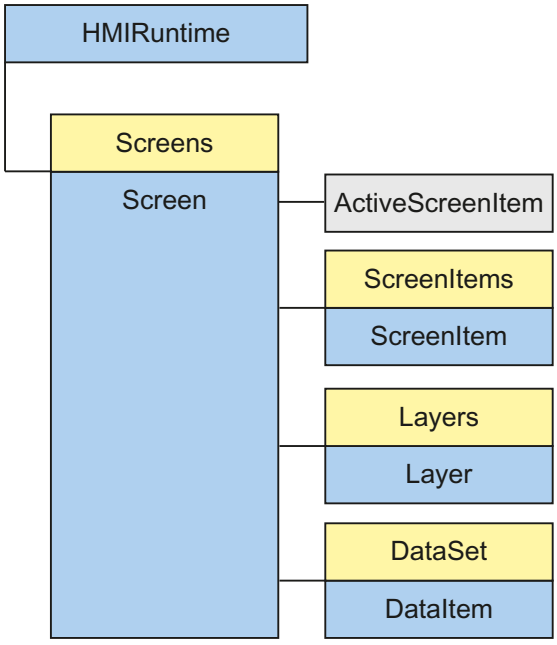
BackColor-Eigenschaft (Seite 316)

ActiveScreenItem-Eigenschaft (Seite 298)

AccessPath-Eigenschaft (Seite 296)

1.14.2.19 Screens-Objekt (Auflistung)

Beschreibung



Mit Hilfe der Bildfenster-Technik können in WinCC Runtime mehrere Bilder gleichzeitig geöffnet sein, wobei aber nur ein Grundbild exisitiert. Die Auflistung "Screens" ermöglicht den Zugriff auf alle geöffneten Bilder in Runtime über den Bildnamen. Die Screens-Auflistung enthält auch alle unsichtbaren Bilder.

Verwendung

Wenn Sie ein Mehrplatzprojekt projektieren, beachten Sie, dass Sie das Serverprefix angeben müssen, wenn Sie auf ein Bild zugreifen, dass nicht lokal auf dem Rechner liegt.

Sie können über die Auflistung "Screens":

- Alle Bilder innerhalb der Auflistung ausgeben oder bearbeiten (Eigenschaft "_NewEnum").
- Die Bilder eines Projektes aufzählen (Eigenschaft "Count").
- Ein bestimmtes Bild der Auflistung bearbeiten (Methode "Item").
- Alle sichtbaren Bilder neu zeichnen lassen (Methode "Refresh").

Die Eigenschaften sind Standard-Eigenschaften und Methoden einer Collection und werden in der WinCC-Dokumentation nicht detailliert beschrieben.

Der Zugriffsschlüssel, der im VBS-Umfeld in der Anweisung HMIRuntime.Screens(<Zugriffsschlüssel>) benötigt wird, muss folgender Syntaxbeschreibung genügen:

```
[<Grundbildname>.]<Bildfenstername>[:<Bildname>] ...
.<Bildfenstername>[:<Bildname>]
```

Dies bedeutet:

- Der Zugriffsschlüssel drückt die Bildhierarchie aus.
- Im Schlüssel können die Bildnamen an jeder Stelle weggelassen werden.
- Die "AccessPath"-Eigenschaft des "Screen"-Objektes entspricht dem vollständigen Zugriffsschlüssel.
- Bildnamen sind aus Kompatibilitätsgründen zu zukünftigen Versionen immer ohne die Dateierweiterung ".PDL" zu schreiben.
- Das Grundbild ist über den Zugriffsschlüssel "" ansprechbar.

Außerdem wurde festgelegt, dass mit dem Index 1 das Grundbild angesprochen werden kann.

Beispiele

Die Bilder werden durch die Angabe der Hierarchie in der Auflistung adressiert. Dabei gibt es zwei Möglichkeiten, mit oder ohne Verwendung des Bildnamens. In den folgenden Beispielen ist ein Grundbild "BaseScreenName" mit einem Bildfenster "ScreenWindow" projektiert. In dem Bildfenster ist ein Bild "ScreenName" enthalten.

Adressierung mit Verwendung des Bildnamens

```
'VBS8
Set objScreen = HMIRuntime.Screens("BaseScreenName.ScreenWindow:ScreenName")
```

Adressierung ohne Verwendung des Bildnamens

```
'VBS9
Set objScreen = HMIRuntime.Screens("ScreenWindow")
```

Referenzieren des Grundbildes auf verschiedene Arten

```
'VBS10
Set objScreen = HMIRuntime.Screens(1)

'VBS11
Set objScreen = HMIRuntime.Screens("")

'VBS12
Set objScreen = HMIRuntime.Screens("BaseScreenName")
```

Siehe auch

ScreenItem-Objekt (Seite 134)

Refresh-Methode (Seite 761)

Item-Methode (Seite 744)

Count-Eigenschaft (Seite 372)

1.14.2.20 SmartTags-Objekt

Beschreibung

Im Faceplate-Typ wurde die Komponente "HMIRuntime" deaktiviert. Für den Faceplate-Typ wurde die neue Komponente "SmartTags" hinzugefügt. Mit dem SmartTags-Objekt können Sie den Faceplate-Typ dynamisieren. Sie können nur auf die Faceplate-Variablen und die Eigenschaften des Faceplate-Typs zugreifen. Sie können nicht auf den normalen WinCC-Variablenhaushalt zugreifen. Der normale WinCC-Variablenhaushalt steht im Faceplate-Typ nicht zur Verfügung.

Verwendung

Sie können über das Objekt "SmartTags":

- Auf die Faceplate-Variablen in einem Faceplate-Typ zugreifen.
  Syntax: SmartTags("<Variablenname>")
- Auf die Eigenschaften eines Faceplate-Typs zugreifen.
  Syntax: SmartTags("Properties\<Propertyname>")

Beispiel 1

In einem Faceplate-Typ fügen Sie ein Rechteck und einen Button ein. Definieren Sie eine Faceplate-Variable var1. Verschalten Sie die Eigenschaft "Width" des Rechtecks mit der Faceplate-Variablen var1. Dynamisieren Sie das Event "OnClick" des Buttons wie folgt mit VBS.

```
'VBS306
Dim w
w = SmartTags("var1")
w = w + 10
SmartTags("var1") = w
```

Wenn Sie Runtime aktivieren, wird bei jedem Klick auf den Button die Faceplate-Variable um 10 inkrementiert. Dadurch wird die Rechteck-Breite um 10 erhöht.

Direktes Lesen und Schreiben mit Objektreferenz

Im folgenden Beispiel erzeugen Sie mit dem SmartTags-Objekt eine Referenz "w" auf "var1".

Das Referenzieren bietet den Vorteil, dass Sie auf die Variable "var1" direkt zugreifen können.

```
'VBS307
Dim w
Set w = SmartTags("var1")
w.value = w.value + 10
```

Beispiel 2

In einem Faceplate-Typ fügen Sie ein Rechteck und einen Button ein. Definieren Sie die instanzspezifische Eigenschaft "wide". Verknüpfen Sie die Eigenschaft "Width" des Rechtecks mit der instanzspezifischen Eigenschaft "wide". Dynamisieren Sie das Event "OnClick" des Buttons wie folgt mit VBS:

```
'VBS308
Dim w
w = SmartTags("Properties\wide")
SmartTags("Properties\wide") = w + 50
```

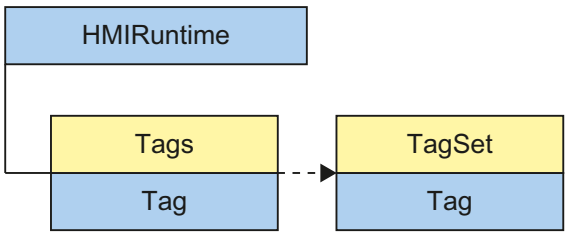
Wenn Sie Runtime aktivieren, wird bei jedem Klick auf den Button die instanzspezifische Eigenschaft "wide" um 50 erhöht. Dadurch wird die Rechteckbreite um 50 erhöht.

Siehe auch

SmartTag-Eigenschaft (Seite 557)

1.14.2.21 Tag-Objekt

Beschreibung



Ein Tag-Objekt (Variable) wird über die Auflistung "Tags" zurückgegeben. Über das Tag-Objekt sind alle Eigenschaften und Methoden einer Variablen ansprechbar.

Beim Anlegen eines Tag-Objektes werden alle Eigenschaften mit folgenden Werten initialisiert:

- Value = VT_EMPTY
- Name = Variablenname
- QualityCode = BAD NON-SPECIFIC
- TimeStamp = 0

- LastError = 0
- ErrorDescription = " "

Hinweis

Eine Zusammenfassung der möglichen Quality Codes finden Sie im WinCC Information System unter dem Stichwort "Kommunikation" > "Diagnose" oder "Kommunikation" > "Quality Codes".

Verwendung

Sie können über das Objekt "Tag":

- Informationen zur Variable auslesen (Eigenschaften "Name", "QualityCode", "TimeStamp", "LastError" und "ErrorDescription")
- Wert einer Variablen setzen (Methode "Write", Eigenschaft "Value")
- Wert einer Variablen lesen (Methode "Read", Eigenschaft "Value")

Lesen des Wertes einer Variablen "Tag1":

```
'VBS13
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
objTag.Read()
MsgBox objTag.Value
```

Deklaration von Variablen in WinCC

Definieren Sie interne Variablen in VB-Skript immer über die Anweisung "Dim", um fehlerhafte Variablenschreibweisen zu verhindern.

Beim Anlegen einer neuen Aktion wird die Anweisung "Option explicit" automatisch und nicht löschbar in den Deklarationsbereich eingetragen.

Verwenden Sie die Anweisung "Option explicit" nicht in Ihrem Code, da es sonst zu Lauzeitfehlern kommen kann.

Beispiel: Deklaration einer VBScript-Variablen "lngVar":

```
'VBS14
Dim lngVar
lngVar = 5
MsgBox lngVar
```

Hinweis

Variablennamen dürfen keine Sonderzeichen enthalten.

Beachten Sie, dass beim Anlegen einer Variable diese keinen Wert enthält (Value = VT_EMPTY). Initialisieren Sie Variablen nach der Deklaration mit einem entsprechenden Wert.

Hinweise zu CrossReference

Alle Variablen, die Sie mit der Standardformulierung

```
HMIRuntime.Tags("Tagname")
```

adressieren, werden durch die CrossReference von WinCC automatisch erfasst und sind in den Bildeigenschaften aufgeführt.

Sollten Sie Variablen mit anderen Formulierungen in Ihrem Code ansprechen, können Sie diese über folgende Sektion der CrossReference bekanntmachen:

```
' WINCC:TAGNAME_SECTION_START
Const TagNameInAction = "TagName"
' WINCC:TAGNAME_SECTION_END
```

Sie können diese Sektion beliebig oft in VBS-Aktionen einfügen.

Hinweis

Die Erfassung zusammengesetzter Variablennamen von der CrossReference kann nicht garantiert werden.

Siehe auch

Name-Eigenschaft (Seite 487)

Beispiel: So lesen Sie Variablenwerte (Seite 803)

Beispiel: So schreiben Sie Variablenwerte (Seite 801)

Write-Methode (Seite 787)

Read-Methode (Seite 757)

Value-Eigenschaft (Seite 655)

TimeStamp-Eigenschaft (Seite 599)

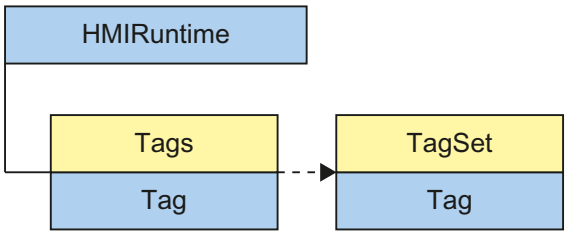
QualityCode-Eigenschaft (Seite 523)

LastError-Eigenschaft (Seite 435)

ErrorDescription-Eigenschaft (Seite 389)

1.14.2.22 Tags-Objekt (Auflistung)

Beschreibung



Die Auflistung "Tags" ermöglicht Ihnen den Zugriff auf Variablen in WinCC Runtime. Als Ergebnis des Zugriffs auf die "Tags"-Auflistung wird ein Objekt vom Typ "Tag" geliefert. Über das Tag-Objekt kann auf alle Eigenschaften und Methoden einer Variable zugegriffen werden.

Hinweis

"Tags" ist eine Auflistung mit eingeschränktem Funktionsumfang. Auf die Variablen innerhalb der Auflistung kann nicht über den Index, sondern nur über den Variablennamen zugegriffen werden. Die Standard-Methoden get_Count und get_NewEnum sind bei der Tags-Auflistung nicht anwendbar.

Verwendung

Der Zugriff auf Variablen der Auflistung erfolgt über:

```
HMIRuntime.Tags("Tagname")
```

Über die Auflistung Tags werden Variablen (Tag-Objekte) für lesenden und schreibenden Zugriff deklariert. Damit der lesende und schreibende Zugriff fehlerfrei ausgeführt wird, müssen die entsprechenden Variablen im WinCC-Variablenhaushalt vorhanden sein.

In VBS können Sie Variablen direkt über den Namen ansprechen und Werte setzen und lesen. Wenn Sie auf zusätzliche Eigenschaften der Variablen zugreifen wollen, z.B. den Qualitycode erfragen, müssen Sie die Variable immer über die Tags-Auflistung ansprechen. Über das zurückgelieferte Tag-Objekt haben Sie Zugriff auf alle Eigenschaften und Methoden der Variablen. Sie müssen für das Objekt eine Instanz bilden, damit z.B. mit HMIRuntime.Tags("Variable").Value=TRUE eine binäre Variable geschrieben wird.

Über die Methode "CreateTagSet" kann ein "TagSet"-Objekt erzeugt werden, das den Zugriff auf mehrere Variablen gleichzeitig ermöglicht.

Beispiel

Wenn Sie Variablen anlegen, haben Sie zwei Möglichkeiten:

- Mit Angabe des Serverprefix: Für Variablen in Mehrplatzsystemen, die nicht lokal abgelegt sind.
- Direkte Verwendung des Variablennamens: Für Variablen, die lokal auf dem Rechner abgelegt sind.

Angabe des Serverprefix

```
'VBS15
Dim objTag
Set objTag = HMIRuntime.Tags("Serverprefix::Tagname")
Wenn Sie das Serverprefix direkt angeben, wird die Eigenschaft "ServerPrefix" mit dem
entsprechenden Wert belegt.
```

Angabe des Variablennamens

```
'VBS16
Dim objTag
Set objTag = HMIRuntime.Tags("Tagname")
Wenn Sie nur den Variablennamen verwenden, werden die Eigenschaften "ServerPrefix" und
"TagPrefix" mit den Werten aus dem aktuellen Kontext (dem aktuellen Bildfenster) belegt.
```

Siehe auch

Beispiel: So lesen Sie Variablenwerte (Seite 803)

Beispiel: So schreiben Sie Variablenwerte (Seite 801)

Item-Methode (Seite 744)

CreateTagSet-Methode (Seite 691)

Tag-Objekt (Seite 146)

1.14.2.23 TagSet-Objekt (Auflistung)

Beschreibung

Das Objekt "TagSet" ermöglicht den gleichzeitigen Zugriff auf mehrere Variablen in einem Aufruf. Dies erfolgt mit besserer Performance und geringerer Kommunikations-Last als beim Einzel-Zugriff auf die verschiedenen Variablen.

Verwendung

Mit dem TagSet-Objekt können Sie :

- Variablen zur Auflistung hinzufügen (Methode "Add")
- Auf die in der Auflistung enthaltenen Tag-Objekte und deren Eigenschaften zugreifen (Methode "Item")
- Alle Variablen der Auflistung schreiben (Methode "Write")
- Alle Variablen der Auflistung lesen (Methode "Read")
- Einzelne Variablen aus der Auflistung entfernen (Methode "Remove")
- Alle Variablen aus der Auflistung entfernen (Methode "RemoveAll")

Der Zugriff auf Variablen der Auflistung erfolgt über:

```
'VBS169
Dim myTags
myTags = HMIRuntime.Tags.CreateTagSet
myTags("Tagname")
```

Damit der lesende und schreibende Zugriff auf Variablen (Tag-Objekte) der Auflistung fehlerfrei ausgeführt werden kann, müssen die entsprechenden Variablen im WinCC-Variablenhaushalt vorhanden sein.

Wenn beim lesenden oder schreibenden Zugriff ein Fehler aufgetreten ist, gibt die eingesetzte Methode über die Eigenschaften "LastError" und "ErrorDescription" eine Fehlermeldung zurück.

Das synchrone Schreiben und Lesen der Variablen ist möglich. Mit dem optionalen Parameter "Writemode" werden mit "1" Prozess-Variablen direkt in das AS geschrieben, z.B. "group.Write 1". Mit dem optionalen Parameter "Readmode" werden mit "1" Prozess-Variablen nicht angemeldet sondern direkt aus dem AS bzw. Kanal gelesen, z.B. "group.Read 1".

Beispiel

Das folgende Beispiel zeigt, wie man ein TagSet-Objekt erzeugt, Variablen hinzufügt und Werte schreibt.

```
'VBS168
Build a Reference to the TagSet Object
Dim group
```

```
Set group = HMIRuntime.Tags.CreateTagSet
'Add Tags to the Collection
group.Add "Motor1"
group.Add "Motor2"
'Set the Values of the Tags
group("Motor1").Value = 3
group("Motor2").Value = 9
'Write the Values to the DataManager
group.Write
```

Siehe auch

LastError-Eigenschaft (Seite 435)

Beispiel: So lesen Sie Variablenwerte (Seite 803)

Beispiel: So schreiben Sie Variablenwerte (Seite 801)

Write-Methode (Seite 787)

RemoveAll-Methode (Seite 766)

Remove-Methode (Seite 762)

Read-Methode (Seite 757)

Item-Methode (Seite 744)

ErrorDescription-Eigenschaft (Seite 389)

Count-Eigenschaft (Seite 372)

Add-Methode (Seite 688)

Tags-Objekt (Auflistung) (Seite 149)

Tag-Objekt (Seite 146)

1.14.3 Objekt-Typen des Objekts ScreenItem

1.14.3.1 Objekt-Typen des Objekts ScreenItem

Einleitung

Im Folgenden finden Sie alle verfügbaren Typen des Objektes "ScreenItem".

Die Ausprägungen des Objektes "ScreenItem" stellen alle im WinCC Graphics Designer verfügbaren Grafikobjekte dar.

Die Objekttypen sind entsprechend der Anordnung im Graphics Designer in folgende Gruppen aufgeteilt:

- Standard-Objekte
- Smart-Objekte

- Windows-Objekte
- Rohr-Objekte
- Controls

Weiterhin gibt es die Objekttypen

- Anwender-Objekt
- Gruppe

Siehe auch

ScreenItems-Objekt (Auflistung) (Seite 138)

ScreenItem-Objekt (Seite 134)

Gruppe (Seite 294)

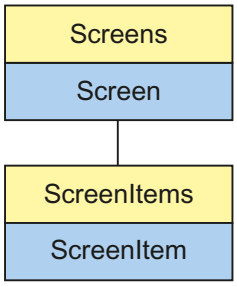
Anwender-Objekt (Seite 293)

Controls (Seite 226)

1.14.3.2 Standard-Objekte

Ellipse

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Ellipse".

Typkennzeichnung in VBS

HMIEllipse

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Ellipse1" um 10 Pixel nach rechts verschoben:

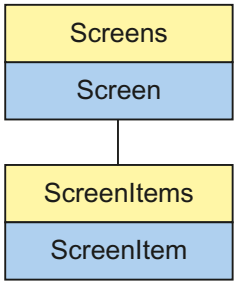
```
'VBS17
Dim objEllipse
Set objEllipse = ScreenItems("Ellipse1")
objEllipse.Left = objEllipse.Left + 10
```

Siehe auch

Fillstyle-Eigenschaft (Seite 398)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
RadiusWidth-Eigenschaft (Seite 525)
RadiusHeight-Eigenschaft (Seite 525)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashBorderColor-Eigenschaft (Seite 402)
FlashBackColor-Eigenschaft (Seite 401)
FillingIndex-Eigenschaft (Seite 397)
Filling-Eigenschaft (Seite 397)
FillColor-Eigenschaft (Seite 396)
Enabled-Eigenschaft (Seite 386)
BorderWidth-Eigenschaft (Seite 337)
BorderStyle-Eigenschaft (Seite 337)
BorderFlashColorOn-Eigenschaft (Seite 337)
BorderFlashColorOff-Eigenschaft (Seite 336)
BorderColor-Eigenschaft (Seite 335)
BorderBackColor-Eigenschaft (Seite 335)
BackFlashColorOn-Eigenschaft (Seite 319)
BackFlashColorOff-Eigenschaft (Seite 318)
BackColor-Eigenschaft (Seite 316)
Layer-Eigenschaft (Seite 437)

Ellipsenbogen

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Ellipsenbogen".

Typkennzeichnung in VBS

HMIEllipticalArc

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "EllipseArc1" um 10 Pixel nach rechts verschoben:

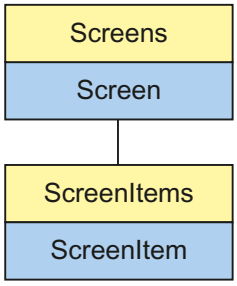
```
'VBS18
Dim objEllipseArc
Set objEllipseArc = ScreenItems("EllipseArc1")
objEllipseArc.Left = objEllipseArc.Left + 10
```

Siehe auch

RadiusHeight-Eigenschaft (Seite 525)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
StartAngle-Eigenschaft (Seite 563)
RadiusWidth-Eigenschaft (Seite 525)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashBorderColor-Eigenschaft (Seite 402)
EndAngle-Eigenschaft (Seite 388)
Enabled-Eigenschaft (Seite 386)
BorderWidth-Eigenschaft (Seite 337)
BorderStyle-Eigenschaft (Seite 337)
BorderFlashColorOn-Eigenschaft (Seite 337)
BorderFlashColorOff-Eigenschaft (Seite 336)
BorderColor-Eigenschaft (Seite 335)
BorderBackColor-Eigenschaft (Seite 335)
Layer-Eigenschaft (Seite 437)

Ellipsensegment

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Ellipsensegment".

Typkennzeichnung in VBS

HMIEllipseSegment

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "EllipseSegment1" um 10 Pixel nach rechts verschoben:

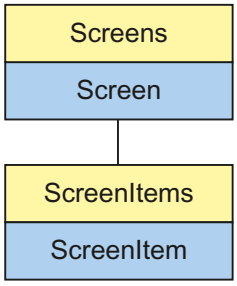
```
'VBS19
Dim objEllipseSeg
Set objEllipseSeg = ScreenItems("EllipseSegment1")
objEllipseSeg.Left = objEllipseSeg.Left + 10
```

Siehe auch

Layer-Objekt (Seite 129)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
StartAngle-Eigenschaft (Seite 563)
RadiusWidth-Eigenschaft (Seite 525)
RadiusHeight-Eigenschaft (Seite 525)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Height-Eigenschaft (Seite 421)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashBorderColor-Eigenschaft (Seite 402)
FlashBackColor-Eigenschaft (Seite 401)
Fillstyle-Eigenschaft (Seite 398)
FillingIndex-Eigenschaft (Seite 397)
Filling-Eigenschaft (Seite 397)
FillColor-Eigenschaft (Seite 396)
EndAngle-Eigenschaft (Seite 388)
Enabled-Eigenschaft (Seite 386)
BorderWidth-Eigenschaft (Seite 337)
BorderStyle-Eigenschaft (Seite 337)
BorderFlashColorOn-Eigenschaft (Seite 337)
BorderFlashColorOff-Eigenschaft (Seite 336)
BorderColor-Eigenschaft (Seite 335)
BorderBackColor-Eigenschaft (Seite 335)
BackFlashColorOn-Eigenschaft (Seite 319)
BackFlashColorOff-Eigenschaft (Seite 318)
BackColor-Eigenschaft (Seite 316)
Layer-Eigenschaft (Seite 437)

Kreis

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Kreis".

Typkennzeichnung in VBS

HMICircle

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Circle1" um 10 Pixel nach rechts verschoben:

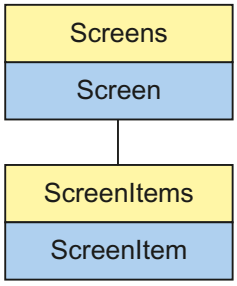
```
'VBS20
Dim objCircle
Set objCircle = ScreenItems("Circle1")
objCircle.Left = objCircle.Left + 10
```

Siehe auch

Eigenschaften (Seite 295)
BorderStyle-Eigenschaft (Seite 337)
Activate-Methode (Seite 686)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
Radius-Eigenschaft (Seite 524)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashBorderColor-Eigenschaft (Seite 402)
FlashBackColor-Eigenschaft (Seite 401)
Fillstyle-Eigenschaft (Seite 398)
FillingIndex-Eigenschaft (Seite 397)
Filling-Eigenschaft (Seite 397)
FillColor-Eigenschaft (Seite 396)
Enabled-Eigenschaft (Seite 386)
BorderWidth-Eigenschaft (Seite 337)
BorderFlashColorOn-Eigenschaft (Seite 337)
BorderFlashColorOff-Eigenschaft (Seite 336)
BorderColor-Eigenschaft (Seite 335)
BorderBackColor-Eigenschaft (Seite 335)
BackFlashColorOn-Eigenschaft (Seite 319)
BackFlashColorOff-Eigenschaft (Seite 318)
BackColor-Eigenschaft (Seite 316)
Layer-Eigenschaft (Seite 437)

Kreisbogen

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Kreisbogen".

Typkennzeichnung in VBS

HMICircularArc

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "CircularArc1" um 10 Pixel nach rechts verschoben:

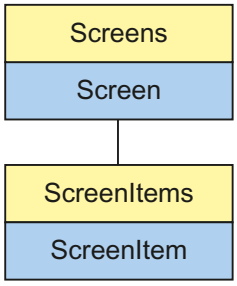
```
'VBS21
Dim objCircularArc
Set objCircularArc = ScreenItems("CircularArc1")
objCircularArc.Left = objCircularArc.Left + 10
```

Siehe auch

StartAngle-Eigenschaft (Seite 563)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
Radius-Eigenschaft (Seite 524)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashBorderColor-Eigenschaft (Seite 402)
EndAngle-Eigenschaft (Seite 388)
Enabled-Eigenschaft (Seite 386)
BorderWidth-Eigenschaft (Seite 337)
BorderStyle-Eigenschaft (Seite 337)
BorderFlashColorOn-Eigenschaft (Seite 337)
BorderFlashColorOff-Eigenschaft (Seite 336)
BorderColor-Eigenschaft (Seite 335)
BorderBackColor-Eigenschaft (Seite 335)
Layer-Eigenschaft (Seite 437)

Kreissegment

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Kreissegment".

Typkennzeichnung in VBS

HMICircleSegment

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "PieSegment1" um 10 Pixel nach rechts verschoben:

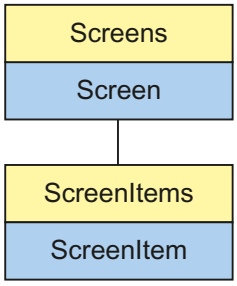
```
'VBS22
Dim objCircleSeg
Set objCircleSeg = ScreenItems("PieSegment1")
objCircleSeg.Left = objCircleSeg.Left + 10
```

Siehe auch

Type-Eigenschaft (Seite 642)
BorderColor-Eigenschaft (Seite 335)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
StartAngle-Eigenschaft (Seite 563)
Radius-Eigenschaft (Seite 524)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashBorderColor-Eigenschaft (Seite 402)
FlashBackColor-Eigenschaft (Seite 401)
Fillstyle-Eigenschaft (Seite 398)
FillingIndex-Eigenschaft (Seite 397)
Filling-Eigenschaft (Seite 397)
FillColor-Eigenschaft (Seite 396)
EndAngle-Eigenschaft (Seite 388)
Enabled-Eigenschaft (Seite 386)
BorderWidth-Eigenschaft (Seite 337)
BorderStyle-Eigenschaft (Seite 337)
BorderFlashColorOn-Eigenschaft (Seite 337)
BorderFlashColorOff-Eigenschaft (Seite 336)
BorderBackColor-Eigenschaft (Seite 335)
BackFlashColorOn-Eigenschaft (Seite 319)
BackFlashColorOff-Eigenschaft (Seite 318)
BackColor-Eigenschaft (Seite 316)
Layer-Eigenschaft (Seite 437)

Linie

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Linie".

Typkennzeichnung in VBS

HMILine

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Line1" um 10 Pixel nach rechts verschoben:

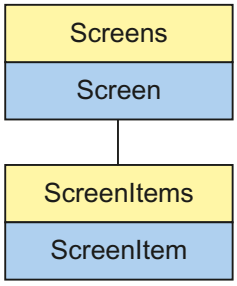
```
'VBS23
Dim objLine
Set objLine = ScreenItems("Line1")
objLine.Left = objLine.Left + 10
```

Siehe auch

PasswordLevel-Eigenschaft (Seite 508)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
RotationAngle-Eigenschaft (Seite 529)
ReferenceRotationTop-Eigenschaft (Seite 527)
ReferenceRotationLeft-Eigenschaft (Seite 526)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashBorderColor-Eigenschaft (Seite 402)
Enabled-Eigenschaft (Seite 386)
BorderWidth-Eigenschaft (Seite 337)
BorderStyle-Eigenschaft (Seite 337)
BorderFlashColorOn-Eigenschaft (Seite 337)
BorderFlashColorOff-Eigenschaft (Seite 336)
BorderEndStyle-Eigenschaft (Seite 336)
BorderColor-Eigenschaft (Seite 335)
BorderBackColor-Eigenschaft (Seite 335)
Layer-Eigenschaft (Seite 437)

Polygon

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Polygon".

Typkennzeichnung in VBS

HMIPolygon

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Polygon1" um 10 Pixel nach rechts verschoben:

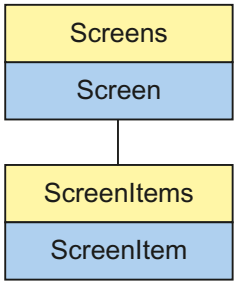
```
'VBS24
Dim objPolygon
Set objPolygon = ScreenItems("Polygon1")
objPolygon.Left = objPolygon.Left + 10
```

Siehe auch

ReferenceRotationTop-Eigenschaft (Seite 527)
BackFlashColorOn-Eigenschaft (Seite 319)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
RotationAngle-Eigenschaft (Seite 529)
ReferenceRotationLeft-Eigenschaft (Seite 526)
PointCount-Eigenschaft (Seite 518)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Index-Eigenschaft (Seite 428)
Height-Eigenschaft (Seite 421)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashBorderColor-Eigenschaft (Seite 402)
FlashBackColor-Eigenschaft (Seite 401)
Fillstyle-Eigenschaft (Seite 398)
FillingIndex-Eigenschaft (Seite 397)
Filling-Eigenschaft (Seite 397)
FillColor-Eigenschaft (Seite 396)
Enabled-Eigenschaft (Seite 386)
BorderWidth-Eigenschaft (Seite 337)
BorderStyle-Eigenschaft (Seite 337)
BorderFlashColorOn-Eigenschaft (Seite 337)
BorderFlashColorOff-Eigenschaft (Seite 336)
BorderColor-Eigenschaft (Seite 335)
BorderBackColor-Eigenschaft (Seite 335)
BackFlashColorOff-Eigenschaft (Seite 318)
BackColor-Eigenschaft (Seite 316)
ActualPointTop-Eigenschaft (Seite 300)
ActualPointLeft-Eigenschaft (Seite 299)

Polygonzug

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Polygonzug".

Typkennzeichnung in VBS

HMIPolyLine

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Polyline1" um 10 Pixel nach rechts verschoben:

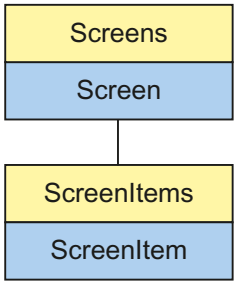
```
'VBS25
Dim objPolyline
Set objPolyline = ScreenItems("Polyline1")
objPolyline.Left = objPolyline.Left + 10
```

Siehe auch

Layer-Objekt (Seite 129)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
RotationAngle-Eigenschaft (Seite 529)
ReferenceRotationTop-Eigenschaft (Seite 527)
ReferenceRotationLeft-Eigenschaft (Seite 526)
PointCount-Eigenschaft (Seite 518)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Index-Eigenschaft (Seite 428)
Height-Eigenschaft (Seite 421)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashBorderColor-Eigenschaft (Seite 402)
Enabled-Eigenschaft (Seite 386)
BorderWidth-Eigenschaft (Seite 337)
BorderStyle-Eigenschaft (Seite 337)
BorderFlashColorOn-Eigenschaft (Seite 337)
BorderFlashColorOff-Eigenschaft (Seite 336)
BorderEndStyle-Eigenschaft (Seite 336)
BorderColor-Eigenschaft (Seite 335)
BorderBackColor-Eigenschaft (Seite 335)
ActualPointTop-Eigenschaft (Seite 300)
ActualPointLeft-Eigenschaft (Seite 299)
Layer-Eigenschaft (Seite 437)

Rechteck

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Rechteck".

Typkennzeichnung in VBS

HMIRectangle

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Rectangle1" um 10 Pixel nach rechts verschoben:

```
'VBS26
Dim objRectangle
Set objRectangle = ScreenItems("Rectangle1")
objRectangle.Left = objRectangle.Left + 10
```

Hinweise zur Fehlerbehandlung

Rechteck und Rundrechteck werden im Objektmodell auf einen Typ "HMIRectangle" abgebildet. Da die beiden Objekte dennoch teilweise unterschiedliche Eigenschaften aufweisen, sollten Sie das Vorhandensein der Eigenschaften (dynamische Typ-Ermittlung in Runtime) über eine Ausnahmebehandlung abfragen. Die Ausnahmebehandlung wird durch folgende Anweisung für die entsprechende Prozedur aktiviert:

```
On Error Resume Next
```

Die Anweisung führt dazu, dass die VBScript-Engine nach einem Laufzeitfehler die Folgeanweisung ausführt.

In der Folgezeile kann dann der Fehlercode über das Err-Objekt überprüft werden. Um die Behandlung von Laufzeitfehlern im Skript wieder abzuschalten, wird folgende Anweisung verwendet:

```
On Error Goto 0
```

Die Fehlerbehandlung ist immer auf die Prozedurebene bezogen. Löst ein Skript in einer Prozedur einen Fehler aus, prüft VBScript, ob auf dieser Ebene eine Fehlerbehandlung implementiert ist. Ist dies nicht der Fall, geht die Kontrolle eine Ebene höher (zur rufenden Prozedur). Gibt es dort ebenfalls keine Fehlerbehandlung, wird die Kontrolle wiederum eine Ebene höher übergeben. Dies setzt sich solange fort, bis entweder die oberste Modulebene erreicht ist oder der Code zur Laufzeitfehlerbehandlung gefunden wird. Fehlt die Aktivierung der Runtime-Fehlerbehandlung, wird die Kontrolle auf der Hauptebene an die interne VBScript-Laufzeitfehlerbehandlung übergeben. Diese zeigt dann einen Fehlerdialog an und bricht das Skript ab.

Die "On Error Resume Next"-Anweisung kann auf jeder Ebene (also auch in Prozeduren) eingebaut werden. Mit dem Einsatz der Fehlerbehandlung ist prinzipiell feststellbar, ob der Anwender den gewünschten Implementierungstyp auch wirklich verwendet.

Außerdem kann garantiert werden, dass es nicht zu einem Abbruch der Ausführung aufgrund eines fehlerhaften Zugriffs auf das Objekt kommt.

Beispiel zur Fehlerbehandlung

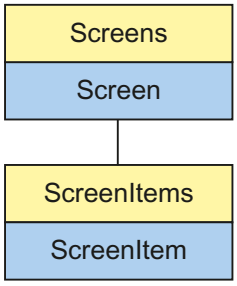
```
Sub OnClick(ByVal Item)
'VBS27
Dim objScreenItem
'
'Activation of errorhandling:
On Error Resume Next
For Each objScreenItem In ScreenItems
If "HMIRectangle" = objScreenItem.Type Then
'
'=== Property "RoundCornerHeight" only available for RoundRectangle
objScreenItem.RoundCornerHeight = objScreenItem.RoundCornerHeight * 2
If 0 <> Err.Number Then
HMIRuntime.Trace objScreenItem.Name & ": no RoundedRectangle" & vbCrLf
'
'Delete error message
Err.Clear
End If
End If
Next
On Error Goto 0  'Deactivation of errorhandling
End Sub
```

Siehe auch

Eigenschaften (Seite 295)
BorderFlashColorOn-Eigenschaft (Seite 337)
Activate-Methode (Seite 686)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashBorderColor-Eigenschaft (Seite 402)
FlashBackColor-Eigenschaft (Seite 401)
Fillstyle-Eigenschaft (Seite 398)
FillingIndex-Eigenschaft (Seite 397)
Filling-Eigenschaft (Seite 397)
FillColor-Eigenschaft (Seite 396)
Enabled-Eigenschaft (Seite 386)
BorderWidth-Eigenschaft (Seite 337)
BorderStyle-Eigenschaft (Seite 337)
BorderFlashColorOff-Eigenschaft (Seite 336)
BorderColor-Eigenschaft (Seite 335)
BorderBackColor-Eigenschaft (Seite 335)
BackFlashColorOn-Eigenschaft (Seite 319)
BackFlashColorOff-Eigenschaft (Seite 318)
BackColor-Eigenschaft (Seite 316)
Layer-Eigenschaft (Seite 437)

Rundrechteck

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Rundrechteck".

Typkennzeichnung in VBS

HMIRoundRectangle

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "RoundedRectangle1" um 10 Pixel nach rechts verschoben:

```
'VBS28
Dim objRoundedRectangle
Set objRoundedRectangle = ScreenItems("RoundedRectangle1")
objRoundedRectangle.Left = objRoundedRectangle.Left + 10
```

Hinweise zur Fehlerbehandlung

Rechteck und Rundrechteck werden im Objektmodell auf einen Typ "HMIRectangle" abgebildet. Da die beiden Objekte dennoch teilweise unterschiedliche Eigenschaften aufweisen, sollten Sie das Vorhandensein der Eigenschaften (dynamische Typ-Ermittlung in Runtime) über eine Ausnahmebehandlung abfragen. Die Ausnahmebehandlung wird durch folgende Anweisung für die entsprechende Prozedur aktiviert:

```
On Error Resume Next
```

Die Anweisung führt dazu, dass die VBScript-Engine nach einem Laufzeitfehler die Folgeanweisung ausführt.

In der Folgezeile kann dann der Fehlercode über das Err-Objekt überprüft werden. Um die Behandlung von Laufzeitfehlern im Skript wieder abzuschalten, wird folgende Anweisung verwendet:

```
On Error Goto 0
```

Die Fehlerbehandlung ist immer auf die Prozedurebene bezogen. Löst ein Skript in einer Prozedur einen Fehler aus, prüft VBScript, ob auf dieser Ebene eine Fehlerbehandlung implementiert ist. Ist dies nicht der Fall, geht die Kontrolle eine Ebene höher (zur rufenden Prozedur). Gibt es dort ebenfalls keine Fehlerbehandlung, wird die Kontrolle wiederum eine Ebene höher übergeben. Dies setzt sich solange fort, bis entweder die oberste Modulebene erreicht ist oder der Code zur Laufzeitfehlerbehandlung gefunden wird. Fehlt die Aktivierung der Runtime-Fehlerbehandlung, wird die Kontrolle auf der Hauptebene an die interne VBScript-Laufzeitfehlerbehandlung übergeben. Diese zeigt dann einen Fehlerdialog an und bricht das Skript ab.

Die "On Error Resume Next"-Anweisung kann auf jeder Ebene (also auch in Prozeduren) eingebaut werden. Mit dem Einsatz der Fehlerbehandlung ist prinzipiell feststellbar, ob der Anwender den gewünschten Implementierungstyp auch wirklich verwendet.

Außerdem kann garantiert werden, dass es nicht zu einem Abbruch der Ausführung aufgrund eines fehlerhaften Zugriffs auf das Objekt kommt.

Beispiel zur Fehlerbehandlung

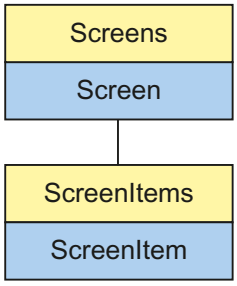
```
Sub OnClick(ByVal Item)
'VBS29
Dim objScreenItem
On Error Resume Next    'Activation of errorhandling
For Each objScreenItem In ScreenItems
If "HMIRectangle" = objScreenItem.Type Then
'
'=== Property "RoundCornerHeight" available only for RoundRectangle
objScreenItem.RoundCornerHeight = objScreenItem.RoundCornerHeight * 2
If 0 <> Err.Number Then
HMIRuntime.Trace objScreenItem.ObjectName & ": no RoundedRectangle" & vbCrLf
Err.Clear    'Delete errormessage
End If
End If
Next
On Error Goto 0    'Deactivation of errorhandling
End Sub
```

Siehe auch

FlashBackColor-Eigenschaft (Seite 401)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
RoundCornerWidth-Eigenschaft (Seite 530)
RoundCornerHeight-Eigenschaft (Seite 530)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashBorderColor-Eigenschaft (Seite 402)
Fillstyle-Eigenschaft (Seite 398)
FillingIndex-Eigenschaft (Seite 397)
Filling-Eigenschaft (Seite 397)
FillColor-Eigenschaft (Seite 396)
Enabled-Eigenschaft (Seite 386)
BorderWidth-Eigenschaft (Seite 337)
BorderStyle-Eigenschaft (Seite 337)
BorderFlashColorOn-Eigenschaft (Seite 337)
BorderFlashColorOff-Eigenschaft (Seite 336)
BorderColor-Eigenschaft (Seite 335)
BorderBackColor-Eigenschaft (Seite 335)
BackFlashColorOn-Eigenschaft (Seite 319)
BackFlashColorOff-Eigenschaft (Seite 318)
BackColor-Eigenschaft (Seite 316)
Layer-Eigenschaft (Seite 437)

Statischer Text

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Statischer Text".

Typkennzeichnung in VBS

HMITextField

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "StaticText1" um 10 Pixel nach rechts verschoben:

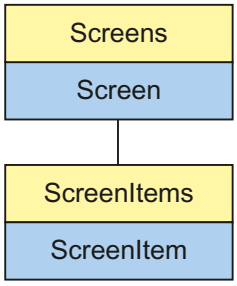
```
'VBS30
Dim objStaticText
Set objStaticText = ScreenItems("StaticText1")
objStaticText.Left = objStaticText.Left + 10
```

Siehe auch

ObjectName-Eigenschaft (Seite 490)
BorderFlashColorOn-Eigenschaft (Seite 337)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
Textliste (Seite 208)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
Orientation-Eigenschaft (Seite 504)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
ForeFlashColorOn-Eigenschaft (Seite 413)
ForeFlashColorOff-Eigenschaft (Seite 413)
ForeColor-Eigenschaft (Seite 412)
FontUnderline-Eigenschaft (Seite 411)
FontSize-Eigenschaft (Seite 411)
FontName-Eigenschaft (Seite 410)
FontItalic-Eigenschaft (Seite 409)
FontBold-Eigenschaft (Seite 409)
FlashRateForeColor-Eigenschaft (Seite 405)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashForeColor-Eigenschaft (Seite 402)
FlashBorderColor-Eigenschaft (Seite 402)
FlashBackColor-Eigenschaft (Seite 401)
Fillstyle-Eigenschaft (Seite 398)
FillingIndex-Eigenschaft (Seite 397)
Filling-Eigenschaft (Seite 397)
FillColor-Eigenschaft (Seite 396)
Enabled-Eigenschaft (Seite 386)
BorderWidth-Eigenschaft (Seite 337)
BorderStyle-Eigenschaft (Seite 337)
BorderFlashColorOff-Eigenschaft (Seite 336)

Verbinder

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Verbinder".

Typkennzeichnung in VBS

HMIConnector

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Connector1" um 10 Pixel nach rechts verschoben:

```
'VBS31
Dim objConnector
Set objConnector = ScreenItems("Connector1")
objConnector.Left = objConnector.Left + 10
```

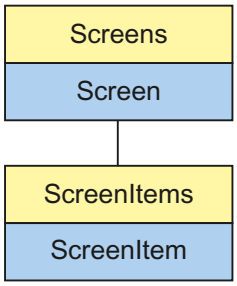
Siehe auch

ScreenItems-Objekt (Auflistung) (Seite 138)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
TopConnectedObjectName-Eigenschaft (Seite 619)
TopConnectedConnectionPointIndex-Eigenschaft (Seite 619)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
Parent-Eigenschaft (Seite 506)
Orientation-Eigenschaft (Seite 504)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Eigenschaft (Seite 437)
Height-Eigenschaft (Seite 421)
Enabled-Eigenschaft (Seite 386)
BottomConnectedObjectName-Eigenschaft (Seite 338)
BottomConnectedConnectionPointIndex-Eigenschaft (Seite 338)

1.14.3.3 Smart-Objekte

3D-Balken

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "3D-Balken".

Typkennzeichnung in VBS

HMIBar

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "3DBar1" um 10 Pixel nach rechts verschoben:

```
'VBS32
Dim objBar
Set objBar = ScreenItems("3DBar1")
objBar.Left = objBar.Left + 10
```

Hinweise zur Fehlerbehandlung

Balken und 3D-Balken werden im Objektmodell auf einen Typ "HMIBar" abgebildet. Da die beiden Objekte dennoch teilweise unterschiedliche Eigenschaften aufweisen, sollten Sie das Vorhandensein der Eigenschaften (dynamische Typ-Ermittlung in Runtime) über eine Ausnahmebehandlung abfragen. Die Ausnahmebehandlung wird durch folgende Anweisung für die entsprechende Prozedur aktiviert:

```
On Error Resume Next
```

Die Anweisung führt dazu, dass die VBScript-Engine nach einem Laufzeitfehler die Folgeanweisung ausführt.

In der Folgezeile kann dann der Fehlercode über das Err-Objekt überprüft werden. Um die Behandlung von Laufzeitfehlern im Skript wieder abzuschalten, wird folgende Anweisung verwendet:

```
On Error Goto 0
```

Die Fehlerbehandlung ist immer auf die Prozedurebene bezogen. Löst ein Skript in einer Prozedur einen Fehler aus, prüft VBScript, ob auf dieser Ebene eine Fehlerbehandlung implementiert ist. Ist dies nicht der Fall, geht die Kontrolle eine Ebene höher (zur rufenden Prozedur). Gibt es dort ebenfalls keine Fehlerbehandlung, wird die Kontrolle wiederum eine Ebene höher übergeben. Dies setzt sich solange fort, bis entweder die oberste Modulebene erreicht ist oder der Code zur Laufzeitfehlerbehandlung gefunden wird. Fehlt die Aktivierung der Runtime-Fehlerbehandlung, wird die Kontrolle auf der Hauptebene an die interne VBScript-Laufzeitfehlerbehandlung übergeben. Diese zeigt dann einen Fehlerdialog an und bricht das Skript ab.

Die "On Error Resume Next"-Anweisung kann auf jeder Ebene (also auch in Prozeduren) eingebaut werden. Mit dem Einsatz der Fehlerbehandlung ist prinzipiell feststellbar, ob der Anwender den gewünschten Implementierungstyp auch wirklich verwendet.

Außerdem kann garantiert werden, dass es nicht zu einem Abbruch der Ausführung aufgrund eines fehlerhaften Zugriffs auf das Objekt kommt.

Beispiel zur Fehlerbehandlung

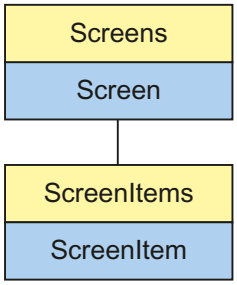
```
'VBS148
Sub OnClick(ByVal Item)
Dim objScreenItem
'
'Activation of errorhandling:
On Error Resume Next
For Each objScreenItem In ScreenItems
If "HMIBar" = objScreenItem.Type Then
'
'=== Property "Layer00Value" only available for 3D bar
objScreenItem.Layer00Value = objScreenItem.Layer00Value * 2
If 0 <> Err.Number Then
HMIRuntime.Trace objScreenItem.Name & ": no 3D bar" & vbCrLf
'
'Delete error message
Err.Clear
End If
End If
Next
On Error Goto 0 'Deactivation of errorhandling
End Sub
```

Siehe auch

Type-Eigenschaft (Seite 642)
Layer08Color-Eigenschaft (Seite 444)
BorderStyle-Eigenschaft (Seite 337)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
ZeroPointValue-Eigenschaft (Seite 684)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
Process-Eigenschaft (Seite 521)
PredefinedAngles-Eigenschaft (Seite 520)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Min-Eigenschaft (Seite 484)
Max-Eigenschaft (Seite 466)
LightEffect-Eigenschaft (Seite 455)
Left-Eigenschaft (Seite 454)
Layer10Value-Eigenschaft (Seite 453)
Layer09Value-Eigenschaft (Seite 453)
Layer08Value-Eigenschaft (Seite 452)
Layer07Value-Eigenschaft (Seite 452)
Layer06Value-Eigenschaft (Seite 452)
Layer05Value-Eigenschaft (Seite 451)
Layer04Value-Eigenschaft (Seite 451)
Layer03Value-Eigenschaft (Seite 451)
Layer02Value-Eigenschaft (Seite 450)
Layer01Value-Eigenschaft (Seite 450)
Layer00Value-Eigenschaft (Seite 450)
Layer10Color-Eigenschaft (Seite 445)
Layer09Color-Eigenschaft (Seite 444)
Layer07Color-Eigenschaft (Seite 444)
Layer06Color-Eigenschaft (Seite 443)
Layer05Color-Eigenschaft (Seite 443)
Layer04Color-Eigenschaft (Seite 443)
Layer03Color-Eigenschaft (Seite 442)
Layer02Color-Eigenschaft (Seite 442)

Applikationsfenster

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Applikationsfenster".

Typkennzeichnung in VBS

HMIApplicationWindow

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "ApplicationWindow1" um 10 Pixel nach rechts verschoben:

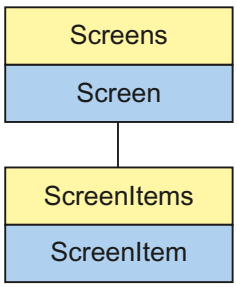
```
'VBS33
Dim objAppWindow
Set objAppWindow = ScreenItems("ApplicationWindow1")
objAppWindow.Left = objAppWindow.Left + 10
```

Siehe auch

Eigenschaften (Seite 295)
Activate-Methode (Seite 686)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
WindowBorder-Eigenschaft (Seite 674)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
Template-Eigenschaft (Seite 574)
Parent-Eigenschaft (Seite 506)
OnTop-Eigenschaft (Seite 494)
ObjectName-Eigenschaft (Seite 490)
Moveable-Eigenschaft (Seite 485)
MaximizeButton-Eigenschaft (Seite 467)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
Enabled-Eigenschaft (Seite 386)
CloseButton-Eigenschaft (Seite 350)
Caption-Eigenschaft (Seite 342)
Application-Eigenschaft (Seite 307)

Balken

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Balken".

Typkennzeichnung in VBS

HMIBar

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Bar1" um 10 Pixel nach rechts verschoben:

```
'VBS34
Dim objBar
Set objBar = ScreenItems("Bar1")
objBar.Left = objBar.Left + 10
```

Hinweise zur Fehlerbehandlung

Balken und 3D-Balken werden im Objektmodell auf einen Typ "HMIBar" abgebildet. Da die beiden Objekte dennoch teilweise unterschiedliche Eigenschaften aufweisen, sollten Sie das Vorhandensein der Eigenschaften (dynamische Typ-Ermittlung in Runtime) über eine Ausnahmebehandlung abfragen. Die Ausnahmebehandlung wird durch folgende Anweisung für die entsprechende Prozedur aktiviert:

```
On Error Resume Next
```

Die Anweisung führt dazu, dass die VBScript-Engine nach einem Laufzeitfehler die Folgeanweisung ausführt.

In der Folgezeile kann dann der Fehlercode über das Err-Objekt überprüft werden. Um die Behandlung von Laufzeitfehlern im Skript wieder abzuschalten, wird folgende Anweisung verwendet:

```
On Error Goto 0
```

Die Fehlerbehandlung ist immer auf die Prozedurebene bezogen. Löst ein Skript in einer Prozedur einen Fehler aus, prüft VBScript, ob auf dieser Ebene eine Fehlerbehandlung implementiert ist. Ist dies nicht der Fall, geht die Kontrolle eine Ebene höher (zur rufenden Prozedur). Gibt es dort ebenfalls keine Fehlerbehandlung, wird die Kontrolle wiederum eine Ebene höher übergeben. Dies setzt sich solange fort, bis entweder die oberste Modulebene erreicht ist oder der Code zur Laufzeitfehlerbehandlung gefunden wird. Fehlt die Aktivierung der Runtime-Fehlerbehandlung, wird die Kontrolle auf der Hauptebene an die interne VBScript-Laufzeitfehlerbehandlung übergeben. Diese zeigt dann einen Fehlerdialog an und bricht das Skript ab.

Die "On Error Resume Next"-Anweisung kann auf jeder Ebene (also auch in Prozeduren) eingebaut werden. Mit dem Einsatz der Fehlerbehandlung ist prinzipiell feststellbar, ob der Anwender den gewünschten Implementierungstyp auch wirklich verwendet.

Außerdem kann garantiert werden, dass es nicht zu einem Abbruch der Ausführung aufgrund eines fehlerhaften Zugriffs auf das Objekt kommt.

Beispiel zur Fehlerbehandlung

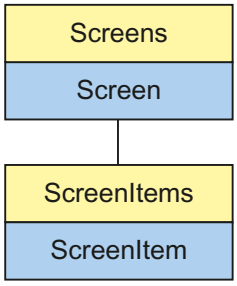
```
'VBS147
Sub OnClick(ByVal Item)
Dim objScreenItem
'
'Activation of errorhandling:
On Error Resume Next
For Each objScreenItem In ScreenItems
If "HMIBar" = objScreenItem.Type Then
'
'=== Property "LimitHigh4" only available for bar
objScreenItem.LimitHigh4 = objScreenItem.LimitHigh4 * 2
If 0 <> Err.Number Then
HMIRuntime.Trace objScreenItem.Name & ": no bar" & vbCrLf
'
'Delete error message
Err.Clear
End If
End If
Next
On Error Goto 0 'Deactivation of errorhandling
End Sub
```

Siehe auch

ToolTipText-Eigenschaft (Seite 617)
Layer-Objekt (Seite 129)
ColorChangeType-Eigenschaft (Seite 355)
Average-Eigenschaft (Seite 314)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
ZeroPointValue-Eigenschaft (Seite 684)
ZeroPoint-Eigenschaft (Seite 683)
Width-Eigenschaft (Seite 673)
WarningLow-Eigenschaft (Seite 673)
WarningHigh-Eigenschaft (Seite 672)
Visible-Eigenschaft (Seite 671)
TypeWarningLow-Eigenschaft (Seite 646)
TypeWarningHigh-Eigenschaft (Seite 646)
TypeToleranceLow-Eigenschaft (Seite 646)
TypeToleranceHigh-Eigenschaft (Seite 645)
TypeLimitLow5-Eigenschaft (Seite 645)
TypeLimitLow4-Eigenschaft (Seite 645)
TypeLimitHigh5-Eigenschaft (Seite 644)
TypeLimitHigh4-Eigenschaft (Seite 644)
TypeAlarmLow-Eigenschaft (Seite 644)
TypeAlarmHigh-Eigenschaft (Seite 644)
Type-Eigenschaft (Seite 642)
TrendColor-Eigenschaft (vor WinCC V7) (Seite 623)
Trend-Eigenschaft (Seite 620)
Top-Eigenschaft (Seite 618)
ToleranceLow-Eigenschaft (Seite 605)
ToleranceHigh-Eigenschaft (Seite 604)
ScalingType-Eigenschaft (Seite 536)
Scaling-Eigenschaft (Seite 535)
ScaleTicks-Eigenschaft (Seite 535)
ScaleColor-Eigenschaft (Seite 535)
RightComma-Eigenschaft (Seite 528)
Process-Eigenschaft (Seite 521)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Min-Eigenschaft (Seite 484)

Bildfenster

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Bildfenster".

Typkennzeichnung in VBS

HMIScreenWindow

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "ScreenWindow1" um 10 Pixel nach rechts verschoben:

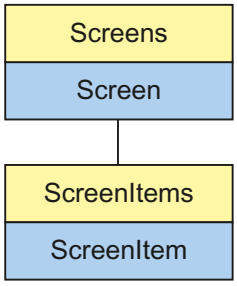
```
'VBS35
Dim objScrWindow
Set objScrWindow = ScreenItems("ScreenWindow1")
objScrWindow.Left = objScrWindow.Left + 10
```

Siehe auch

ServerPrefix-Eigenschaft (Seite 548)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Zoom-Eigenschaft (Seite 684)
WindowBorder-Eigenschaft (Seite 674)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
UpdateCycle-Eigenschaft (Seite 649)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
TagPrefix-Eigenschaft (Seite 572)
ScrollPositionY-Eigenschaft (Seite 539)
ScrollPositionX-Eigenschaft (Seite 539)
ScrollBars-Eigenschaft (Seite 539)
ScreenName-Eigenschaft (Seite 537)
Screens-Eigenschaft (Seite 538)
Parent-Eigenschaft (Seite 506)
OnTop-Eigenschaft (Seite 494)
OffsetTop-Eigenschaft (Seite 493)
OffsetLeft-Eigenschaft (Seite 493)
ObjectName-Eigenschaft (Seite 490)
Moveable-Eigenschaft (Seite 485)
MaximizeButton-Eigenschaft (Seite 467)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
Enabled-Eigenschaft (Seite 386)
CloseButton-Eigenschaft (Seite 350)
CaptionText-Eigenschaft (Seite 344)
Caption-Eigenschaft (Seite 342)
AdaptSize-Eigenschaft (Seite 301)
AdaptPicture-Eigenschaft (Seite 301)
MenuToolBarConfig-Eigenschaft (Seite 473)

Control

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Control".

Der Objekt-Typ Control nimmt immer die Eigenschaften des von Ihnen ausgewählten Control-Typs an. Bei von WinCC bereitgestellten Controls finden Sie die Eigenschaften unter der Beschreibung der entsprechenden Controls.

Bei Controls von Fremdanbietern werden die Eigenschaften von den Controls mitgebracht und sind damit nicht Bestandteil dieser Beschreibung. Sie können die Control-Eigenschaften jedoch über die Eigenschaft "Item" abfragen.

Typkennzeichnung in VBS

Spezielle WinCC-Typbezeichnung oder versionsunabhängige ProgID

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Control1" um 10 Pixel nach rechts verschoben:

```
'VBS36
Dim objControl
Set objControl = ScreenItems("Control1")
objControl.Left = objControl.Left + 10
```

Besonderheit

Bei von WinCC bereitgestellten Controls wird eine spezielle Kennung als Typ zurückgeliefert. Sie ist unter der Rubrik "Typkennzeichnung in VBS" bei den Einzelbeschreibungen der WinCC-Controls zu finden.

Verwenden von Fremdanbieter-Controls

Bei WinCC-fremden Controls wird die versionsunabhängige ProgID als Typ zurückgeliefert.

Aus der ProgID können Sie die versionsabhängige ProgID oder den "User friendly Name" ermitteln: In folgendem Beispiel ist "Control1" ein im Bild eingebettetes Control, das über die Type-Eigenschaft bereits die versionsunabhängige ProgID zurückgibt.

Hinweis

Da nicht jedes Control eine versionsabhängige ProgID besitzt, sollte zum Abfragen der versionsabhängigen ProgID oder des UserFriendlyName eine Fehlerbehandlung eingebaut werden. Wird keine Fehlerbehandlung verwendet, wird der Code sofort ohne Ergebnis beendet, wenn keine ProgID gefunden wird.

Ermitteln Sie die versionsabhängige ProgID wie folgt:

```
'VBS37
Dim objControl
Dim strCurrentVersion
Set objControl = ScreenItems("Control1")
strCurrentVersion = CreateObject("WScript.Shell").RegRead("HKCR\" & objControl.Type &
"\CurVer\")
MsgBox strCurrentVersion
```

Hinweis

Damit obiges Beispiel funktioniert, sollte ein Multimediacontrol ins Bild eingefügt werden.

Ermitteln Sie den UserFriendlyName wie folgt:

```
'VBS38
Dim objControl
Dim strFriendlyName
Set objControl = ScreenItems("Control1")
strFriendlyName = CreateObject("WScript.Shell").RegRead("HKCR\" & objControl.Type & "\")
MsgBox strFriendlyName
```

Hinweis

Damit obiges Beispiel funktioniert, sollte ein Multimediacontrol ins Bild eingefügt werden.

Wenn Sie ein nicht von WinCC bereitgestelltes Control verwenden, kann es vorkommen, dass die vom Control mitgebrachten Eigenschaften namensgleich sind mit den allgemeinen ScreenItem-Eigenschaften. In diesem Fall haben die ScreenItem-Eigenschaften Vorrang. Auf die "verdeckten" Eigenschaften eines Fremdanbieter-Controls kann über die zusätzliche Eigenschaft "object" zugegriffen werden. Sprechen Sie die Eigenschaften eines Fremdanbieter-Controls also z.B. in folgender Form an:

```
Control.object.type
```

Bei Namensgleicheit werden die Eigenschaften des ScreenItem-Objektes verwendet., wenn Sie die folgende Form verwenden:

```
Control.type
```

Verfügbare WinCC-Controls

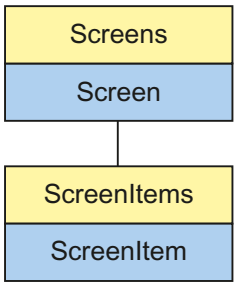
- WinCC Alarm Control
- WinCC Digital/Analog Clock
- WinCC FunctionTrendControl
- WinCC Gauge Control
- WinCC Media Control
- WinCC OnlineTableControl
- WinCC OnlineTrendControl
- WinCC Push Button Control
- WinCC Slider Control
- WinCC UserArchiveControl
- HMI Symbol Library

Siehe auch

Object-Eigenschaft (Seite 489)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Eigenschaft (Seite 437)
Height-Eigenschaft (Seite 421)
Enabled-Eigenschaft (Seite 386)

EA-Feld

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "EA-Feld".

Typkennzeichnung in VBS

HMIIOField

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "IOField1" um 10 Pixel nach rechts verschoben:

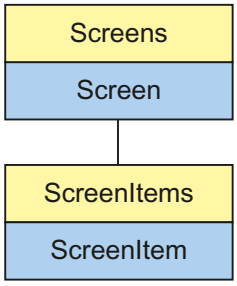
```
'VBS39
Dim objIOField
Set objIOField = ScreenItems("IOField1")
objIOField.Left = objIOField.Left + 10
```

Siehe auch

OperationMessage-Eigenschaft (Seite 495)
EditAtOnce-Eigenschaft (Seite 386)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
OutputValue-Eigenschaft (Seite 505)
OutputFormat-Eigenschaft (Seite 505)
Orientation-Eigenschaft (Seite 504)
OperationReport-Eigenschaft (Seite 503)
ObjectName-Eigenschaft (Seite 490)
LimitMin-Eigenschaft (Seite 457)
LimitMax-Eigenschaft (Seite 457)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
HiddenInput-Eigenschaft (Seite 421)
Height-Eigenschaft (Seite 421)
ForeFlashColorOn-Eigenschaft (Seite 413)
ForeFlashColorOff-Eigenschaft (Seite 413)
ForeColor-Eigenschaft (Seite 412)
FontUnderline-Eigenschaft (Seite 411)
FontSize-Eigenschaft (Seite 411)
FontName-Eigenschaft (Seite 410)
FontItalic-Eigenschaft (Seite 409)
FontBold-Eigenschaft (Seite 409)
FlashRateForeColor-Eigenschaft (Seite 405)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashForeColor-Eigenschaft (Seite 402)
FlashBorderColor-Eigenschaft (Seite 402)
FlashBackColor-Eigenschaft (Seite 401)
Fillstyle-Eigenschaft (Seite 398)
FillColor-Eigenschaft (Seite 396)

Faceplate-Instanz

Beschreibung



Objekt-Typ des ScreenItem-Objekts. Repräsentiert das Grafikobjekt "Faceplate-Instanz".

Typkennzeichnung in VBS

HMIFaceplateObject

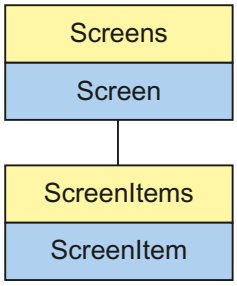
Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "FaceplateInstanz1" um 10 Pixel nach rechts verschoben:

```
'VBS309
Dim objFaceplateObject
Set objFaceplateObject = ScreenItems("FaceplateInstanz1")
objFaceplateObject.Left = objFaceplateObject.Left + 10
```

Grafik-Objekt

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Grafik-Objekt".

Typkennzeichnung in VBS

HMIGraphicView

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "GraphicObject1" um 10 Pixel nach rechts verschoben:

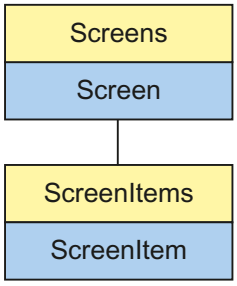
```
'VBS40
Dim objGraphicView
Set objGraphicView= ScreenItems("GraphicObject1")
objGraphicView.Left = objGraphicView.Left + 10
```

Siehe auch

Parent-Eigenschaft (Seite 506)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
PicUseTransColor-Eigenschaft (Seite 518)
PictureName-Eigenschaft (Seite 515)
PicTransColor-Eigenschaft (Seite 514)
PicReferenced-Eigenschaft (Seite 513)
PasswordLevel-Eigenschaft (Seite 508)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashBorderColor-Eigenschaft (Seite 402)
FlashBackColor-Eigenschaft (Seite 401)
Fillstyle-Eigenschaft (Seite 398)
FillingIndex-Eigenschaft (Seite 397)
Filling-Eigenschaft (Seite 397)
FillColor-Eigenschaft (Seite 396)
Enabled-Eigenschaft (Seite 386)
BorderWidth-Eigenschaft (Seite 337)
BorderStyle-Eigenschaft (Seite 337)
BorderFlashColorOn-Eigenschaft (Seite 337)
BorderFlashColorOff-Eigenschaft (Seite 336)
BorderColor-Eigenschaft (Seite 335)
BorderBackColor-Eigenschaft (Seite 335)
BackFlashColorOn-Eigenschaft (Seite 319)
BackFlashColorOff-Eigenschaft (Seite 318)
BackColor-Eigenschaft (Seite 316)

Kombinationsfeld

Beschreibung



Objekt-Typ des ScreenItem-Objekts. Repräsentiert das Grafikobjekt "Kombinationsfeld".

Typkennzeichnung in VBS

HMIComboBox

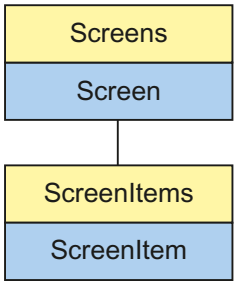
Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "ComboBox1" um 10 Pixel nach rechts verschoben:

```
'VBS21
Dim objComboBox
Set objComboBox = ScreenItems("ComboBox1")
objComboBox.Left = objComboBox.Left + 10
```

Listenfeld

Beschreibung



Objekt-Typ des ScreenItem-Objekts. Repräsentiert das Grafikobjekt "Listenfeld".

Typkennzeichnung in VBS

HMIListBox

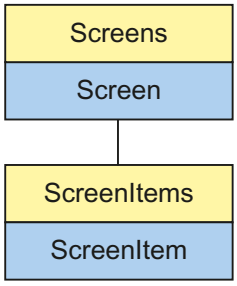
Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "ListBox1" um 10 Pixel nach rechts verschoben:

```
'VBS21
Dim objListBox
Set objListBox = ScreenItems("ListBox1")
objListBox.Left = objListBox.Left + 10
```

Mehrzeiliger Text

Beschreibung



Objekt-Typ des ScreenItem-Objekts. Repräsentiert das Grafikobjekt "Mehrzeiliger Text".

Typkennzeichnung in VBS

HMIMultiLineEdit

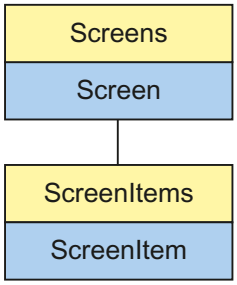
Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "MultiLineEdit1" um 10 Pixel nach rechts verschoben:

```
'VBS21
Dim objMultiLineEdit
Set objMultiLineEdit = ScreenItems("MultiLineEdit1")
objMultiLineEdit.Left = objMultiLineEdit.Left + 10
```

OLE-Objekt

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "OLE-Objekt". Der Rückgabewert ist vom Typ STRING.

Typkennzeichnung in VBS

Versionsunabhängige ProgID

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "OLEElement1" um 10 Pixel nach rechts verschoben:

```
'VBS41
Dim objOLEElement
Set objOLEElement = ScreenItems("OLEElement1")
objOLEElement.Left = objOLEElement.Left + 10
```

Besonderheit

Bei OLE-Objekten wird die versionsunabhängige ProgID als Typ zurückgeliefert.

Aus der ProgID können Sie die versionsabhängige ProgID oder den "User friendly Name" ermitteln: In folgendem Beispiel ist "OLEObjekt1" ein im Bild eingebettetes Control, das über die Type-Eigenschaft bereits die versionsunabhängige ProgID zurückgibt.

Hinweis

Da nicht jedes Control eine versionsabhängige ProgID besitzt, sollte zum Abfragen der versionsabhängigen ProgID oder des UserFriendlyName eine Fehlerbehandlung eingebaut werden. Wird keine Fehlerbehandlung verwendet, wird der Code sofort ohne Ergebnis beendet, wenn keine ProgID gefunden wird.

Ermitteln Sie die versionsabhängige ProgID wie folgt:

```
'VBS42
Dim objControl
Dim strCurrentVersion
Set objControl = ScreenItems("OLEElement1")
strCurrentVersion = CreateObject("WScript.Shell").RegRead("HKCR\" & objControl.Type &
"\CurVer\")
MsgBox strCurrentVersion
```

Hinweis

Damit obiges Beispiel funktioniert, sollte ein Word-Dokument als OLE-Objekt ins Bild eingebettet werden.

Ermitteln Sie den User friendly Name wie folgt:

```
'VBS43
Dim objControl
Dim strFriendlyName
Set objControl = ScreenItems("OLEElement1")
strFriendlyName = CreateObject("WScript.Shell").RegRead("HKCR\" & objControl.Type & "\")
MsgBox strFriendlyName
```

Hinweis

Damit obiges Beispiel funktioniert, sollte ein Word-Dokument als OLE-Objekt ins Bild eingebettet werden.

Verwenden von OLE-Objekten

Wenn Sie ein OLE-Objekt verwenden, kann es vorkommen, dass die vom OLE-Objekt mitgebrachten Eigenschaften namensgleich sind mit den allgemeinen ScreenItem-Eigenschaften. In diesem Fall haben die ScreenItem-Eigenschaften vorrang. Auf die "verdeckten" Eigenschaften eines OLE-Objektes kann über die zusätzliche Eigenschaft "object" zugegriffen werden. Sprechen Sie die Eigenschaften eines OLE-Objektes also z.B. in folgender Form an:

```
OLEObjekt.object.type
```

Verwenden Sie nur die Form

```
OLEObjekt.type
```

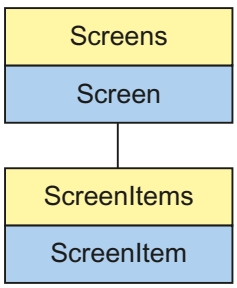
werden bei Namensgleicheit die Eigenschaften des ScreenItem-Objektes verwendet.

Siehe auch

Height-Eigenschaft (Seite 421)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
Parent-Eigenschaft (Seite 506)
Object-Eigenschaft (Seite 489)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Eigenschaft (Seite 437)
Enabled-Eigenschaft (Seite 386)

Sammelanzeige

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Sammelanzeige".

Typkennzeichnung in VBS

HMIGroupDisplay

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "GroupDisplay1" um 10 Pixel nach rechts verschoben:

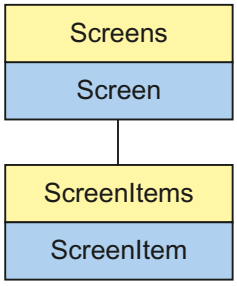
```
'VBS44
Dim objGroupDisplay
Set objGroupDisplay = ScreenItems("GroupDisplay1")
objGroupDisplay.Left = objGroupDisplay.Left + 10
```

Siehe auch

Activate-Methode (Seite 686)
MCKQBackColorOn-Eigenschaft (Seite 471)
FontBold-Eigenschaft (Seite 409)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
UserValue4-Eigenschaft (Seite 653)
UserValue3-Eigenschaft (Seite 653)
UserValue2-Eigenschaft (Seite 653)
UserValue1-Eigenschaft (Seite 652)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
SignificantMask-Eigenschaft (Seite 556)
SameSize-Eigenschaft (Seite 534)
Relevant-Eigenschaft (Seite 527)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
MessageClass-Eigenschaft (Seite 481)
MCText-Eigenschaft (Seite 472)
MCKQTextFlash-Eigenschaft (Seite 472)
MCKQTextColorOn-Eigenschaft (Seite 471)
MCKQTextColorOff-Eigenschaft (Seite 471)
MCKQBackFlash-Eigenschaft (Seite 471)
MCKQBackColorOff-Eigenschaft (Seite 470)
MCKOTextFlash-Eigenschaft (Seite 470)
MCKOTextColorOn-Eigenschaft (Seite 470)
MCKOTextColorOff-Eigenschaft (Seite 469)
MCKOBackFlash-Eigenschaft (Seite 469)
MCKOBackColorOn-Eigenschaft (Seite 469)
MCKOBackColorOff-Eigenschaft (Seite 469)
MCGUTextFlash-Eigenschaft (Seite 468)
MCGUTextColorOn-Eigenschaft (Seite 468)
MCGUTextColorOff-Eigenschaft (Seite 468)
MCGUBackFlash-Eigenschaft (Seite 467)
MCGUBackColorOn-Eigenschaft (Seite 467)
MCGUBackColorOff-Eigenschaft (Seite 467)

Textliste

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Textliste".

Typkennzeichnung in VBS

HMISymbolicIOField

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "TextList1" um 10 Pixel nach rechts verschoben:

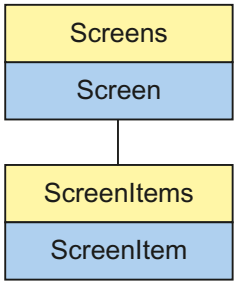
```
'VBS45
Dim objSymIO
Set objSymIO = ScreenItems("TextList1")
objSymIO.Left = objSymIO.Left + 10
```

Siehe auch

Type-Eigenschaft (Seite 642)
FontUnderline-Eigenschaft (Seite 411)
BackFlashColorOff-Eigenschaft (Seite 318)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
UnselTextColor-Eigenschaft (Seite 648)
UnselBGColor-Eigenschaft (Seite 648)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
SelTextColor-Eigenschaft (Seite 546)
SelBGColor-Eigenschaft (Seite 541)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
OutputValue-Eigenschaft (Seite 505)
Orientation-Eigenschaft (Seite 504)
OperationReport-Eigenschaft (Seite 503)
OperationMessage-Eigenschaft (Seite 495)
ObjectName-Eigenschaft (Seite 490)
NumberLines-Eigenschaft (Seite 488)
ListType-Eigenschaft (Seite 459)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
LanguageSwitch-Eigenschaft (Seite 434)
ItemBorderWidth-Eigenschaft (Seite 432)
ItemBorderStyle-Eigenschaft (Seite 432)
ItemBorderColor-Eigenschaft (Seite 431)
ItemBorderBackColor-Eigenschaft (Seite 431)
Height-Eigenschaft (Seite 421)
ForeFlashColorOn-Eigenschaft (Seite 413)
ForeFlashColorOff-Eigenschaft (Seite 413)
ForeColor-Eigenschaft (Seite 412)
FontSize-Eigenschaft (Seite 411)
FontName-Eigenschaft (Seite 410)
FontItalic-Eigenschaft (Seite 409)
FontBold-Eigenschaft (Seite 409)
FlashRateForeColor-Eigenschaft (Seite 405)

Zustandsanzeige

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Zustandsanzeige".

Typkennzeichnung in VBS

HMIGraphicIOField

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "StatusDisplay1" um 10 Pixel nach rechts verschoben:

```
'VBS46
Dim objGraphicIO
Set objGraphicIO= ScreenItems("StatusDisplay1")
objGraphicIO.Left = objGraphicIO.Left + 10
```

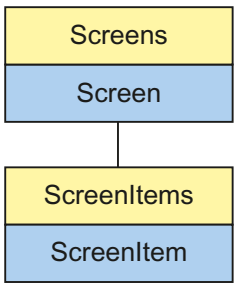
Siehe auch

Layer-Objekt (Seite 129)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Index-Eigenschaft (Seite 428)
Height-Eigenschaft (Seite 421)
FlashRateFlashPic-Eigenschaft (Seite 405)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashPicUseTransColor-Eigenschaft (Seite 404)
FlashPicture-Eigenschaft (Seite 403)
FlashPicTransColor-Eigenschaft (Seite 403)
FlashPicReferenced-Eigenschaft (Seite 403)
FlashFlashPicture-Eigenschaft (Seite 402)
FlashBorderColor-Eigenschaft (Seite 402)
Enabled-Eigenschaft (Seite 386)
BorderWidth-Eigenschaft (Seite 337)
BorderStyle-Eigenschaft (Seite 337)
BorderFlashColorOn-Eigenschaft (Seite 337)
BorderFlashColorOff-Eigenschaft (Seite 336)
BorderColor-Eigenschaft (Seite 335)
BorderBackColor-Eigenschaft (Seite 335)
BasePicUseTransColor-Eigenschaft (Seite 323)
BasePicture-Eigenschaft (Seite 323)
BasePicTransColor-Eigenschaft (Seite 322)
BasePicReferenced-Eigenschaft (Seite 322)

1.14.3.4 Windows-Objekte

Button

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Button".

Typkennzeichnung in VBS

HMIButton

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Button1" um 10 Pixel nach rechts verschoben:

```
'VBS47
Dim cmdButton
Set cmdButton = ScreenItems("Button1")
cmdButton.Left = cmdButton.Left + 10
```

Hinweise zur Fehlerbehandlung

Buttons und PushButtons werden im Objektmodell auf einen Typ "HMIButton" abgebildet. Da die Objekte dennoch teilweise unterschiedliche Eigenschaften aufweisen, sollten Sie das Vorhandensein der Eigenschaften (dynamische Typ-Ermittlung in Runtime) über eine Ausnahmebehandlung abfragen. Die Ausnahmebehandlung wird durch folgende Anweisung für die entsprechende Prozedur aktiviert:

```
On Error Resume Next
```

Die Anweisung führt dazu, dass die VBScript-Engine nach einem Laufzeitfehler die Folgeanweisung ausführt.

In der Folgezeile kann dann der Fehlercode über das Err-Objekt überprüft werden. Um die Behandlung von Laufzeitfehlern im Skript wieder abzuschalten, wird folgende Anweisung verwendet:

```
On Error Goto 0
```

Die Fehlerbehandlung ist immer auf die Prozedurebene bezogen. Löst ein Skript in einer Prozedur einen Fehler aus, prüft VBScript, ob auf dieser Ebene eine Fehlerbehandlung implementiert ist. Ist dies nicht der Fall, geht die Kontrolle eine Ebene höher (zur rufenden Prozedur). Gibt es dort ebenfalls keine Fehlerbehandlung, wird die Kontrolle wiederum eine Ebene höher übergeben. Dies setzt sich solange fort, bis entweder die oberste Modulebene erreicht ist oder der Code zur Laufzeitfehlerbehandlung gefunden wird. Fehlt die Aktivierung der Runtime-Fehlerbehandlung, wird die Kontrolle auf der Hauptebene an die interne VBScript-Laufzeitfehlerbehandlung übergeben. Diese zeigt dann einen Fehlerdialog an und bricht das Skript ab.

Die "On Error Resume Next"-Anweisung kann auf jeder Ebene (also auch in Prozeduren) eingebaut werden. Mit dem Einsatz der Fehlerbehandlung ist prinzipiell feststellbar, ob der Anwender den gewünschten Implementierungstyp auch wirklich verwendet.

Außerdem kann garantiert werden, dass es nicht zu einem Abbruch der Ausführung aufgrund eines fehlerhaften Zugriffs auf das Objekt kommt.

Beispiel zur Fehlerbehandlung

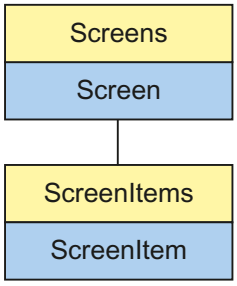
```
Sub OnClick(ByVal Item)
'VBS48
Dim objScreenItem
On Error Resume Next    'Activation of errorhandling
For Each objScreenItem In ScreenItems
If objScreenItem.Type = "HMIButton" Then
'
'=== Property "Text" available only for Standard-Button
objScreenItem.Text = "Windows"
If 0 <> Err.Number Then
HMIRuntime.Trace objScreenItem.ObjectName & ": no Windows-Button" & vbCrLf
Err.Clear    'Delete error message
End If
End If
Next
On Error Goto 0    'Deactivation of errorhandling
End Sub
```

Siehe auch

Top-Eigenschaft (Seite 618)
FlashBorderColor-Eigenschaft (Seite 402)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
WindowsStyle-Eigenschaft (Seite 675)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
ToolTipText-Eigenschaft (Seite 617)
Textliste (Seite 208)
PictureUp-Eigenschaft (Seite 516)
PictureDown-Eigenschaft (Seite 515)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
Orientation-Eigenschaft (Seite 504)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Hotkey-Eigenschaft (Seite 426)
Height-Eigenschaft (Seite 421)
ForeFlashColorOn-Eigenschaft (Seite 413)
ForeFlashColorOff-Eigenschaft (Seite 413)
ForeColor-Eigenschaft (Seite 412)
FontUnderline-Eigenschaft (Seite 411)
FontSize-Eigenschaft (Seite 411)
FontName-Eigenschaft (Seite 410)
FontItalic-Eigenschaft (Seite 409)
FontBold-Eigenschaft (Seite 409)
FlashRateForeColor-Eigenschaft (Seite 405)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashForeColor-Eigenschaft (Seite 402)
FlashBackColor-Eigenschaft (Seite 401)
Fillstyle-Eigenschaft (Seite 398)
FillingIndex-Eigenschaft (Seite 397)
Filling-Eigenschaft (Seite 397)
FillColor-Eigenschaft (Seite 396)
Enabled-Eigenschaft (Seite 386)

Check-Box

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Check-Box".

Typkennzeichnung in VBS

HMICheckBox

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "CheckBox1" um 10 Pixel nach rechts verschoben:

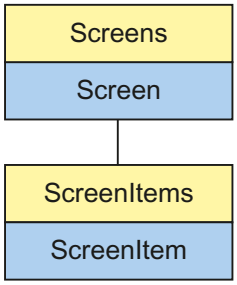
```
'VBS49
Dim chkCheckBox
Set chkCheckBox = ScreenItems("CheckBox1")
chkCheckBox.Left = chkCheckBox.Left + 10
```

Siehe auch

FontSize-Eigenschaft (Seite 411)
BackColor-Eigenschaft (Seite 316)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
Textliste (Seite 208)
Process-Eigenschaft (Seite 521)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
Orientation-Eigenschaft (Seite 504)
OperationMessage-Eigenschaft (Seite 495)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Index-Eigenschaft (Seite 428)
Height-Eigenschaft (Seite 421)
ForeFlashColorOn-Eigenschaft (Seite 413)
ForeFlashColorOff-Eigenschaft (Seite 413)
ForeColor-Eigenschaft (Seite 412)
FontUnderline-Eigenschaft (Seite 411)
FontName-Eigenschaft (Seite 410)
FontItalic-Eigenschaft (Seite 409)
FontBold-Eigenschaft (Seite 409)
FlashRateForeColor-Eigenschaft (Seite 405)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashForeColor-Eigenschaft (Seite 402)
FlashBorderColor-Eigenschaft (Seite 402)
FlashBackColor-Eigenschaft (Seite 401)
Fillstyle-Eigenschaft (Seite 398)
FillingIndex-Eigenschaft (Seite 397)
Filling-Eigenschaft (Seite 397)
FillColor-Eigenschaft (Seite 396)
Enabled-Eigenschaft (Seite 386)

Radio-Box

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Radio-Box".

Typkennzeichnung in VBS

HMIOptionGroup

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "RadioBox1" um 10 Pixel nach rechts verschoben:

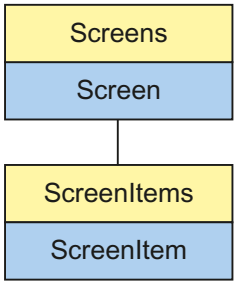
```
'VBS50
Dim objOptionGroup
Set objOptionGroup = ScreenItems("RadioBox1")
objOptionGroup.Left = objOptionGroup.Left + 10
```

Siehe auch

ForeColor-Eigenschaft (Seite 412)
BackFlashColorOn-Eigenschaft (Seite 319)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
Textliste (Seite 208)
Process-Eigenschaft (Seite 521)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
Orientation-Eigenschaft (Seite 504)
OperationMessage-Eigenschaft (Seite 495)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Index-Eigenschaft (Seite 428)
Height-Eigenschaft (Seite 421)
ForeFlashColorOn-Eigenschaft (Seite 413)
ForeFlashColorOff-Eigenschaft (Seite 413)
FontUnderline-Eigenschaft (Seite 411)
FontSize-Eigenschaft (Seite 411)
FontName-Eigenschaft (Seite 410)
FontItalic-Eigenschaft (Seite 409)
FontBold-Eigenschaft (Seite 409)
FlashRateForeColor-Eigenschaft (Seite 405)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashForeColor-Eigenschaft (Seite 402)
FlashBorderColor-Eigenschaft (Seite 402)
FlashBackColor-Eigenschaft (Seite 401)
Fillstyle-Eigenschaft (Seite 398)
FillingIndex-Eigenschaft (Seite 397)
Filling-Eigenschaft (Seite 397)
FillColor-Eigenschaft (Seite 396)
Enabled-Eigenschaft (Seite 386)

Rundbutton

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Rundbutton".

Typkennzeichnung in VBS

HMISwitch

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "RoundButton1" um 10 Pixel nach rechts verschoben:

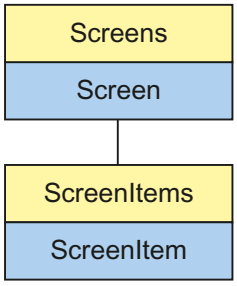
```
'VBS51
Dim objSwitch
Set objSwitch= ScreenItems("RoundButton1")
objSwitch.Left = objSwitch.Left + 10
```

Siehe auch

PicDownUseTransColor-Eigenschaft (Seite 513)
BorderColorTop-Eigenschaft (Seite 336)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
Toggle-Eigenschaft (Seite 604)
Radius-Eigenschaft (Seite 524)
Pressed-Eigenschaft (Seite 520)
PicUpUseTransColor-Eigenschaft (Seite 517)
PicUpTransparent-Eigenschaft (Seite 517)
PicUpReferenced-Eigenschaft (Seite 517)
PictureUp-Eigenschaft (Seite 516)
PictureDown-Eigenschaft (Seite 515)
PictureDeactivated-Eigenschaft (Seite 514)
PicDownTransparent-Eigenschaft (Seite 513)
PicDownReferenced-Eigenschaft (Seite 512)
PicDeactUseTransColor-Eigenschaft (Seite 512)
PicDeactTransparent-Eigenschaft (Seite 512)
PicDeactReferenced-Eigenschaft (Seite 511)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashBorderColor-Eigenschaft (Seite 402)
FlashBackColor-Eigenschaft (Seite 401)
Fillstyle-Eigenschaft (Seite 398)
FillingIndex-Eigenschaft (Seite 397)
Filling-Eigenschaft (Seite 397)
FillColor-Eigenschaft (Seite 396)
Enabled-Eigenschaft (Seite 386)

Slider

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Slider".

Typkennzeichnung in VBS

HMISlider

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Slider1" um 10 Pixel nach rechts verschoben:

```
'VBS53
Dim sldSlider
Set sldSlider = ScreenItems("Slider1")
sldSlider.Left = sldSlider.Left + 10
```

Hinweise zur Fehlerbehandlung

Slider und WinCC Slider Controls werden im Objektmodell auf einen Typ "HMISlider" abgebildet. Da die Objekte dennoch teilweise unterschiedliche Eigenschaften aufweisen, sollten Sie das Vorhandensein der Eigenschaften (dynamische Typ-Ermittlung in Runtime) über eine Ausnahmebehandlung abfragen. Die Ausnahmebehandlung wird durch folgende Anweisung für die entsprechende Prozedur aktiviert:

```
On Error Resume Next
```

Die Anweisung führt dazu, dass die VBScript-Engine nach einem Laufzeitfehler die Folgeanweisung ausführt.

In der Folgezeile kann dann der Fehlercode über das Err-Objekt überprüft werden. Um die Behandlung von Laufzeitfehlern im Skript wieder abzuschalten, wird folgende Anweisung verwendet:

```
On Error Goto 0
```

Die Fehlerbehandlung ist immer auf die Prozedurebene bezogen. Löst ein Skript in einer Prozedur einen Fehler aus, prüft VBScript, ob auf dieser Ebene eine Fehlerbehandlung implementiert ist. Ist dies nicht der Fall, geht die Kontrolle eine Ebene höher (zur rufenden Prozedur). Gibt es dort ebenfalls keine Fehlerbehandlung, wird die Kontrolle wiederum eine Ebene höher übergeben. Dies setzt sich solange fort, bis entweder die oberste Modulebene erreicht ist oder der Code zur Laufzeitfehlerbehandlung gefunden wird. Fehlt die Aktivierung der Runtime-Fehlerbehandlung, wird die Kontrolle auf der Hauptebene an die interne VBScript-Laufzeitfehlerbehandlung übergeben. Diese zeigt dann einen Fehlerdialog an und bricht das Skript ab.

Die "On Error Resume Next"-Anweisung kann auf jeder Ebene (also auch in Prozeduren) eingebaut werden. Mit dem Einsatz der Fehlerbehandlung ist prinzipiell feststellbar, ob der Anwender den gewünschten Implementierungstyp auch wirklich verwendet.

Außerdem kann garantiert werden, dass es nicht zu einem Abbruch der Ausführung aufgrund eines fehlerhaften Zugriffs auf das Objekt kommt.

Beispiel zur Fehlerbehandlung

```
Sub OnClick(Byval Item)
'VBS194
Dim ScreenItem
' activating error handling:
On Error Resume Next
For Each ScreenItem In ScreenItems
If ScreenItem.Type = "HMISlider" Then
'=== Property "BevelColorUp" only exists for a WinCC Slider Control
ScreenItem.BevelColorUp = 1
If (Err.Number <> 0) Then
HMIRuntime.Trace(ScreenItem.ObjectName + ": no Windows-Slider" + vbCrlf)
' delete error message
Err.Clear
End If
'=== Property "BorderStyle" only exists for a Windows-Slider
ScreenItem.BorderStyle = 1
If (Err.Number <> 0) Then
HMIRuntime.Trace(ScreenItem.ObjectName + ": no WinCC Slider Control" + vbCrlf)
Err.Clear
End If
End If
Next
On Error GoTo 0  ' deactivating error handling
End Sub
```

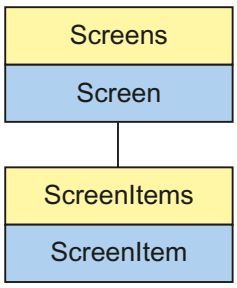
Siehe auch

Height-Eigenschaft (Seite 421)
BackColorBottom-Eigenschaft (Seite 318)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
WindowsStyle-Eigenschaft (Seite 675)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
SmallChange-Eigenschaft (Seite 557)
Process-Eigenschaft (Seite 521)
PasswordLevel-Eigenschaft (Seite 508)
Parent-Eigenschaft (Seite 506)
OperationReport-Eigenschaft (Seite 503)
OperationMessage-Eigenschaft (Seite 495)
ObjectName-Eigenschaft (Seite 490)
Min-Eigenschaft (Seite 484)
Max-Eigenschaft (Seite 466)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
FlashRateBorderColor-Eigenschaft (Seite 405)
FlashRateBackColor-Eigenschaft (Seite 404)
FlashBorderColor-Eigenschaft (Seite 402)
FlashBackColor-Eigenschaft (Seite 401)
Fillstyle-Eigenschaft (Seite 398)
FillingIndex-Eigenschaft (Seite 397)
Filling-Eigenschaft (Seite 397)
FillColor-Eigenschaft (Seite 396)
ExtendedOperation-Eigenschaft (Seite 394)
Enabled-Eigenschaft (Seite 386)
Direction-Eigenschaft (Seite 383)
ColorTop-Eigenschaft (Seite 357)
ColorBottom-Eigenschaft (Seite 354)
BlinkMode-Eigenschaft (Seite 339)
BorderWidth-Eigenschaft (Seite 337)
BorderStyle-Eigenschaft (Seite 337)
BorderFlashColorOn-Eigenschaft (Seite 337)

1.14.3.5 Rohr-Objekte

Polygonrohr

Beschreibung



Objekt-Typ des ScreenItem-Objekts. Repräsentiert das Grafikobjekt "Polygonrohr".

Typkennzeichnung in VBS

HMITubePolyline

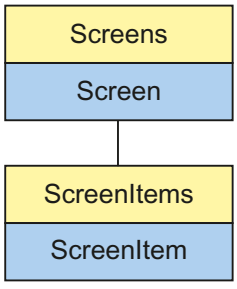
Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "TubePolyline1" um 10 Pixel nach rechts verschoben:

```
'VBS24
Dim objTubePolyline
Set objTubePolyline = ScreenItems("TubePolyline1")
objTubePolyline.Left = objTubePolyline.Left + 10
```

T-Stück

Beschreibung



Objekt-Typ des ScreenItem-Objekts. Repräsentiert das Grafikobjekt "T-Stück".

Typkennzeichnung in VBS

HMITubeTeeObject

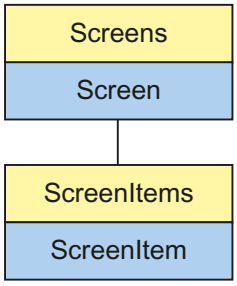
Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "TubeTeeObject1" um 10 Pixel nach rechts verschoben:

```
'VBS21
Dim objTubeTeeObject
Set objTubeTeeObject = ScreenItems("TubeTeeObject1")
objTubeTeeObject.Left = objTubeTeeObject.Left + 10
```

Doppel-T-Stück

Beschreibung



Objekt-Typ des ScreenItem-Objekts. Repräsentiert das Grafikobjekt "Doppel-T-Stück".

Typkennzeichnung in VBS

HMITubeDoubleTeeObject

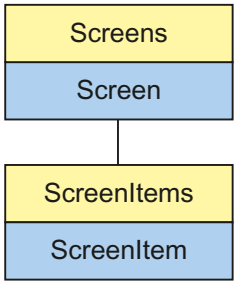
Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "TubeDoubleTeeObject1" um 10 Pixel nach rechts verschoben:

```
'VBS21
Dim objTubeDoubleTeeObject
Set objTubeDoubleTeeObject = ScreenItems("TubeDoubleTeeObject1")
objTubeDoubleTeeObject.Left = objTubeDoubleTeeObject.Left + 10
```

Rohrbogen

Beschreibung



Objekt-Typ des ScreenItem-Objekts. Repräsentiert das Grafikobjekt "Rohrbogen".

Typkennzeichnung in VBS

HMITubeArcObject

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "TubeArcObject1" um 10 Pixel nach rechts verschoben:

```
'VBS24
Dim objTubeArcObject
Set objTubeArcObject = ScreenItems("TubeArcObject1")
objTubeArcObject.Left = objTubeArcObject.Left + 10
```

1.14.3.6 Controls

Controls

Besonderheiten bei Controls

Bei WinCC-fremden Controls wird die versionsunabhängige ProgID als Typ zurückgeliefert.

Aus der ProgID können Sie die versionsabhängige ProgID oder den "User friendly Name" ermitteln: In folgendem Beispiel ist "Control1" ein im Bild eingebettetes Control, das über die Type-Eigenschaft bereits die versionsunabhängige ProgID zurückgibt.

Hinweis

Da nicht jedes Control eine versionsabhängige ProgID besitzt, sollte zum Abfragen der versionsabhängigen ProgID oder des UserFriendlyName eine Fehlerbehandlung eingebaut werden. Wird keine Fehlerbehandlung verwendet, wird der Code sofort ohne Ergebnis beendet, wenn keine ProgID gefunden wird.

Ermitteln Sie die versionsabhängige ProgID wie folgt:

```
'VBS153
Dim objControl
Dim strCurrentVersion
Set objControl = ScreenItems("Control1")
strCurrentVersion = CreateObject("WScript.Shell").RegRead("HKCR\" & objControl.Type &
"\CurVer\")
MsgBox strCurrentVersion
```

Hinweis

Damit obiges Beispiel funktioniert, sollte ein Multimediacontrol ins Bild eingefügt werden.

Ermitteln Sie den User friendly Name wie folgt:

```
'VBS154
Dim objControl
Dim strFriendlyName
Set objControl = ScreenItems("Control1")
strFriendlyName = CreateObject("WScript.Shell").RegRead("HKCR\" & objControl.Type & "\")
MsgBox strFriendlyName
```

Hinweis

Damit obiges Beispiel funktioniert, sollte ein Multimediacontrol ins Bild eingefügt werden.

Restriktionen von VBS für Dynamisierungen an Controls

Wenn Controls mit VBS dynamisiert werden sollen, dann müssen folgende Voraussetzungen erfüllt sein:

Methoden

Deklaration "ByRef" darf nur als "Variant" implementiert sein (ByRef xxx as Variant)

Deklaration "ByVal" darf nur mit Variablentypen implementiert sein (ByVal xxx as Long)

Eigenschaften

Deklaration "ByRef" darf nur als "Variant" implementiert sein (ByRef xxx as Variant)

Deklaration "ByVal" darf nur mit Variablentypen implementiert sein (ByVal xxx as Long)

Ereignisse

Deklaration "ByRef" ist nicht erlaubt.

Deklaration "ByVal" darf nur als "Variant" implementiert sein (ByVal xxx as Variant)

Arrays

Wenn Arrays verwendet werden, müssen diese mit (ByRef xxx As Variant) deklariert sein.

Damit Arrays in Variants übergeben werden können, muss eine Variant-Variable als zusätzliche Zwischenvariable dazwischen geschaltet werden nach folgendem Schema:

```
'VBS151
Dim arrayPoints(200)
Dim vArrayCoercion      'Variant for array Coercion
' Make the VBS Array compatibile with the OLE Automation
vArrayCoercion = (arrayPoints)
objTrendControl.DataXY = vArrayCoercion   ' this array will occur in the control
```

Verwenden von Fremdanbieter-Controls

Wenn Sie ein nicht von WinCC bereitgestelltes Control verwenden, kann es vorkommen, dass die vom Control mitgebrachten Eigenschaften namensgleich sind mit den allgemeinen ScreenItem-Eigenschaften. In diesem Fall haben die ScreenItem-Eigenschaften Vorrang. Auf die "verdeckten" Eigenschaften eines Fremdanbieter-Controls kann über die zusätzliche Eigenschaft "object" zugegriffen werden. Sprechen Sie die Eigenschaften eines Fremdanbieter-Controls also z.B. in folgender Form an:

```
Control.object.type
```

Wenn Sie die folgende Form verwenden, dann werden bei Namensgleichheit die Eigenschaften des ScreenItem-Objekts verwendet:

```
Control.type
```

Doppelte Parameter

Bei Verwendung eines nicht WinCC-eigenen Controls kann es ebenso vorkommen, dass in den Event-Prototypen ein Parameter mit dem Namen "Item" existiert. In diesem Fall wird der Name des Parameters in den vorgelegten VBS-Prototypen nach "ObjectItem" umbenannt. Falls auch dieser Name bereits existiert, wird der Name durch Anfügen von Ziffern unterschieden.

Verfügbare WinCC-Controls

- HMI Symbol Library
- WinCC AlarmControl
- WinCC Alarm Control (vor WinCC V7)
- WinCC Digital/Analog Clock
- WinCC FunctionTrendControl
- WinCC Function Trend Control (vor WinCC V7)
- WinCC Gauge Control
- WinCC Media Control
- WinCC OnlineTableControl
- WinCC Online Table Control (vor WinCC V7)
- WinCC OnlineTrendControl
- WinCC Online Trend Control (vor WinCC V7)
- WinCC Push Button Control
- WinCC RulerControl
- WinCC Slider Control
- WinCC UserArchiveControl

Siehe auch

HMI Symbol Library (Seite 248)
WinCC Slider Control (Seite 278)
WinCC Push Button Control (Seite 271)
WinCC Online Trend Control (vor WinCC V7) (Seite 291)
WinCC Online Table Control (vor WinCC V7) (Seite 289)
WinCC Gauge Control (Seite 261)
WinCC Function Trend Control (vor WinCC V7) (Seite 287)
WinCC Digital Analog Clock (Seite 255)
WinCC Alarm Control (vor WinCC V7) (Seite 285)
WinCC UserArchiveControl (Seite 282)
WinCC RulerControl (Seite 275)
WinCC OnlineTrendControl (Seite 267)
WinCC OnlineTableControl (Seite 263)
WinCC FunctionTrendControl (Seite 257)
WinCC AlarmControl (Seite 251)

Auflistungen der Controls

Column-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "Column" konfigurieren Sie die Eigenschaften der Spalten im WinCC UserArchiveControl.

Verwendung in den Controls

- WinCC UserArchiveControl (Seite 282)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "colobj.ColumnName" der Auflistungsname "Column" weggelassen: "colobj.Name".

ColumnAlias (Seite 358)	ColumnFlagUnique (Seite 361)	ColumnPosition (Seite 363)	ColumnSort (Seite 366)
ColumnAlign (Seite 359)	ColumnHideText (Seite 361)	ColumnPrecisions (Seite 363)	ColumnSortIndex (Seite 367)
ColumnAutoPrecisions (Seite 359)	ColumnHideTitleText (Seite 362)	ColumnReadAccess (Seite 364)	ColumnStartValue (Seite 367)
ColumnCaption (Seite 359)	ColumnIndex (Seite 362)	ColumnReadonly (Seite 364)	ColumnStringLength (Seite 367)
ColumnCount (Seite 359)	ColumnLeadingZeros (Seite 362)	ColumnRepos (Seite 364)	ColumnTimeFormat (Seite 367)
ColumnDateFormat (Seite 360)	ColumnLength (Seite 362)	ColumnShowDate (Seite 365)	ColumnType (Seite 368)
ColumnDMVarName (Seite 360)	ColumnMaxValue (Seite 363)	ColumnShowIcon (Seite 366)	ColumnVisible (Seite 368)
ColumnExponentialFormat (Seite 360)	ColumnMinValue (Seite 363)	ColumnShowTitleIcon (Seite 366)	ColumnWriteAccess (Seite 369)
ColumnFlagNotNull (Seite 361)	ColumnName (Seite 363)		

Siehe auch

GetColumn-Methode (Seite 695)

GetColumnCollection-Methode (Seite 696)

HitlistColumn-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "HitlistColumn" konfigurieren Sie die in der Hitliste verwendeten Meldeblöcke des WinCC AlarmControl.

Verwendung in den Controls

- WinCC AlarmControl (Seite 251)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "hitlistobj.HitlistColumnName" der Auflistungsname "HitlistColumn" weggelassen: "hitlistobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

HitlistColumnAdd (Seite 422)	HitlistColumnRepos (Seite 423)	HitListMaxSourceItems (Seite 424)
HitlistColumnCount (Seite 422)	HitlistColumnSort (Seite 423)	HitListMaxSourceItemsWarn (Seite 425)
HitlistColumnIndex (Seite 422)	HitlistColumnSortIndex (Seite 424)	HitListRelTime (Seite 425)
HitlistColumnName (Seite 423)	HitlistColumnVisible (Seite 424)	HitListRelTimeFactor (Seite 425)
HitlistColumnRemove (Seite 423)	HitListDefaultSort (Seite 424)	HitListRelTimeFactorType (Seite 425)

Siehe auch

GetHitlistColumn-Methode (Seite 697)

GetHistlistColumnCollection-Methode (Seite 698)

MessageBlock-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "MessageBlock" konfigurieren Sie die Meldeblöcke im WinCC AlarmControl.

Verwendung in den Controls

- WinCC AlarmControl (Seite 251)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von

"messageobj.MessageBlockName" der Auflistungsname "MessageBlock" weggelassen: "messageobj.Name".

MessageBlockAlign (Seite 473)	MessageBlockFlashOn (Seite 475)	MessageBlockLength (Seite 477)	MessageBlockShowIcon (Seite 479)
MessageBlockAutoPrecisions (Seite 473)	MessageBlockHideText (Seite 476)	MessageBlockName (Seite 478)	MessageBlockShowTitleIcon (Seite 479)
MessageBlockCaption (Seite 474)	MessageBlockHideTitleText (Seite 476)	MessageBlockPrecisions (Seite 478)	MessageBlockTextId (Seite 480)
MessageBlockCount (Seite 474)	MessageBlockID (Seite 476)	MessageBlockSelected (Seite 478)	MessageBlockTimeFormat (Seite 480)
MessageBlockDateFormat	MessageBlockIndex (Seite 477)	MessageBlockShowDate (Seite 479)	MessageBlockType (Seite 481)
MessageBlockExponentialFormat (Seite 475)	MessageBlockLeadingZeros (Seite 477)		

Siehe auch

GetMessageBlock-Methode (Seite 699)

GetMessageBlockCollection-Methode (Seite 700)

MessageColumn-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "MessageColumn" konfigurieren Sie die in den Meldelisten verwendeten Meldeblöcke des WinCC AlarmControl.

Verwendung in den Controls

- WinCC AlarmControl (Seite 251)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "messagecol.MessageColumnName" der Auflistungsname "MessageColumn" weggelassen: "messagecol.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

MessageColumnAdd (Seite 481)	MessageColumnName (Seite 482)	MessageColumnSort (Seite 483)
MessageColumnCount (Seite 481)	MessageColumnRemove (Seite 482)	MessageColumnSortIndex (Seite 483)
MessageColumnIndex (Seite 482)	MessageColumnRepos (Seite 482)	MessageColumnVisible (Seite 483)

Siehe auch

GetMessageColumn-Methode (Seite 701)

GetMessageColumnCollection-Methode (Seite 702)

OperatorMessage-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "OperatorMessage" konfigurieren Sie die Bedienmeldungen, die im WinCC AlarmControl angezeigt werden.

Verwendung in den Controls

- WinCC AlarmControl (Seite 251)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "opmessobj.OperatorMessageName" der Auflistungsname "OperatorMessage" weggelassen: "opmessobj.Name".

OperatorMessageID (Seite 495)	OperatorMessageSource5 (Seite 498)	OperatorMessageSourceType3 (Seite 501)
OperatorMessageIndex (Seite 495)	OperatorMessageSource6 (Seite 498)	OperatorMessageSourceType4 (Seite 501)
OperatorMessageName (Seite 496)	OperatorMessageSource7 (Seite 499)	OperatorMessageSourceType5 (Seite 501)
OperatorMessageNumber (Seite 496)	OperatorMessageSource8 (Seite 499)	OperatorMessageSourceType6 (Seite 502)
OperatorMessageSelected (Seite 497)	OperatorMessageSource9 (Seite 499)	OperatorMessageSourceType7 (Seite 502)
OperatorMessageSource1 (Seite 497)	OperatorMessageSource10 (Seite 500)	OperatorMessageSourceType8 (Seite 502)
OperatorMessageSource2 (Seite 497)	OperatorMessageSourceType1 (Seite 500)	OperatorMessageSourceType9 (Seite 503)
OperatorMessageSource3 (Seite 497)	OperatorMessageSourceType2 (Seite 500)	OperatorMessageSourceType10 (Seite 503)
OperatorMessageSource4 (Seite 498)		

Siehe auch

GetOperatorMessage-Methode (Seite 703)

GetOperatorMessageCollection-Methode (Seite 704)

Row-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "Row" greifen Sie auf die Zeilen der auf Tabellen basierenden Controls zu. Das Row-Objekt bezieht sich auf die Runtime-Daten in den Tabellen.

Verwendung in den Controls

WinCC AlarmControl (Seite 251)	WinCC OnlineTableControl (Seite 263)
WinCC RulerControl (Seite 275)	WinCC UserArchiveControl (Seite 282)

Verfügbare Methoden des Objekts

SelectAll (Seite 771)	SelectRow (Seite 772)
UnselectAll (Seite 786)	UnselectRow (Seite 787)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "rowobj.RowCellCount" der Auflistungsname "Row" weggelassen: "rowobj.CellCount".

RowCellCount (Seite 530)	RowCellText (Seite 530)
RowCount (Seite 531)	RowNumber (Seite 531)

Siehe auch

GetRow-Methode (Seite 705)

GetRowCollection-Methode (Seite 707)

GetSelectedRow-Methode (Seite 713)

GetSelectedRows-Methode (Seite 714)

RulerBlock-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "Block" konfigurieren Sie die Blöcke des WinCC RulerControl.

Verwendung in den Controls

- WinCC RulerControl (Seite 275)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "rulerblockobj.BlockName" der Auflistungsname "Block" weggelassen: "rulerblockobj.Name".

BlockAlign (Seite 329)	BlockHideText (Seite 331)	BlockPrecisions (Seite 333)
BlockAutoPrecisions (Seite 330)	BlockHideTitleText (Seite 331)	BlockShowDate (Seite 333)
BlockCaption (Seite 330)	BlockID	BlockShowIcon (Seite 333)
BlockCount (Seite 330)	BlockIndex (Seite 332)	BlockShowTitleIcon (Seite 334)
BlockDateFormat (Seite 330)	BlockLength (Seite 332)	BlockTimeFormat (Seite 334)
BlockExponentialFormat (Seite 331)	BlockName (Seite 333)	BlockUseSourceFormat (Seite 334)

Siehe auch

GetRulerBlock-Methode (Seite 708)

GetRulerBlockCollection-Methode (Seite 709)

RulerColumn-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "Column" konfigurieren Sie die Spalten des Linealfensters im WinCC RulerControl.

Verwendung in den Controls

- WinCC RulerControl (Seite 275)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "rulercolobj.ColumnName" der Auflistungsname "Column" weggelassen: "rulercolobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

ColumnAdd (Seite 358)	ColumnName (Seite 363)	ColumnSort (Seite 366)
ColumnCount (Seite 359)	ColumnRemove (Seite 364)	ColumnSortIndex (Seite 367)
ColumnIndex (Seite 362)	ColumnRepos (Seite 364)	ColumnVisible (Seite 368)

Siehe auch

GetRulerColumn-Methode (Seite 710)

GetRulerColumnCollection-Methode (Seite 711)

StatisticAreaColumn-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "Column" konfigurieren Sie die Spalten des Statistikbereichfensters im WinCC RulerControl.

Verwendung in den Controls

- WinCC RulerControl (Seite 275)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "statareacolobj.ColumnName" der Auflistungsname "Column" weggelassen: "statareacolobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

ColumnAdd (Seite 358)	ColumnName (Seite 363)	ColumnSort (Seite 366)
ColumnCount (Seite 359)	ColumnRemove (Seite 364)	ColumnSortIndex (Seite 367)
ColumnIndex (Seite 362)	ColumnRepos (Seite 364)	ColumnVisible (Seite 368)

Siehe auch

GetStatisticAreaColumn-Methode (Seite 715)

GetStatisticAreaColumnCollection-Methode (Seite 716)

StatisticResultColumn-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "Column" konfigurieren Sie die Spalten des Statistikfensters im WinCC RulerControl.

Verwendung in den Controls

- WinCC RulerControl (Seite 275)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "statrescolobj.ColumnName" der Auflistungsname "Column" weggelassen: "statrescolobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

ColumnAdd (Seite 358)	ColumnName (Seite 363)	ColumnSort (Seite 366)
ColumnCount (Seite 359)	ColumnRemove (Seite 364)	ColumnSortIndex (Seite 367)
ColumnIndex (Seite 362)	ColumnRepos (Seite 364)	ColumnVisible (Seite 368)

Siehe auch

GetStatisticResultColumn-Methode (Seite 717)

GetStatisticResultColumnCollection-Methode (Seite 718)

StatusbarElement-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "StatusbarElement" konfigurieren Sie die Eigenschaften der Statuszeile der Controls.

Verwendung in den Controls

WinCC AlarmControl (Seite 251)	WinCC FunctionTrendControl (Seite 257)	WinCC OnlineTableControl (Seite 263)
WinCC OnlineTrendControl (Seite 267)	WinCC RulerControl (Seite 275)	WinCC UserArchiveControl (Seite 282)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "statusbarobj.StatusbarElementName" der Auflistungsname "StatusbarElement" weggelassen: "statusbarobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

StatusbarElementAdd (Seite 565)	StatusbarElementIndex (Seite 566)	StatusbarElementText (Seite 567)
StatusbarElementAutoSize (Seite 565)	StatusbarElementName (Seite 566)	StatusbarElementTooltipText (Seite 567)
StatusbarElementCount (Seite 565)	StatusbarElementRemove (Seite 566)	StatusbarElementUserDefined (Seite 568)
StatusbarElementIconId (Seite 566)	StatusbarElementRename (Seite 567)	StatusbarElementVisible (Seite 568)
StatusbarElementId (Seite 566)	StatusbarElementRepos (Seite 567)	StatusbarElementWidth (Seite 568)

Siehe auch

GetStatusbarElement-Methode (Seite 719)

GetStatusbarElementCollection-Methode (Seite 720)

TimeAxis-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "TimeAxis" konfigurieren Sie die Eigenschaften der Zeitachse im WinCC OnlineTrendControl.

Verwendung in den Controls

- WinCC OnlineTrendControl (Seite 267)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "timeaxisobj.TimeAxisName" der Auflistungsname "TimeAxis" weggelassen: "timeaxisobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

TimeAxisActualize	TimeAxisIndex (Seite 581)	TimeAxisRepos (Seite 583)
TimeAxisAdd (Seite 579)	TimeAxisInTrendColor	TimeAxisShowDate (Seite 584)
TimeAxisAlign (Seite 579)	TimeAxisLabel (Seite 582)	TimeAxisTimeFormat (Seite 584)
TimeAxisBeginTime (Seite 579)	TimeAxisMeasurePoints (Seite 582)	TimeAxisTimeRangeBase (Seite 584)
TimeAxisColor (Seite 580)	TimeAxisName (Seite 582)	TimeAxisTimeRangeFactor (Seite 585)
TimeAxisCount (Seite 580)	TimeAxisRangeType (Seite 583)	TimeAxisTrendWindow (Seite 585)
TimeAxisDateFormat (Seite 580)	TimeAxisRemove (Seite 583)	TimeAxisVisible (Seite 585)
TimeAxisEndTime (Seite 580)	TimeAxisRename (Seite 583)	

Siehe auch

GetTimeAxis-Methode (Seite 722)

GetTimeAxisCollection-Methode (Seite 723)

TimeColumn-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "TimeColumn" konfigurieren Sie die Eigenschaften der Zeitspalte im WinCC OnlineTableControl.

Verwendung in den Controls

- WinCC OnlineTableControl (Seite 263)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "timecolobj.TimeColumnName" der Auflistungsname "TimeColumn" weggelassen: "timecolobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

TimeColumnActualize (Seite 586)	TimeColumnHideText (Seite 589)	TimeColumnShowDate (Seite 592)
TimeColumnAdd (Seite 586)	TimeColumnHideTitleText (Seite 590)	TimeColumnShowIcon (Seite 592)
TimeColumnAlign (Seite 587)	TimeColumnIndex (Seite 590)	TimeColumnShowTitleIcon (Seite 593)
TimeColumnBackColor (Seite 587)	TimeColumnLength (Seite 590)	TimeColumnSort (Seite 593)
TimeColumnBeginTime (Seite 588)	TimeColumnMeasurePoints (Seite 590)	TimeColumnSortIndex (Seite 593)
TimeColumnCaption (Seite 588)	TimeColumnName (Seite 591)	TimeColumnTimeFormat (Seite 593)
TimeColumnCount (Seite 588)	TimeColumnRangeType (Seite 591)	TimeColumnTimeRangeBase (Seite 594)
TimeColumnDateFormat (Seite 588)	TimeColumnRemove (Seite 591)	TimeColumnTimeRangeFactor (Seite 594)
TimeColumnEndTime (Seite 589)	TimeColumnRename (Seite 591)	TimeColumnUseValueColumnColors (Seite 595)
TimeColumnForeColor (Seite 589)	TimeColumnRepos (Seite 592)	TimeColumnVisible (Seite 595)

Siehe auch

GetTimeColumn-Methode (Seite 724)

GetTimeColumnCollection-Methode (Seite 725)

ToolbarButton-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "ToolbarButton" konfigurieren Sie die Eigenschaften der Symbolleiste der Controls.

Verwendung in den Controls

WinCC AlarmControl (Seite 251)	WinCC FunctionTrendControl (Seite 257)	WinCC OnlineTableControl (Seite 263)
WinCC OnlineTrendControl (Seite 267)	WinCC RulerControl (Seite 275)	WinCC UserArchiveControl (Seite 282)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "toolbarobj.ToolbarButtonName" der Auflistungsname "ToolbarButton" weggelassen: "toolbarobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

ToolbarButtonActive (Seite 607)	ToolbarButtonId (Seite 611)	ToolbarButtonRename (Seite 612)
ToolbarButtonAdd (Seite 607)	ToolbarButtonIndex (Seite 611)	ToolbarButtonRepos (Seite 613)
ToolbarButtonBeginGroup (Seite 607)	ToolbarButtonLocked (Seite 612)	ToolbarButtonTooltipText (Seite 613)
ToolbarButtonCount (Seite 611)	ToolbarButtonName (Seite 612)	ToolbarButtonUserDefined (Seite 613)
ToolbarButtonEnabled (Seite 611)	ToolbarButtonPasswordLevel (Seite 612)	ToolbarButtonVisible (Seite 613)
ToolbarButtonHotKey (Seite 611)	ToolbarButtonRemove (Seite 612)	

Siehe auch

GetToolbarButton-Methode (Seite 727)

GetToolbarButtonCollection-Methode (Seite 728)

Trend-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "Trend" konfigurieren Sie die Eigenschaften der Kurven. Mit den Methoden "InsertData" und "RemoveData" füllen Sie die Kurve mit Daten bzw. löschen die Kurve. Über die Methode "GetRulerData" greifen Sie auf die Daten an einer bestimmten Stelle der Kurve zu.

Verwendung in den Controls

WinCC FunctionTrendControl (Seite 257)	WinCC OnlineTrendControl (Seite 267)

Verfügbare Methoden des Objekts

GetRulerData (Seite 712)	InsertData (Seite 743)	RemoveData (Seite 768)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "trendobj.Trendname" der Auflistungsname "Trend" weggelassen: "trendobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

Eigenschaften in WinCC FunctionTrendControl und WinCC OnlineTrendControl

TrendAdd (Seite 621)	TrendLineWidth (Seite 626)	TrendRemove (Seite 630)
TrendColor (Seite 622)	TrendLowerLimit (Seite 626)	TrendRename (Seite 630)
TrendCount (Seite 623)	TrendLowerLimitColor (Seite 626)	TrendRepos (Seite 630)
TrendExtendedColorSet	TrendLowerLimitColoring (Seite 626)	TrendTrendWindow (Seite 632)
TrendFill (Seite 624)	TrendName (Seite 627)	TrendUncertainColor (Seite 633)
TrendFillColor (Seite 624)	TrendPointColor (Seite 627)	TrendUncertainColoring (Seite 633)
TrendIndex (Seite 624)	TrendPointStyle	TrendUpperLimit (Seite 633)
TrendLabel (Seite 625)	TrendPointWidth (Seite 628)	TrendUpperLimitColor (Seite 633)
TrendLineStyle (Seite 625)	TrendProvider (Seite 628)	TrendUpperLimitColoring (Seite 634)
TrendLineType (Seite 625)	TrendProviderCLSID (Seite 629)	TrendVisible (Seite 635)

Eigenschaften im WinCC OnlineTrendControl

TrendAutoRangeBeginTagName (Seite 621)	TrendAutoRangeSource (Seite 622)	TrendValueAlignment
TrendAutoRangeBeginValue (Seite 621)	TrendSelectTagName (Seite 630)	TrendValueAxis (Seite 634)
TrendAutoRangeEndTagName (Seite 621)	TrendTagName (Seite 631)	TrendValueUnit
TrendAutoRangeEndValue (Seite 622)	TrendTimeAxis (Seite 632)	

Eigenschaften im WinCC FunctionTrendControl

TrendActualize (Seite 621)	TrendSelectTagNameX (Seite 630)	TrendTimeRangeBase (Seite 632)
TrendBeginTime (Seite 622)	TrendSelectTagNameY (Seite 631)	TrendTimeRangeFactor (Seite 632)
TrendEndTime (Seite 623)	TrendTagNameX (Seite 631)	TrendXAxis (Seite 641)
TrendMeasurePoints (Seite 627)	TrendTagNameY (Seite 631)	TrendYAxis (Seite 642)
TrendRangeType		

Siehe auch

GetTrend-Methode (Seite 729)

GetTrendCollection-Methode (Seite 730)

TrendWindow-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "TrendWindow" konfigurieren Sie die Eigenschaften der Kurvenfenster.

Verwendung in den Controls

WinCC FunctionTrendControl (Seite 257)	WinCC OnlineTrendControl (Seite 267)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "trendwndobj.TrendWindowName" der Auflistungsname "TrendWindow" weggelassen: "trendwndobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

Eigenschaften in WinCC FunctionTrendControl und WinCC OnlineTrendControl

TrendWindowAdd (Seite 635)	TrendWindowGridInTrendColor (Seite 637)	TrendWindowRulerColor (Seite 639)
TrendWindowCoarseGrid (Seite 635)	TrendWindowHorizontalGrid (Seite 637)	TrendWindowRulerLayer
TrendWindowCoarseGridColor (Seite 636)	TrendWindowIndex (Seite 638)	TrendWindowRulerStyle (Seite 639)
TrendWindowCount (Seite 636)	TrendWindowName (Seite 638)	TrendWindowRulerWidth (Seite 640)
TrendWindowFineGrid (Seite 636)	TrendWindowRemove (Seite 638)	TrendWindowSpacePortion (Seite 640)
TrendWindowFineGridColor (Seite 636)	TrendWindowRename (Seite 638)	TrendWindowVerticalGrid (Seite 641)
TrendWindowForegroundTrendGrid (Seite 637)	TrendWindowRepos (Seite 638)	TrendWindowVisible (Seite 641)

Eigenschaften im WinCC OnlineTrendControl

TrendWindowStatisticRulerColor (Seite 640)	TrendWindowStatisticRulerStyle (Seite 640)	TrendWindowStatisticRulerWidth (Seite 641)

Siehe auch

GetTrendWindow-Methode (Seite 731)

GetTrendWindowCollection-Methode (Seite 732)

ValueAxis-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "ValueAxis" konfigurieren Sie die Eigenschaften der Wertachse im WinCC OnlineTrendControl.

Verwendung in den Controls

- WinCC OnlineTrendControl (Seite 267)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "valueaxisobj.ValueAxisName" der Auflistungsname "ValueAxis" weggelassen: "valueaxisobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

ValueAxisAdd (Seite 657)	ValueAxisEndValue (Seite 659)	ValueAxisRemove (Seite 661)
ValueAxisAlign (Seite 657)	ValueAxisExponentialFormat (Seite 659)	ValueAxisRename (Seite 661)
ValueAxisAutoPrecisions (Seite 657)	ValueAxisIndex (Seite 659)	ValueAxisRepos (Seite 661)
ValueAxisAutoRange (Seite 658)	ValueAxisInTrendColor	ValueAxisScalingType (Seite 661)
ValueAxisBeginValue (Seite 658)	ValueAxisLabel (Seite 660)	ValueAxisTrendWindow (Seite 662)
ValueAxisColor (Seite 658)	ValueAxisName (Seite 660)	ValueAxisVisible (Seite 662)
ValueAxisCount (Seite 658)	ValueAxisPrecisions (Seite 661)	

Siehe auch

GetValueAxis-Methode (Seite 733)

GetValueAxisCollection-Methode (Seite 734)

ValueColumn-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "ValueColumn" konfigurieren Sie die Eigenschaften der Wertspalte im WinCC OnlineTableControl.

Verwendung in den Controls

- WinCC OnlineTableControl (Seite 263)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "valcolobj.ValueColumnName" der Auflistungsname "ValueColumn" weggelassen: "valcolobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

ValueColumnAdd (Seite 662)	ValueColumnHideTitleText (Seite 665)	ValueColumnRepos (Seite 667)
ValueColumnAlign (Seite 662)	ValueColumnIndex (Seite 665)	ValueColumnSelectTagName (Seite 668)
ValueColumnAutoPrecisions (Seite 663)	ValueColumnLength (Seite 666)	ValueColumnShowIcon (Seite 668)
ValueColumnBackColor (Seite 663)	ValueColumnName (Seite 666)	ValueColumnShowTitleIcon (Seite 668)
ValueColumnCaption (Seite 664)	ValueColumnPrecisions (Seite 666)	ValueColumnSort (Seite 668)
ValueColumnCount (Seite 664)	ValueColumnProvider (Seite 666)	ValueColumnSortIndex (Seite 669)
ValueColumnExponentialFormat (Seite 664)	ValueColumnProviderCLSID (Seite 667)	ValueColumnTagName (Seite 669)
ValueColumnForeColor (Seite 664)	ValueColumnRemove (Seite 667)	ValueColumnTimeColumn (Seite 669)
ValueColumnHideText (Seite 665)	ValueColumnRename (Seite 667)	ValueColumnVisible (Seite 670)

Siehe auch

GetValueColumn-Methode (Seite 735)

GetValueColumnCollection-Methode (Seite 736)

XAxis-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "Xaxis" konfigurieren Sie die Eigenschaften der X-Achse im WinCC FunctionTrendControl.

Verwendung in den Controls

- WinCC FunctionTrendControl (Seite 257)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "xaxisobj.XAxisName" der Auflistungsname "XAxis" weggelassen: "xaxisobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

XAxisAdd (Seite 676)	XAxisEndValue (Seite 679)	XAxisRemove (Seite 680)
XAxisAlign	XAxisExponentialFormat (Seite 679)	XAxisRename (Seite 682)
XAxisAutoPrecisions (Seite 677)	XAxisIndex (Seite 682)	XAxisRepos (Seite 681)
XAxisAutoRange (Seite 678)	XAxisInTrendColor	XAxisScalingType (Seite 681)
XAxisBeginValue (Seite 678)	XAxisLabel (Seite 680)	XAxisTrendWindow (Seite 681)
XAxisColor (Seite 678)	XAxisName (Seite 680)	XAxisVisible (Seite 682)
XAxisCount (Seite 682)	XAxisPrecisions (Seite 680)	

Siehe auch

GetXAxis-Methode (Seite 738)

GetXAxisCollection-Methode (Seite 739)

YAxis-Objekt (Auflistung)

Beschreibung

Die Auflistung eines Controls ist ein Daten-Container, der eine vom Benutzer veränderbare Anzahl von Objekten des gleichen Typs speichern kann.

Mit dem Auflistungsobjekt "Yaxis" konfigurieren Sie die Eigenschaften der Y-Achse im WinCC FunctionTrendControl.

Verwendung in den Controls

- WinCC FunctionTrendControl (Seite 257)

Verfügbare Eigenschaften des Objekts

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "yaxisobj.YAxisName" der Auflistungsname "YAxis" weggelassen: "yaxisobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus, z. B. "Add", "Remove" oder "Rename".

YAxisAdd (Seite 676)	YAxisEndValue (Seite 679)	YAxisRemove (Seite 680)
YAxisAlign (Seite 677)	YAxisExponentialFormat (Seite 679)	YAxisRename (Seite 683)
YAxisAutoPrecisions (Seite 677)	YAxisIndex (Seite 683)	YAxisRepos (Seite 681)
YAxisAutoRange (Seite 678)	YAxisInTrendColor (Seite 679)	YAxisScalingType (Seite 681)
YAxisBeginValue (Seite 678)	YAxisLabel (Seite 680)	YAxisTrendWindow (Seite 681)
YAxisColor (Seite 678)	YAxisName (Seite 680)	YAxisVisible (Seite 682)
YAxisCount (Seite 683)	YAxisPrecisions (Seite 680)	

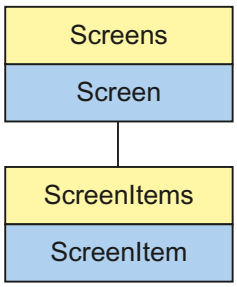
Siehe auch

GetYAxis-Methode (Seite 740)

GetYAxisCollection-Methode (Seite 741)

HMI Symbol Library

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "HMI Symbol Library".

Typkennzeichnung in VBS

HMISymbolLibrary

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Control1" um 20 Pixel nach rechts verschoben:

```
'VBS64
Dim objControl
Set objControl = ScreenItems("Control1")
objControl.Left = objControl.Left + 20
```

Eigenschaften

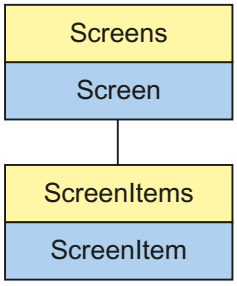
Dieser Objekt-Typ hat folgende Eigenschaften:

Siehe auch

Left-Eigenschaft (Seite 454)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Controls (Seite 226)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
Stretch-Eigenschaft (Seite 570)
Picture-Eigenschaft (Seite 514)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Object-Eigenschaft (Seite 489)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
ForeColor-Eigenschaft (Seite 412)
Flip-Eigenschaft (Seite 406)
Enabled-Eigenschaft (Seite 386)
Cursor-Eigenschaft (Seite 373)
BlinkColor-Eigenschaft (Seite 328)
BackStyle-Eigenschaft (Seite 320)
BackColor-Eigenschaft (Seite 316)

WinCC AlarmControl

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "WinCC AlarmControl" ab WinCC V7.0.

Typkennzeichnung in VBS

HMIAlarmControl

Verfügbare Auflistungs-Objekte

HitlistColumn (Seite 231)	Row (Seite 235)
MessageBlock (Seite 232)	StatusbarElement (Seite 239)
MessageColumn (Seite 233)	ToolbarButton (Seite 241)
OperatorColumn (Seite 234)	

Verfügbare Methoden in VBS

Activate	GetOperatorMessageCollection	MoveToLastLine	ShowHideList
ActivateDynamic	GetRow (Seite 705)	MoveToLastPage	ShowHitList
AttachDB	GetRowCollection (Seite 707)	MoveToNextLine	ShowInfoText
CopyRows	GetSelectedRow (Seite 713)	MoveToNextPage	ShowLockDialog
DeactivateDynamic	GetSelectedRows (Seite 714)	MoveToPreviousLine	ShowLockList
DetachDB	GetStatusbarElement	MoveToPreviousPage	ShowLongTermArchiveList
Export	GetStatusbarElementCollection	Print	ShowMessageList
GetHitlistColumn	GetToolbarButton	QuitHorn	ShowPropertyDialog
GetHitlistColumnCollection	GetToolbarButtonCollection	QuitSelected	ShowSelectionDialog
GetMessageBlock	HideAlarm	QuitVisible	ShowShortTermArchiveList
GetMessageBlockCollection	LockAlarm	ShowComment	ShowSortDialog

GetMessageColumn	LoopInAlarm	ShowDisplayOptionsDialog	ShowTimebaseDialog
GetMessageColumnCollection	MoveToFirstLine	ShowEmergencyQuitDialog	UnhideAlarm
GetOperatorMessage	MoveToFirstPage	ShowHelp	UnlockAlarm

Verfügbare Eigenschaften in VBS

Wenn Sie mithilfe eines Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "messagecol.MessageColumnName" der Auflistungsname "MessageColumn" weggelassen: "messagecol.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus wie z. B. "Add", "Remove" oder "Rename".

Activate	HitListRelTime	OperatorMessageSource10	StatusbarElementRepos
AllServer	HitListRelTimeFactor	OperatorMessageSourceType1	StatusbarElementText
ApplyProjectSettings	HitListRelTimeFactorType	OperatorMessageSourceType2	StatusbarElementTooltipText
AutoCompleteColumns	HorizontalGridLines	OperatorMessageSourceType3	StatusbarElementUserDefined
AutoCompleteRows	IconSpace	OperatorMessageSourceType4	StatusbarElementVisible
AutoScroll	LineColor (Seite 457)	OperatorMessageSourceType5	StatusbarElementWidth
AutoSelectionColors	LineWidth (Seite 459)	OperatorMessageSourceType6	StatusbarFontColor
AutoSelectionRectColor	LongTermArchiveConsistency	OperatorMessageSourceType7	StatusbarShowTooltips
BackColor	MessageBlockAlign	OperatorMessageSourceType8	StatusbarText
BorderColor	MessageBlockAutoPrecisions	OperatorMessageSourceType9	StatusbarUseBackColor
BorderWidth	MessageBlockCaption	OperatorMessageSourceType10	StatusbarVisible
Caption	MessageBlockCount	PageMode	TableColor
CellCut	MessageBlockDateFormat	PageModeMessageNumber	TableColor2
CellSpaceBottom	MessageBlockExponentialFormat	PrintJobName	TableForeColor
CellSpaceLeft	MessageBlockFlashOn	RowCellCount (Seite 530)	TableForeColor2
CellSpaceRight	MessageBlockHideText	RowCellText (Seite 530)	TimeBase
CellSpaceTop	MessageBlockHideTitleText	RowCount (Seite 531)	TitleColor
Closeable	MessageBlockIndex	RowNumber (Seite 531)	TitleCut
ColumnResize	MessageBlockLeadingZeros	RowScrollbar	TitleDarkShadowColor
ColumnTitleAlign	MessageBlockLength	RowTitleAlign	TitleForeColor
ColumnTitles	MessageBlockPrecisions	RowTitles	TitleGridLineColor

DefaultMsgFilterSQL	MessageBlockSelected	RTPersistence	TitleLightShadowColor
DefaultSort	MessageBlockShowDate	RTPersistencePasswordLevel	TitleSort
DefaultSort2	MessageBlockShowIcon	RTPersistenceType	TitleStyle
DefaultSort2Column	MessageBlockShowTitleIcon	SelectedCellColor	ToolbarAlignment
DisplayOptions	MessageBlockTextId	SelectedCellForeColor	ToolbarBackColor
DoubleClickAction	MessageBlockType	SelectedRowColor	ToolbarButtonActive
ExportDirectoryChangeable	MessageColumnAdd	SelectedRowForeColor	ToolbarButtonAdd
ExportDirectoryname	MessageColumnCount	SelectedTitleColor	ToolbarButtonBeginGroup
ExportFileExtension	MessageColumnIndex	SelectedTitleForeColor	ToolbarButtonClick
ExportFilename	MessageColumnName	SelectionColoring	ToolbarButtonCount
ExportFilenameChangeable	MessageColumnRemove	SelectionRect	ToolbarButtonEnabled
ExportFormatGuid	MessageColumnRepos	SelectionRectColor	ToolbarButtonHotKey
ExportFormatName	MessageColumnSort	SelectionRectWidth	ToolbarButtonId
ExportParameters	MessageColumnSortIndex	SelectionType	ToolbarButtonIndex
ExportSelection	MessageColumnVisible	ServerNames	ToolbarButtonLocked
ExportShowDialog	MessageListType	ShowSortButton	ToolbarButtonName
ExportXML	Moveable	ShowSortIcon	ToolbarButtonPasswordLevel
Font	MsgFilterSQL	ShowSortIndex	ToolbarButtonRemove
GridLineColor	OperatorMessageID	ShowTitle	ToolbarButtonRename
GridLineWidth	OperatorMessageIndex	Sizeable	ToolbarButtonRepos
HitlistColumnAdd	OperatorMessageName	SkinName	ToolbarButtonTooltipText
HitlistColumnCount	OperatorMessageNumber	SortSequence	ToolbarButtonUserDefined
HitlistColumnIndex	OperatorMessageSelected	StatusbarBackColor	ToolbarButtonVisible
HitlistColumnName	OperatorMessageSource1	StatusbarElementAdd	ToolbarShowTooltips
HitlistColumnRemove	OperatorMessageSource2	StatusbarElementAutoSize	ToolbarUseBackColor
HitlistColumnRepos	OperatorMessageSource3	StatusbarElementCount	ToolbarUseHotKeys
HitlistColumnSort	OperatorMessageSource4	StatusbarElementIconId	ToolbarVisible
HitlistColumnSortIndex	OperatorMessageSource5	StatusbarElementId	UseMessageColor
HitlistColumnVisible	OperatorMessageSource6	StatusbarElementIndex	UseSelectedTitleColor
HitListDefaultSort	OperatorMessageSource7	StatusbarElementName	UseTableColor2
HitListMaxSourceItems	OperatorMessageSource8	StatusbarElementRemove	VerticalGridLines
HitListMaxSourceItemsWarn	OperatorMessageSource9	StatusbarElementRename	

Beispiel

In einem bereits vorhandenen WinCC AlarmControl wird eine Selektion der Meldungen festgelegt. Für die Spalten werden Eigenschaften im Skript projektiert.

Voraussetzung

- Im Graphics Designer haben Sie bereits ein "WinCC AlarmControl" mit dem Namen "Control1" in ein Prozessbild eingefügt. Für das Beispiel wurde aus dem Demo-Projekt das Bild "C_015_Native_Alarms_Sel.pdl" verwendet.
- Im Graphics Designer ist ein Button eingefügt. Sie haben für den Button das Ereignis "Mausklick" projektiert, mit einer VBS-Aktion und folgendem Skript.
- Sie haben in Ihrem Projekt bereits Meldungen projektiert. Oder verwenden Sie das Demo-Projekt, aus dem die Meldungen für die Verwendung des Beispiels stammen.
- In Runtime wurden bereits Meldungen ausgelöst. Im Demo-Projekt wurde auf die Buttons "kommen" und "gehen" geklickt.

```
'VBS366
Sub OnClick(ByVal Item)
Dim objControl
Dim objMessColumn
Dim objMessBlock

Set objControl = ScreenItems("Control1")
objControl.ApplyProjectSettings = False
Set objMessBlock = objControl.GetMessageBlock("Date")
objMessBlock.DateFormat = "dd.MM.yy"
Set objMessColumn = objControl.GetMessageColumn("Time")
objMessColumn.Visible = False
objControl.MsgFilterSQL = "MSGNR >= 5 AND Priority = 0"
End Sub
```

Hinweis

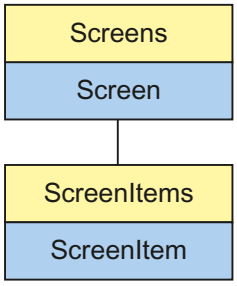
Weitere Beispiele zur Verwendung der Eigenschaften und Methoden finden Sie in den Beschreibungen der Get-Methoden der Controls und unter "Beispiele zu VBScript/Beispiele in WinCC/Dynamisieren der Controls".

Siehe auch

Controls (Seite 226)

WinCC Digital Analog Clock

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "WinCC Digital Analog Clock".

Typkennzeichnung in VBS

HMIClock

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Control1" um 11 Pixel nach rechts verschoben:

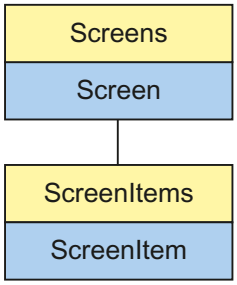
```
'VBS55
Dim objControl
Set objControl = ScreenItems("Control1")
objControl.Left = objControl.Left + 11
```

Siehe auch

Parent-Eigenschaft (Seite 506)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Controls (Seite 226)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
Ticks-Eigenschaft (Seite 577)
TicksColor-Eigenschaft (Seite 578)
SquareExtent-Eigenschaft (Seite 563)
SecondNeedleWidth-Eigenschaft (Seite 540)
SecondNeedleHeight-Eigenschaft (Seite 540)
Picture-Eigenschaft (Seite 514)
ObjectName-Eigenschaft (Seite 490)
Object-Eigenschaft (Seite 489)
MinuteNeedleWidth-Eigenschaft (Seite 485)
MinuteNeedleHeight-Eigenschaft (Seite 484)
LocaleID-Eigenschaft (Seite 460)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
HourNeedleWidth-Eigenschaft (Seite 427)
HourNeedleHeight-Eigenschaft (Seite 426)
Height-Eigenschaft (Seite 421)
Handtype-Eigenschaft (Seite 420)
HandFillColor-Eigenschaft (Seite 420)
ForeColor-Eigenschaft (Seite 412)
Font-Eigenschaft (Vor WinCC V7) (Seite 408)
FocusRect-Eigenschaft (Seite 407)
Enabled-Eigenschaft (Seite 386)
BackStyle-Eigenschaft (Seite 320)
BackColor-Eigenschaft (Seite 316)
Analog-Eigenschaft (Seite 306)

WinCC FunctionTrendControl

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "WinCC FunctionTrendControl ab WinCC V7.0.

Typkennzeichnung in VBS

HMIFunctionTrendControl

Verfügbare Auflistungs-Objekte

StatusbarElement (Seite 239)	Trend (Seite 242)	XAxis (Seite 247)
ToolbarButton (Seite 241)	TrendWindow (Seite 244)	YAxis (Seite 247)

Verfügbare Methoden in VBS

Activate	GetToolbarButtonCollection	MoveAxis	ShowTrendSelection
ActivateDynamic	GetTrend	NextTrend	StartStopUpdate
AttachDB	GetTrendCollection	OneToOneView	ZoomArea
DeactivateDynamic	GetTrendWindow	PreviousTrend	ZoomInOut
DetachDB	GetTrendWindowCollection	Print	ZoomInOutX
Export	GetXAxis	ShowHelp	ZoomInOutY
GetStatusbarElement	GetXAxisCollection	ShowPropertyDialog	ZoomMove
GetStatusbarElementCollection	GetYAxis	ShowTagSelection	
GetToolbarButton	GetYAxisCollection	ShowTimeSelection	

Verfügbare Eigenschaften in VBS

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "xaxisobj.XAxisName" der Auflistungsname "XAxis" weggelassen: "xaxisobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus wie z. B. "Add", "Remove" oder "Rename".

BackColor	StatusbarElementTooltipText	TrendLineWidth	TrendWindowVerticalGrid
BorderColor	StatusbarElementUserDefined	TrendLowerLimit	TrendWindowVisible
BorderWidth	StatusbarElementVisible	TrendLowerLimitColor	TrendXAxis
Caption	StatusbarElementWidth	TrendLowerLimitColoring	TrendYAxis
Closeable	StatusbarFontColor	TrendMeasurePoints	UseTrendNameAsLabel
ConnectTrendWindows	StatusbarShowTooltips	TrendName	XAxisAdd
ExportDirectoryChangeable	StatusbarText	TrendPointColor	XAxisAlign
ExportDirectoryname	StatusbarUseBackColor	TrendPointStyle	XAxisAutoPrecisions
ExportFileExtension	StatusbarVisible	TrendPointWidth	XAxisAutoRange
ExportFilename	TimeBase	TrendProvider	XAxisBeginValue
ExportFilenameChangeable	ToolbarAlignment	TrendProviderCLSID	XAxisColor
ExportFormatGuid	ToolbarBackColor	TrendRangeType	XAxisCount
ExportFormatName	ToolbarButtonActive	TrendRemove	XAxisEndValue
ExportSelection	ToolbarButtonAdd	TrendRename	XAxisExponentialFormat
ExportShowDialog	ToolbarButtonBeginGroup	TrendRepos	XAxisIndex
ExportParameters	ToolbarButtonClick	TrendSelectTagNameX	XAxisInTrendColor
ExportXML	ToolbarButtonCount	TrendSelectTagNameY	XAxisLabel
Font	ToolbarButtonEnabled	TrendTagNameX	XAxisName
GraphDirection	ToolbarButtonHotKey	TrendTagNameY	XAxisPrecisions
LineColor	ToolbarButtonId	TrendTimeRangeBase	XAxisRemove
LineWidth	ToolbarButtonIndex	TrendTimeRangeFactor	XAxisRename
LoadDataImmediately	ToolbarButtonLocked	TrendTrendWindow	XAxisRepos
Moveable	ToolbarButtonName	TrendUncertainColor	XAxisScalingType
Online	ToolbarButtonPasswordLevel	TrendUncertainColoring	XAxisTrendWindow
PrintJobName	ToolbarButtonRemove	TrendUpperLimit	XAxisVisible
RTPersistence	ToolbarButtonRename	TrendUpperLimitColor	XAxisAdd
RTPersistencePasswordLevel	ToolbarButtonRepos	TrendUpperLimitColoring	XAxisAlign
RTPersistenceType	ToolbarButtonTooltipText	TrendVisible	XAxisAutoPrecisions
ShowRuler	ToolbarButtonUserDefined	TrendWindowAdd	XAxisAutoRange
ShowRulerInAxis	ToolbarButtonVisible	TrendWindowCoarseGrid	XAxisBeginValue
ShowScrollbars	ToolbarShowTooltips	TrendWindowCount	XAxisColor
ShowTitle	ToolbarUseBackColor	TrendWindowCoarseGridColor	YAxisCount

Sizeable	ToolbarUseHotKeys	TrendWindowFineGrid	XAxisEndValue
ShowTrendIcon	ToolbarVisible	TrendWindowFineGridColor	XAxisExponentialFormat
SkinName	TrendActualize	TrendWindowForegroundTrendGrid	YAxisIndex
StatusbarBackColor	TrendAdd	TrendWindowGridInTrendColor	XAxisInTrendColor
StatusbarElementAdd	TrendBeginTime	TrendWindowHorizontalGrid	XAxisLabel
StatusbarElementAutoSize	TrendColor	TrendWindowIndex	XAxisName
StatusbarElementCount	TrendCount	TrendWindowName	XAxisPrecisions
StatusbarElementIconId	TrendEndTime	TrendWindowRemove	XAxisRemove
StatusbarElementId	TrendExtendedColorSet	TrendWindowRename	YAxisRename
StatusbarElementIndex	TrendFill	TrendWindowRepos	XAxisRepos
StatusbarElementName	TrendFillColor	TrendWindowRulerColor	XAxisScalingType
StatusbarElementRemove	TrendIndex	TrendWindowRulerLayer	XAxisTrendWindow
StatusbarElementRename	TrendLabel	TrendWindowRulerStyle	XAxisVisible
StatusbarElementRepos	TrendLineStyle	TrendWindowRulerWidth	
StatusbarElementText	TrendLineType	TrendWindowSpacePortion	

Beispiele

In einem WinCC FunctionTrendControl wird eine Kurve angezeigt, das mit einem Anwenderarchiv verbunden ist. Für die Kurve werden verschiedene Eigenschaften im Skript projektiert. Bezüglich der Datenanbindung wird die "StartID" des Anwenderarchivs und die Anzahl der Messpunkte geändert.

Voraussetzung

- Im Graphics Designer ist ein "WinCC FunctionTrendControl" mit dem Namen "Control1" in ein Prozessbild eingefügt.
- Im Graphics Designer ist ein Button eingefügt. Sie haben für den Button das Ereignis "Mausklick" projektiert, mit einer VBS-Aktion und folgendem Skript.
- Sie haben in Ihrem Projekt bereits ein Anwenderarchiv projektiert. Oder verwenden Sie das Demo-Projekt, aus dem das Anwenderarchiv des Beispiels stammt.

```
'VBS363
Sub OnClick(ByVal Item)
Dim objFXControl
Dim objTrendWindow
Dim objTrend
Dim objXAxis
Dim objYAxis
Dim startID
Dim FXServerDataX(3)
Dim FXServerDataY(3)
' create reference to FXControl
Set objFXControl = ScreenItems("Control1")
' create reference to new window, x and y axis
Set objTrendWindow = objFXControl.GetTrendWindowCollection.AddItem("myWindow")
```

```
Set objXAxis = objFXControl.GetXAxisCollection.AddItem("myXAxis")
Set objYAxis = objFXControl.GetYAxisCollection.AddItem("myYAxis")
' assign x and y axis to the window
objXAxis.TrendWindow = objTrendWindow.Name
objYAxis.TrendWindow = objTrendWindow.Name
' add new trend
Set objTrend = objFXControl.GetTrendCollection.AddItem("myTrend1")
' configure trend data connection (UserArchive)
objTrend.Provider = 3
startID = CLng(4)
FXServerDataX(0) = "Setpoint"
FXServerDataX(1) = "ParabelX"
FXServerDataX(3) = startID
FXServerDataY(0) = "Setpoint"
FXServerDataY(1) = "ParabelY"
FXServerDataY(3) = startID
objTrend.MeasurePoints = 50
objTrend.SetTagName "Setpoint\ParabelX", "Setpoint\ParabelY", FXServerDataX, FXServerDataY
' assign trend properties
objTrend.Color = RGB(255,0,0)
objTrend.PointStyle = 1
objTrend.TrendWindow = objTrendWindow.Name
objTrend.XAxis = objXAxis.Name
objTrend.YAxis = objYAxis.Name
End Sub
```

Hinweis

Weitere Beispiele zur Verwendung der Eigenschaften und Methoden finden Sie in den Beschreibungen der Get-Methoden der Controls und unter "Beispiele zu VBScript/Beispiele in WinCC/Dynamisieren der Controls".

Siehe auch

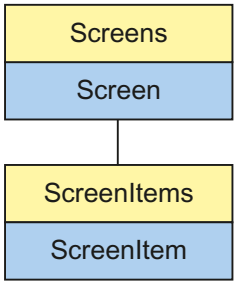
Controls (Seite 226)

ServerDataX (Seite 546)

ServerDataY (Seite 547)

WinCC Gauge Control

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "WinCC Gauge Control".

Typkennzeichnung in VBS

HMIGauge

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Control1" um 14 Pixel nach rechts verschoben:

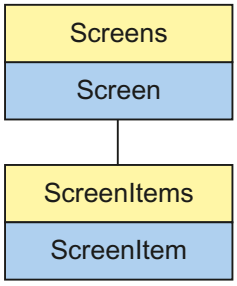
```
'VBS58
Dim objControl
Set objControl = ScreenItems("Control1")
objControl.Left = objControl.Left + 14
```

Siehe auch

WarningColor-Eigenschaft (Seite 672)
Object-Eigenschaft (Seite 489)
BackColor-Eigenschaft (Seite 316)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Objekt-Typen des Objekts ScreenItem (Seite 152)
Width-Eigenschaft (Seite 673)
Warning-Eigenschaft (Seite 672)
Visible-Eigenschaft (Seite 671)
ValueMin-Eigenschaft (Seite 670)
ValueMax-Eigenschaft (Seite 670)
ValueColumnAlignment-Eigenschaft (Seite 663)
UnitText-Eigenschaft (Seite 648)
UnitOffset-Eigenschaft (Seite 647)
UnitFont-Eigenschaft (Seite 647)
UnitColor-Eigenschaft (Seite 647)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
TicWidth-Eigenschaft (Seite 577)
TicTextOffset-Eigenschaft (Seite 577)
TicTextColor-Eigenschaft (Seite 576)
TicOffset-Eigenschaft (Seite 576)
TicFont-Eigenschaft (Seite 575)
TicColor-Eigenschaft (Seite 575)
ShowWarning-Eigenschaft (Seite 555)
ShowPeak-Eigenschaft (Seite 550)
ShowNormal-Eigenschaft (Seite 550)
ShowDecimalPoint-Eigenschaft (Seite 550)
ShowDanger-Eigenschaft (Seite 549)
Rectangular-Eigenschaft (Seite 526)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
NormalColor-Eigenschaft (Seite 488)
NeedleColor-Eigenschaft (Seite 488)
LocaleID-Eigenschaft (Seite 460)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)

WinCC Media Control

Beschreibung



Objekt-Typ des ScreenItem-Objekts. Repräsentiert das Grafikobjekt "WinCC Media Control" ab WinCC V7.0.

Objekt-Typ des ScreenItem-Objekts. Repräsentiert das Grafikobjekt "WinCC Media Control" ab WinCC V7.0.

Typkennzeichnung in VBS

HMIMediaControl

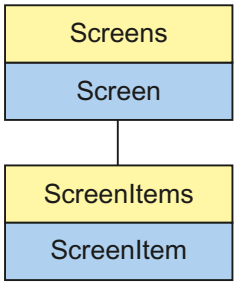
Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Control1" um 16 Pixel nach rechts verschoben:

```
'VBS60
Dim objControl
Set objControl = ScreenItems("Control1")
objControl.Left = objControl.Left + 16
```

WinCC OnlineTableControl

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "WinCC OnlineTableControl" ab WinCC V7.0.

Typkennzeichnung in VBS

HMIOnlineTableControl

Verfügbare Auflistungs-Objekte

Row (Seite 235)	ToolbarButton (Seite 241)
StatusbarElement (Seite 239)	ValueColumn (Seite 246)
TimeColumn (Seite 240)	

Verfügbare Methoden in VBS

Activate	GetRow (Seite 705)	GetToolbarButtonCollection	Print
ActivateDynamic	GetRowCollection (Seite 707)	GetValueColumn	SelectedStatisticArea
AttachDB	GetSelectedRow (Seite 713)	GetValueColumnCollection	ShowColumnSelection
CalculateStatistic	GetSelectedRows (Seite 714)	MoveToFirst	ShowHelp
CopyRows	GetStatusbarElement	MoveToLast	ShowPropertyDialog
DeactivateDynamic	GetStatusbarElementCollection	MoveToNext	ShowTagSelection
DetachDB	GetTimeColumn	MoveToPrevious	ShowTimeSelection
Edit	GetTimeColumnCollection	NextColumn	StartStopUpdate
Export	GetToolbarButton	PreviousColumn	

Verfügbare Eigenschaften in VBS

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "timecolobj.TimeColumnName" der Auflistungsname "TimeColumn" weggelassen: "timecolobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus wie z. B. "Add", "Remove" oder "Rename".

AutoCompleteColumns	RTPersistence	TimeColumnAdd	ToolbarButtonId
AutoCompleteRows	RTPersistencePasswordLevel	TimeColumnAlign	ToolbarButtonIndex
AutoSelectionColors	RTPersistenceType	TimeColumnBackColor	ToolbarButtonLocked
AutoSelectionRectColor	SelectedCellColor	TimeColumnBeginTime	ToolbarButtonName

BackColor	SelectedCellForeColor	TimeColumnCaption	ToolbarButtonPasswordLevel
BorderColor	SelectedRowColor	TimeColumnCount	ToolbarButtonRemove
BorderWidth	SelectedRowForeColor	TimeColumnDateFormat	ToolbarButtonRename
Caption	SelectedTitleColor	TimeColumnEndTime	ToolbarButtonRepos
CellCut	SelectedTitleForeColor	TimeColumnForeColor	ToolbarButtonTooltipText
CellSpaceBottom	SelectionColoring	TimeColumnHideText	ToolbarButtonUserDefined
CellSpaceLeft	SelectionRect	TimeColumnHideTitleText	ToolbarButtonVisible
CellSpaceRight	SelectionRectColor	TimeColumnIndex	ToolbarShowTooltips
CellSpaceTop	SelectionRectWidth	TimeColumnLength	ToolbarUseBackColor
Closeable	SelectionType	TimeColumnMeasurePoints	ToolbarUseHotKeys
ColumnResize	ShowSortButton	TimeColumnName	ToolbarVisible
ColumnScrollbar	ShowSortIcon	TimeColumnRangeType	UseColumnBackColor
ColumnTitleAlign	ShowSortIndex	TimeColumnRemove	UseColumnForeColor
ColumnTitles	ShowTitle	TimeColumnRename	UseSelectedTitleColor
EnableEdit	Sizeable	TimeColumnRepos	UseTableColor2
ExportDirectoryChangeable	SkinName	TimeColumnShowDate	ValueColumnAdd
ExportDirectoryname	SortSequence	TimeColumnShowIcon	ValueColumnAlign
ExportFileExtension	StatusbarBackColor	TimeColumnShowTitleIcon	ValueColumnAutoPrecisions
ExportFilename	StatusbarElementAdd	TimeColumnSort	ValueColumnBackColor
ExportFilenameChangeable	StatusbarElementAutoSize	TimeColumnSortIndex	ValueColumnCaption
ExportFormatGuid	StatusbarElementCount	TimeColumnTimeFormat	ValueColumnCount
ExportFormatName	StatusbarElementIconId	TimeColumnTimeRangeBase	ValueColumnExponentialFormat
ExportParameters	StatusbarElementId	TimeColumnTimeRangeFactor	ValueColumnForeColor
ExportSelection	StatusbarElementIndex	TimeColumnUseValueColumnColors	ValueColumnHideText
ExportShowDialog	StatusbarElementName	TimeColumnVisible	ValueColumnHideTitleText
ExportXML	StatusbarElementRemove	TimeStepBase	ValueColumnIndex
Font	StatusbarElementRename	TimeStepFactor	ValueColumnLength
GridLineColor	StatusbarElementRepos	TitleColor	ValueColumnName
GridLineWidth	StatusbarElementText	TitleCut	ValueColumnPrecisions
HorizontalGridLines	StatusbarElementTooltipText	TitleDarkShadowColor	ValueColumnProvider
IconSpace	StatusbarElementUserDefined	TitleForeColor	ValueColumnProviderCLSID
LineColor	StatusbarElementVisible	TitleGridLineColor	ValueColumnRemove
LineWidth	StatusbarElementWidth	TitleLightShadowColor	ValueColumnRename
LoadDataImmediately	StatusbarFontColor	TitleSort	ValueColumnRepos
Moveable	StatusbarShowTooltips	TitleStyle	ValueColumnSelectTagName
Online	StatusbarText	ToolbarAlignment	ValueColumnShowIcon
PrintJobName	StatusbarUseBackColor	ToolbarBackColor	ValueColumnShowTitleIcon
RowCellCount (Seite 530)	StatusbarVisible	ToolbarButtonActive	ValueColumnSort

RowCellText (Seite 530)	TableColor	ToolbarButtonAdd	ValueColumnSortIndex
RowCount (Seite 531)	TableColor2	ToolbarButtonBeginGroup	ValueColumnState
RowNumber (Seite 531)	TableForeColor	ToolbarButtonClick	ValueColumnTagName
RowScrollbar	TableForeColor2	ToolbarButtonCount	ValueColumnTimeColumn
RowTitleAlign	TimeBase	ToolbarButtonEnabled	ValueColumnVisible
RowTitles	TimeColumnActualize	ToolbarButtonHotKey	VerticalGridLines

Beispiel

In einem bereits vorhandenen WinCC OnlineTableControl wird eine zusätzliche Spalte eingefügt, die mit einer Archivvariablen verbunden wird. Für das Control und die Spalte werden verschiedene Eigenschaften im Skript projektiert.

Voraussetzung

- Im Graphics Designer haben Sie bereits ein "WinCC OnlineTableControl" mit dem Namen "Control1" in ein Prozessbild eingefügt. Das Control besteht aus einer Zeitspalte und drei Wertspalten. Für das Beispiel wurde aus dem Demo-Projekt das Bild "B_025_V7_Arch_TableControl.PDL" verwendet.
- Im Graphics Designer ist ein Button eingefügt. Sie haben für den Button das Ereignis "Mausklick" projektiert, mit einer VBS-Aktion und folgendem Skript.
- Sie haben in Ihrem Projekt bereits Archive und Archivvariablen projektiert. Oder verwenden Sie das Demo-Projekt, aus dem das Archiv des Beispiels stammt.

```
'VBS362
Sub OnClick(ByVal Item)
Dim objControl
Dim objTimeColumn
Dim objValueColumn
Set objControl = ScreenItems("Control1")
' Control wide spezification
objControl.ColumnResize = False
objControl.TimeBase = 1
objControl.TimeColumnTimeFormat = "HH:mm:ss tt"
objControl.TimeColumnLength = 20
' properties for Time column
Set objTimeColumn = objControl.GetTimeColumn("Time column 1")
objTimeColumn.DateFormat = "dd/MM/yy"
' properties for a new 4th value column with connection to archive tag "Trend_4"
Set objValueColumn = objControl.GetValueColumnCollection.AddItem("Trend 4")
objValueColumn.Caption = "Trend 4"
objValueColumn.Length = 10
objValueColumn.Align = 1
objValueColumn.Provider = 1
objValueColumn.TagName = "G_Archive\Trend_4"
objValueColumn.TimeColumn = "Time column 1"
End Sub
```

Hinweis

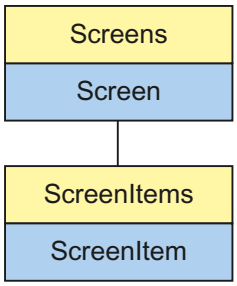
Weitere Beispiele zur Verwendung der Eigenschaften und Methoden finden Sie in den Beschreibungen der Get-Methoden der Controls und unter "Beispiele zu VBScript/Beispiele in WinCC/Dynamisieren der Controls".

Siehe auch

Controls (Seite 226)

WinCC OnlineTrendControl

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "WinCC OnlineTrendControl" ab WinCC V7.0.

Typkennzeichnung in VBS

HMIOnlineTrendControl

Verfügbare Auflistungs-Objekte

StatusbarElement (Seite 239)	ToolbarButton (Seite 241)	TrendWindow (Seite 244)
TimeAxis (Seite 239)	Trend (Seite 242)	ValueAxis (Seite 245)

Verfügbare Methoden in VBS

Activate	GetTimeAxisCollection	MoveToFirst	ShowPropertyDialog
ActivateDynamic-Methode	GetToolbarButton (Seite 727)	MoveToLast	ShowTagSelection
AttachDB-Methode	GetToolbarButtonCollection (Seite 728)	MoveToNext	ShowTimeSelection

CalculateStatistic	GetTrend	MoveToPrevious	ShowTrendSelection
DeactivateDynamic	GetTrendCollection	NextTrend	StartStopUpdate
DetachDB	GetTrendWindow	OneToOneView	ZoomArea
Export	GetTrendWindowCollection	PreviousTrend	ZoomInOut
GetStatusbarElement	GetValueAxis	Print	ZoomInOutTime
GetStatusbarElementCollection	GetValueAxisCollection	ShowHelp	ZoomInOutValues
GetTimeAxis	MoveAxis	ShowPercentageAxis	ZoomMove

Verfügbare Eigenschaften in VBS

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "trendobj.Trendname" der Auflistungsname "Trend" weggelassen: "trendobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus wie z. B. "Add", "Remove" oder "Rename".

BackColor	StatusbarElementRepos	ToolbarButtonRemove	TrendValueUnit
BorderColor	StatusbarElementText	ToolbarButtonRename	TrendVisible
BorderWidth	StatusbarElementTooltipText	ToolbarButtonRepos	TrendWindowAdd
Caption	StatusbarElementUserDefined	ToolbarButtonTooltipText	TrendWindowCoarseGrid
Closeable	StatusbarElementVisible	ToolbarButtonUserDefined	TrendWindowCoarseGridColor
ConnectTrendWindows	StatusbarElementWidth	ToolbarButtonVisible	TrendWindowCount
ExportDirectoryChangeable	StatusbarFontColor	ToolbarShowTooltips	TrendWindowFineGrid
ExportDirectoryname	StatusbarShowTooltips	ToolbarUseBackColor	TrendWindowFineGridColor
ExportFileExtension	StatusbarText	ToolbarUseHotKeys	TrendWindowForegroundTrendGrid
ExportFilename	StatusbarUseBackColor	ToolbarVisible	TrendWindowGridInTrendColor
ExportFilenameChangeable	StatusbarVisible	TrendAdd	TrendWindowHorizontalGrid
ExportFormatGuid	TimeAxisActualize	TrendAutoRangeBeginTagName	TrendWindowIndex
ExportFormatName	TimeAxisAdd	TrendAutoRangeBeginValue	TrendWindowName
ExportParameters	TimeAxisAlign	TrendAutoRangeEndTagName	TrendWindowRemove
ExportSelection	TimeAxisBeginTime	TrendAutoRangeEndValue	TrendWindowRename
ExportShowDialog	TimeAxisColor	TrendAutoRangeSource	TrendWindowRepos
ExportXML	TimeAxisCount	TrendColor	TrendWindowRulerColor
Font	TimeAxisDateFormat	TrendCount	TrendWindowRulerLayer

GraphDirection	TimeAxisEndTime	TrendExtendedColorSet	TrendWindowRulerStyle
LineColor	TimeAxisIndex	TrendFill	TrendWindowRulerWidth
LineWidth	TimeAxisInTrendColor	TrendFillColor	TrendWindowSpacePortion
LoadDataImmediately	TimeAxisLabel	TrendIndex	TrendWindowStatisticRulerColor
Moveable	TimeAxisMeasurePoints	TrendLabel	TrendWindowStatisticRulerStyle
Online	TimeAxisName	TrendLineStyle	TrendWindowStatisticRulerWidth
PercentageAxis	TimeAxisRangeType	TrendLineType	TrendWindowVerticalGrid
PercentageAxisAlign	TimeAxisRemove	TrendLineWidth	TrendWindowVisible
PercentageAxisColor	TimeAxisRename	TrendLowerLimit	UseTrendNameAsLabel
PrintJobName	TimeAxisRepos	TrendLowerLimitColor	ValueAxisAdd
RTPersistence	TimeAxisShowDate	TrendLowerLimitColoring	ValueAxisAlign
RTPersistencePasswordLevel	TimeAxisTimeFormat	TrendName	ValueAxisAutoPrecisions
RTPersistenceType	TimeAxisTimeRangeBase	TrendPointColor	ValueAxisAutoRange
ShowRuler	TimeAxisTimeRangeFactor	TrendPointStyle	ValueAxisBeginValue
ShowRulerInAxis	TimeAxisTrendWindow	TrendPointWidth	ValueAxisColor
ShowScrollbars	TimeAxisVisible	TrendProvider	ValueAxisCount
ShowStatisticRuler	TimeBase	TrendProviderCLSID	ValueAxisEndValue
ShowTitle	ToolbarAlignment	TrendRemove	ValueAxisExponentialFormat
ShowTrendIcon	ToolbarBackColor	TrendRename	ValueAxisIndex
Sizeable	ToolbarButtonActive	TrendRepos	ValueAxisInTrendColor
SkinName	ToolbarButtonAdd	TrendSelectTagName	ValueAxisLabel
StatusbarBackColor	ToolbarButtonBeginGroup	TrendTagName	ValueAxisName
StatusbarElementAdd	ToolbarButtonClick	TrendTimeAxis	ValueAxisPrecisions
StatusbarElementAutoSize	ToolbarButtonCount	TrendTrendWindow	ValueAxisRemove
StatusbarElementCount	ToolbarButtonEnabled	TrendUncertainColor	ValueAxisRename
StatusbarElementIconId	ToolbarButtonHotKey	TrendUncertainColoring	ValueAxisRepos
StatusbarElementId	ToolbarButtonId	TrendUpperLimit	ValueAxisScalingType
StatusbarElementIndex	ToolbarButtonIndex	TrendUpperLimitColor	ValueAxisTrendWindow
StatusbarElementName	ToolbarButtonLocked	TrendUpperLimitColoring	ValueAxisVisible
StatusbarElementRemove	ToolbarButtonName	TrendValueAlignment	
StatusbarElementRename	ToolbarButtonPasswordLevel	TrendValueAxis	

Beispiel

In einem WinCC OnlineTrendControl werden drei Kurven angezeigt, die mit Archivvariablen verbunden sind. Für die Kurven werden verschiedene Eigenschaften im Skript projektiert.

Voraussetzung

- Im Graphics Designer ist ein "WinCC OnlineTrendControl" mit dem Namen "Control1" in ein Prozessbild eingefügt.
- Im Graphics Designer ist ein Button eingefügt. Sie haben für den Button das Ereignis "Mausklick" projektiert, mit einer VBS-Aktion und folgendem Skript.
- Sie haben in Ihrem Projekt bereits Archive und Archivvariablen projektiert. Oder verwenden Sie das Demo-Projekt, aus dem die Archive des Beispiels stammen.

```
'VBS361
Sub OnClick(ByVal Item)
Dim objTrendControl
Dim objTrendWindow
Dim objTimeAxis
Dim objValueAxis
Dim objTrend
' create reference to TrendControl
Set objTrendControl = ScreenItems("Control1")
' create reference to new window, time and value axis
Set objTrendWindow = objTrendControl.GetTrendWindowCollection.AddItem("myWindow")
Set objTimeAxis = objTrendControl.GetTimeAxisCollection.AddItem("myTimeAxis")
Set objValueAxis = objTrendControl.GetValueAxisCollection.AddItem("myValueAxis")
' assign time and value axis to the window
objTimeAxis.TrendWindow = objTrendWindow.Name
objValueAxis.TrendWindow = objTrendWindow.Name
' assign properties to trendwindow
objTrendWindow.HorizontalGrid = False
' add new trend and assign properties
Set objTrend = objTrendControl.GetTrendCollection.AddItem("myTrend1")
objTrend.Provider = 1
objTrend.TagName = "G_Archive\Trend_1"
objTrend.TrendWindow = objTrendWindow.Name
objTrend.TimeAxis = objTimeAxis.Name
objTrend.ValueAxis = objValueAxis.Name
objTrend.Color = RGB(255,0,0)
objTrend.PointStyle = 0
' add new trend and assign propertys
Set objTrend = objTrendControl.GetTrendCollection.AddItem("myTrend2")
objTrend.Provider = 1
objTrend.TagName = "G_Archive\Trend_2"
objTrend.TrendWindow = objTrendWindow.Name
objTrend.TimeAxis = objTimeAxis.Name
objTrend.ValueAxis = objValueAxis.Name
objTrend.Color = RGB(0,255,0)
objTrend.LineWidth = 3
' add new trend and assign propertys
Set objTrend = objTrendControl.GetTrendCollection.AddItem("myTrend3")
objTrend.Provider = 1
objTrend.TagName = "G_Archive\Trend_3"
objTrend.TrendWindow = objTrendWindow.Name
objTrend.TimeAxis = objTimeAxis.Name
objTrend.ValueAxis = objValueAxis.Name
objTrend.Color = RGB(0,0,255)
objTrend.LineType = 2
```

```
End Sub
```

Hinweis

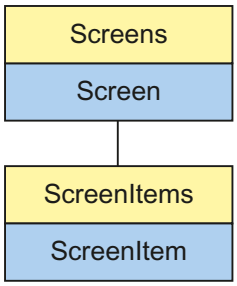
Weitere Beispiele zur Verwendung der Eigenschaften und Methoden finden Sie in den Beschreibungen der Get-Methoden der Controls und unter "Beispiele zu VBScript/Beispiele in WinCC/Dynamisieren der Controls".

Siehe auch

Controls (Seite 226)

WinCC Push Button Control

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "WinCC Push Button Control".

Typkennzeichnung in VBS

HMIButton

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Control1" um 17 Pixel nach rechts verschoben:

```
'VBS61
Dim objControl
Set objControl = ScreenItems("Control1")
objControl.Left = objControl.Left + 17
```

Hinweis

Die Ereignisse KeyDown, KeyUp und KeyPress können durch VBS nicht angesprochen werden. Wenn Controls mit VBS dynamisiert werden sollen, darf kein Parameter mit ByRef deklariert sein.

Hinweise zur Fehlerbehandlung

Buttons und PushButtons werden im Objektmodell auf einen Typ "HMIButton" abgebildet. Da die Objekte dennoch teilweise unterschiedliche Eigenschaften aufweisen, sollten Sie das Vorhandensein der Eigenschaften (dynamische Typ-Ermittlung in Runtime) über eine Ausnahmebehandlung abfragen. Die Ausnahmebehandlung wird durch folgende Anweisung für die entsprechende Prozedur aktiviert:

```
On Error Resume Next
```

Die Anweisung führt dazu, dass die VBScript-Engine nach einem Laufzeitfehler die Folgeanweisung ausführt.

In der Folgezeile kann dann der Fehlercode über das Err-Objekt überprüft werden. Um die Behandlung von Laufzeitfehlern im Skript wieder abzuschalten, wird folgende Anweisung verwendet:

```
On Error Goto 0
```

Die Fehlerbehandlung ist immer auf die Prozedurebene bezogen. Löst ein Skript in einer Prozedur einen Fehler aus, prüft VBScript, ob auf dieser Ebene eine Fehlerbehandlung implementiert ist. Ist dies nicht der Fall, geht die Kontrolle eine Ebene höher (zur rufenden Prozedur). Gibt es dort ebenfalls keine Fehlerbehandlung, wird die Kontrolle wiederum eine Ebene höher übergeben. Dies setzt sich solange fort, bis entweder die oberste Modulebene erreicht ist oder der Code zur Laufzeitfehlerbehandlung gefunden wird. Fehlt die Aktivierung der Runtime-Fehlerbehandlung, wird die Kontrolle auf der Hauptebene an die interne VBScript-Laufzeitfehlerbehandlung übergeben. Diese zeigt dann einen Fehlerdialog an und bricht das Skript ab.

Die "On Error Resume Next"-Anweisung kann auf jeder Ebene (also auch in Prozeduren) eingebaut werden. Mit dem Einsatz der Fehlerbehandlung ist prinzipiell feststellbar, ob der Anwender den gewünschten Implementierungstyp auch wirklich verwendet.

Außerdem kann garantiert werden, dass es nicht zu einem Abbruch der Ausführung aufgrund eines fehlerhaften Zugriffs auf das Objekt kommt.

Beispiel zur Fehlerbehandlung

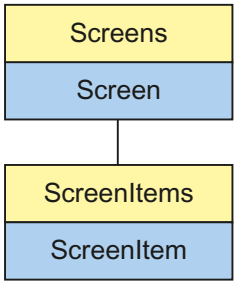
```
'VBS62
Dim objScreenItem
On Error Resume Next    'Activation of errorhandling
For Each objScreenItem In ScreenItems
If objScreenItem.Type = "HMIButton" Then
'
'=== Property "Text" available only for Standard-Button
objScreenItem.Text = "Windows"
If 0 <> Err.Number Then
HMIRuntime.Trace objScreenItem.ObjectName & ": no Windows-Button" & vbCrLf
Err.Clear    'Delete error message
End If
'
'=== Property "Caption" available only for PushButton
objScreenItem.Caption = "Push"
If 0 <> Err.Number Then
HMIRuntime.Trace objScreenItem.ObjectName & ": no Control" & vbCrLf
Err.Clear
End If
End If
Next
On Error Goto 0    'Deactivation of errorhandling
```

Siehe auch

Eigenschaften (Seite 295)
FontName-Eigenschaft (Seite 410)
Activate-Methode (Seite 686)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Controls (Seite 226)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Transparent-Eigenschaft (Seite 620)
Top-Eigenschaft (Seite 618)
PictureUnselected-Eigenschaft (Seite 516)
PictureSelected-Eigenschaft (Seite 515)
Parent-Eigenschaft (Seite 506)
Outline-Eigenschaft (Seite 505)
ObjectName-Eigenschaft (Seite 490)
Object-Eigenschaft (Seite 489)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
FrameWidth-Eigenschaft (Seite 415)
FrameColorUp-Eigenschaft (Seite 414)
FrameColorDown-Eigenschaft (Seite 414)
ForeColor-Eigenschaft (Seite 412)
FontUnderline-Eigenschaft (Seite 411)
FontStrikeThru-Eigenschaft (Seite 411)
FontSize-Eigenschaft (Seite 411)
FontItalic-Eigenschaft (Seite 409)
Font-Eigenschaft (Vor WinCC V7) (Seite 408)
FontBold-Eigenschaft (Seite 409)
FocusRect-Eigenschaft (Seite 407)
Enabled-Eigenschaft (Seite 386)
Caption-Eigenschaft (Seite 342)
BackColor-Eigenschaft (Seite 316)
AutoSize-Eigenschaft (Seite 314)

WinCC RulerControl

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "WinCC RulerControl" ab WinCC V7.0.

Typkennzeichnung in VBS

HMIRulerControl

Verfügbare Auflistungs-Objekte

Row (Seite 235)	StatisticResultColumn (Seite 238)
RulerBlock (Seite 236)	StatusbarElement (Seite 239)
RulerColumn (Seite 236)	ToolbarButton (Seite 241)
StatisticAreaColumn (Seite 237)	

Verfügbare Methoden in VBS

Activate	GetRulerBlock	GetStatisticAreaColumn	GetToolbarButton
ActivateDynamic	GetRulerBlockCollection	GetStatisticAreaColumnCollection	GetToolbarButtonCollection
DeactivateDynamic	GetRulerColumn	GetStatisticResultColumn	ShowHelp
Export	GetRulerColumnCollection	GetStatisticResultColumnCollection	ShowPropertyDialog
GetRow (Seite 705)	GetSelectedRow (Seite 713)	GetStatusbarElement	
GetRowCollection (Seite 707)	GetSelectedRows (Seite 714)	GetStatusbarElementCollection	

Verfügbare Eigenschaften in VBS

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "rulercolobj.ColumnName" der Auflistungsname "Column" weggelassen: "rulercolobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus wie z. B. "Add", "Remove" oder "Rename".

AutoCompleteColumns	ColumnScrollbar	SelectedRowForeColor	TableColor2
AutoCompleteRows	ColumnSort	SelectedTitleColor	TableForeColor
AutoPositon	ColumnSortIndex	SelectedTitleForeColor	TableForeColor2
AutoSelectionColors	ColumnTitleAlign	SelectionColoring	TitleColor
AutoSelectionRectColor	ColumnTitles	SelectionRect	TitleCut
AutoShow	ColumnVisible	SelectionRectColor	TitleDarkShadowColor
BackColor	ExportDirectoryChangeable	SelectionRectWidth	TitleForeColor
BlockAlign	ExportDirectoryname	SelectionType	TitleGridLineColor
BlockAutoPrecisions	ExportFileExtension	ShareSpaceWithSourceControl	TitleLightShadowColor
BlockCaption	ExportFilename	ShowSortButton	TitleSort
BlockCount	ExportFilenameChangeable	ShowSortIcon	TitleStyle
BlockDateFormat	ExportFormatGuid	ShowSortIndex	ToolbarAlignment
BlockExponentialFormat	ExportFormatName	ShowTitle	ToolbarBackColor
BlockHideText	ExportParameters	Sizeable	ToolbarButtonActive
BlockHideTitleText	ExportSelection	SkinName	ToolbarButtonAdd
BlockID	ExportShowDialog	SortSequence	ToolbarButtonBeginGroup
BlockIndex	ExportXML	SourceControl	ToolbarButtonClick
BlockLength	Font	SourceControlType	ToolbarButtonCount
BlockName	GridLineColor	StatusbarBackColor	ToolbarButtonEnabled
BlockPrecisions	GridLineWidth	StatusbarElementAdd	ToolbarButtonHotKey
BlockShowDate	HorizontalGridLines	StatusbarElementAutoSize	ToolbarButtonId
BlockShowIcon	IconSpace	StatusbarElementCount	ToolbarButtonIndex
BlockShowTitleIcon	LineColor	StatusbarElementIconId	ToolbarButtonLocked
BlockTimeFormat	LineWidth	StatusbarElementId	ToolbarButtonName
BlockUseSourceFormat	Moveable	StatusbarElementIndex	ToolbarButtonPasswordLevel
BorderColor	PrintJobName	StatusbarElementName	ToolbarButtonRemove
BorderWidth	RowCellCount (Seite 530)	StatusbarElementRemove	ToolbarButtonRename
Caption	RowCellText (Seite 530)	StatusbarElementRename	ToolbarButtonRepos
CellCut	RowCount (Seite 531)	StatusbarElementRepos	ToolbarButtonTooltipText
CellSpaceBottom	RowNumber (Seite 531)	StatusbarElementText	ToolbarButtonUserDefined
CellSpaceLeft	RowScrollbar	StatusbarElementTooltipText	ToolbarButtonVisible
CellSpaceRight	RowTitleAlign	StatusbarElementUserDefined	ToolbarShowTooltips
CellSpaceTop	RowTitles	StatusbarElementVisible	ToolbarUseBackColor
Closeable	RTPersistence	StatusbarElementWidth	ToolbarUseHotKeys
ColumnAdd	RTPersistencePasswordLevel	StatusbarFontColor	ToolbarVisible
ColumnCount	RTPersistenceType	StatusbarShowTooltips	UseSelectedTitleColor

ColumnIndex	RulerType	StatusbarText	UseSourceBackColors
ColumnName	SelectedCellColor	StatusbarUseBackColor	UseSourceForeColors
ColumnRemove	SelectedCellForeColor	StatusbarVisible	UseTableColor2
ColumnRepos	SelectedRowColor	TableColor	VerticalGridLines
ColumnResize			

Beispiel

In einem Bild mit einem bereits vorhandenen WinCC OnlineTableControl wird zusätzlich ein WinCC RulerControl eingefügt. Das RulerControl beinhaltet ein Statistikfenster, dass die Spalten "Minimum", "Maximum" und "Durchschnitt" anzeigt. Für die ausgewählten Zeilen des OnlineTableControl werden dann die Statistikwerte angezeigt.

Voraussetzung

- Im Graphics Designer haben Sie bereits ein "WinCC OnlineTableControl" mit dem Namen "Control1" in ein Prozessbild eingefügt. Das Control ist mit Archivvariablen oder Prozessvariablen verbunden. Für das Beispiel wurde aus dem Demo-Projekt das Bild "B_025_V7_Arch_TableControl.PDL" verwendet.
- Sie haben zusätzlich ein "WinCC RulerControl" mit dem Namen "Control2" in das Bild eingefügt.
- Im Graphics Designer ist ein Button eingefügt. Sie haben für den Button das Ereignis "Mausklick" projektiert, mit einer VBS-Aktion und folgendem Skript.
- In Runtime haben Sie im OnlineTableControl einige Zeilen markiert.

```
'VBS364
Sub OnClick(ByVal Item)
Dim objRulerControl
Dim objTableControl
Dim objstatColumn
Dim rows

Set objRulerControl = ScreenItems("Control2")
' Use Statistic-window
objRulerControl.RulerType = 2
objRulerControl.SourceControl = "Control1"
' In Statistic-window only columns "Name", "MinValue", MaxValue" and "Average" are shown
Set objstatColumn = objRulerControl.GetStatisticResultColumnCollection
objstatColumn.RemoveItem(4)
objstatColumn.RemoveItem(5)
objstatColumn.RemoveItem(6)
' Get the selected rows of tablecontrol and calculate statistic
Set objTableControl = ScreenItems("Control1")
Set rows = objTableControl.SelectAll
objTableControl.CalculateStatistic
End Sub
```

Hinweis

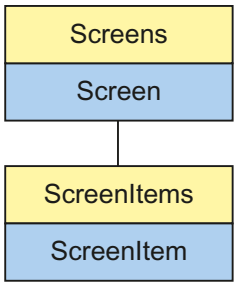
Weitere Beispiele zur Verwendung der Eigenschaften und Methoden finden Sie in den Beschreibungen der Get-Methoden der Controls und unter "Beispiele zu VBScript/Beispiele in WinCC/Dynamisieren der Controls".

Siehe auch

Controls (Seite 226)

WinCC Slider Control

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "WinCC Slider Control".

Typkennzeichnung in VBS

HMISlider

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Control1" um 19 Pixel nach rechts verschoben:

```
'VBS63
Dim objControl
Set objControl = ScreenItems("Control1")
objControl.Left = objControl.Left + 19
```

Hinweise zur Fehlerbehandlung

Slider und WinCC Slider Controls werden im Objektmodell auf einen Typ "HMISlider" abgebildet. Da die Objekte dennoch teilweise unterschiedliche Eigenschaften aufweisen, sollten Sie das Vorhandensein der Eigenschaften (dynamische Typ-Ermittlung in Runtime)

über eine Ausnahmebehandlung abfragen. Die Ausnahmebehandlung wird durch folgende Anweisung für die entsprechende Prozedur aktiviert:

```
On Error Resume Next
```

Die Anweisung führt dazu, dass die VBScript-Engine nach einem Laufzeitfehler die Folgeanweisung ausführt.

In der Folgezeile kann dann der Fehlercode über das Err-Objekt überprüft werden. Um die Behandlung von Laufzeitfehlern im Skript wieder abzuschalten, wird folgende Anweisung verwendet:

```
On Error Goto 0
```

Die Fehlerbehandlung ist immer auf die Prozedurebene bezogen. Löst ein Skript in einer Prozedur einen Fehler aus, prüft VBScript, ob auf dieser Ebene eine Fehlerbehandlung implementiert ist. Ist dies nicht der Fall, geht die Kontrolle eine Ebene höher (zur rufenden Prozedur). Gibt es dort ebenfalls keine Fehlerbehandlung, wird die Kontrolle wiederum eine Ebene höher übergeben. Dies setzt sich solange fort, bis entweder die oberste Modulebene erreicht ist oder der Code zur Laufzeitfehlerbehandlung gefunden wird. Fehlt die Aktivierung der Runtime-Fehlerbehandlung, wird die Kontrolle auf der Hauptebene an die interne VBScript-Laufzeitfehlerbehandlung übergeben. Diese zeigt dann einen Fehlerdialog an und bricht das Skript ab.

Die "On Error Resume Next"-Anweisung kann auf jeder Ebene (also auch in Prozeduren) eingebaut werden. Mit dem Einsatz der Fehlerbehandlung ist prinzipiell feststellbar, ob der Anwender den gewünschten Implementierungstyp auch wirklich verwendet.

Außerdem kann garantiert werden, dass es nicht zu einem Abbruch der Ausführung aufgrund eines fehlerhaften Zugriffs auf das Objekt kommt.

Beispiel zur Fehlerbehandlung

```
Sub OnClick(Byval Item)
'VBS193
Dim ScreenItem
' activating error handling:
On Error Resume Next
For Each ScreenItem In ScreenItems
If ScreenItem.Type = "HMISlider" Then
'=== Property "BevelColorUp" only exists for a WinCC Slider Control
ScreenItem.BevelColorUp = 1
If (Err.Number <> 0) Then
HMIRuntime.Trace(ScreenItem.ObjectName + ": no Windows-Slider" + vbCrlf)
' delete error message
Err.Clear
End If
```

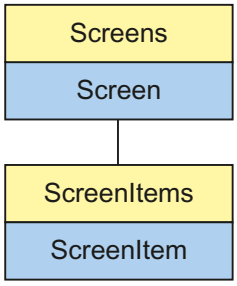
```
'=== Property "BorderStyle" only exists for a Windows-Slider
ScreenItem.BorderStyle = 1
If (Err.Number <> 0) Then
HMIRuntime.Trace(ScreenItem.ObjectName + ": no WinCC Slider Control" + vbCrlf)
Err.Clear
End If
End If
Next
On Error GoTo 0  ' deactivating error handling
End Sub
```

Siehe auch

PictureThumb-Eigenschaft (Seite 516)
BarFillColor-Eigenschaft (Seite 321)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Controls (Seite 226)
WithLabels-Eigenschaft (Seite 676)
WithAxes-Eigenschaft (Seite 676)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
TickStyle-Eigenschaft (Seite 578)
ThumbBackColor-Eigenschaft (Seite 575)
ShowThumb-Eigenschaft (Seite 554)
ShowPosition-Eigenschaft (Seite 551)
ShowBar-Eigenschaft (Seite 549)
RangeMin-Eigenschaft (Seite 526)
RangeMax-Eigenschaft (Seite 525)
Position-Eigenschaft (Seite 518)
PictureBack-Eigenschaft (Seite 514)
Parent-Eigenschaft (Seite 506)
OuterBevelWidth-Eigenschaft (Seite 505)
OuterBevelStyle-Eigenschaft (Seite 504)
ObjectName-Eigenschaft (Seite 490)
Object-Eigenschaft (Seite 489)
LocaleID-Eigenschaft (Seite 460)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
LabelColor-Eigenschaft (Seite 433)
InnerBevelWidth-Eigenschaft (Seite 430)
InnerBevelStyle-Eigenschaft (Seite 429)
InnerBevelOffset-Eigenschaft (Seite 429)
Height-Eigenschaft (Seite 421)
ForeColor-Eigenschaft (Seite 412)
Font-Eigenschaft (Seite 410)
Font-Eigenschaft (Vor WinCC V7) (Seite 408)
FocusWidth-Eigenschaft (Seite 407)
FocusColor-Eigenschaft (Seite 407)

WinCC UserArchiveControl

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "WinCC UserArchiveControl" ab WinCC V7.0.

Typkennzeichnung in VBS

HMIUserArchiveControl

Verfügbare Auflistungs-Objekte

Column (Seite 230)	StatusbarElement (Seite 239)
Row (Seite 235)	ToolbarButton (Seite 241)

Verfügbare Methoden in VBS

Activate	GetRow (Seite 705)	MoveToFirst	ServerImport
ActivateDynamic	GetRowCollection (Seite 707)	MoveToLast	ShowHelp
CopyRows	GetSelectedRow (Seite 713)	MoveToNext	ShowPropertyDialog
CutRows	GetSelectedRows (Seite 714)	MoveToPrevious	ShowSelectArchive
DeactivateDynamic	GetStatusbarElement	PasteRows	ShowSelection
CutRows	GetStatusbarElementCollection	Print	ShowSelectTimeBase
Export	GetToolbarButton	ReadTags	ShowSort
GetColumn	GetToolbarButtonCollection	ServerExport	WriteTags
GetColumnCollection			

Verfügbare Eigenschaften in VBS

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben. Z. B. wird bei der Verwendung von "colobj.ColumnName" der Auflistungsname "Column" weggelassen: "colobj.Name".

Beachten Sie, dass für die WinCC-Controls auch Eigenschaften verfügbar sind, die sich funktional als Methoden auswirken. Diese Eigenschaften zeichnen sich durch entsprechende Namen aus wie z. B. "Add", "Remove" oder "Rename".

ArchiveName	ColumnShowIcon	RowTitles	StatusbarUseBackColor
ArchiveType	ColumnShowTitleIcon	RTPersistence	StatusbarVisible
AutoCompleteColumns	ColumnSort	RTPersistencePasswordLevel	TableColor
AutoCompleteRows	ColumnSortIndex	RTPersistenceType	TableColor2
AutoSelectionColors	ColumnStartValue	SelectArchiveName	TableForeColor
AutoSelectionRectColor	ColumnStringLength	SelectedCellColor	TableForeColor2
BackColor	ColumnTimeFormat	SelectedCellForeColor	TimeBase
BorderColor	ColumnTitleAlign	SelectedRowColor	TitleColor
BorderWidth	ColumnTitles	SelectedRowForeColor	TitleCut
Caption	ColumnType	SelectedTitleColor	TitleDarkShadowColor
CellCut	ColumnVisible	SelectedTitleForeColor	TitleForeColor
CellSpaceBottom	ColumnWriteAccess	SelectionColoring	TitleGridLineColor
CellSpaceLeft	EnableDelete	SelectionRect	TitleLightShadowColor
CellSpaceRight	EnableEdit	SelectionRectColor	TitleSort
CellSpaceTop	EnableInsert	SelectionRectWidth	TitleStyle
Closeable	ExportDirectoryChangeable	SelectionType	ToolbarAlignment
ColumnAlias	ExportDirectoryname	ShowSortButton	ToolbarBackColor
ColumnAlign	ExportFileExtension	ShowSortIcon	ToolbarButtonActive
ColumnAutoPrecisions	ExportFilename	ShowSortIndex	ToolbarButtonAdd
ColumnCaption	ExportFilenameChangeable	ShowTitle	ToolbarButtonBeginGroup
ColumnCount	ExportFormatGuid	Sizeable	ToolbarButtonClick
ColumnDateFormat	ExportFormatName	SkinName	ToolbarButtonCount
ColumnDMVarName	ExportParameters	SortSequence	ToolbarButtonEnabled
ColumnExponentialFormat	ExportSelection	StatusbarBackColor	ToolbarButtonHotKey
ColumnFlagNotNull	ExportShowDialog	StatusbarElementAdd	ToolbarButtonId
ColumnFlagUnique	ExportXML	StatusbarElementAutoSize	ToolbarButtonIndex
ColumnHideText	FilterSQL	StatusbarElementCount	ToolbarButtonLocked
ColumnHideTitleText	Font	StatusbarElementIconId	ToolbarButtonName
ColumnIndex	GridLineColor	StatusbarElementId	ToolbarButtonPasswordLevel
ColumnLeadingZeros	GridLineWidth	StatusbarElementIndex	ToolbarButtonRemove
ColumnLength	HorizontalGridLines	StatusbarElementName	ToolbarButtonRename
ColumnMaxValue	IconSpace	StatusbarElementRemove	ToolbarButtonRepos
ColumnMinValue	LineColor	StatusbarElementRename	ToolbarButtonTooltipText
ColumnName	LineWidth	StatusbarElementRepos	ToolbarButtonUserDefined
ColumnPosition (Seite 363)	Moveable	StatusbarElementText	ToolbarButtonVisible
ColumnPrecisions	PrintJobName	StatusbarElementTooltipText	ToolbarShowTooltips
ColumnReadAccess	RowCellCount (Seite 530)	StatusbarElementUserDefined	ToolbarUseBackColor

ColumnReadonly	RowCellText (Seite 530)	StatusbarElementVisible	ToolbarUseHotKeys
ColumnRepos	RowCount (Seite 531)	StatusbarElementWidth	ToolbarVisible
ColumnResize	RowNumber (Seite 531)	StatusbarFontColor	UseSelectedTitleColor
ColumnScrollbar	RowScrollbar	StatusbarShowTooltips	UseTableColor2
ColumnShowDate	RowTitleAlign	StatusbarText	VerticalGridLines

Beispiel

In einem WinCC UserArchiveControl wird ein Anwenderarchiv angezeigt.

Folgende Aktionen werden über Skript veranlasst:

- Daten selektieren
- Daten exportieren
- Tabelle ausdrucken

Voraussetzung

- Im Graphics Designer ist ein "WinCC UserArchiveControl" mit dem Namen "Control1" in ein Prozessbild eingefügt.
- Im Graphics Designer ist ein Button eingefügt. Sie haben für den Button das Ereignis "Mausklick" projektiert, mit einer VBS-Aktion und folgendem Skript.
- Sie haben in Ihrem Projekt bereits ein Anwenderarchiv projektiert. Oder verwenden Sie das Demo-Projekt, aus dem Sie ein Anwenderarchiv verwenden können.

```
'VBS365
Sub OnClick(ByVal Item)
Dim objUAControl
Dim objColumn
Dim coll
Dim field
' create reference to UserArchivControl
Set objUAControl = ScreenItems("Control1")
' Select user archiv and general column properties
objUAControl.SelectArchiveName = True
objUAControl.ColumnResize = False
objUAControl.ColumnTitleAlign = 1
' properties for ID column
Set objColumn = objUAControl.GetColumn("ID")
objColumn.Length = 2
objColumn.Align = 0
' Select data
objUAControl.FilterSQL = "ID >=3"
'export the content as a CSV-file in the "ua" directory of the project folder
objUAControl.ServerExport
' print the control
objUAControl.PrintJobName = "UserArchiveControl - Table"
objUAControl.Print
End Sub
```

Hinweis

Weitere Beispiele zur Verwendung der Eigenschaften und Methoden finden Sie in den Beschreibungen der Get-Methoden der Controls und unter "Beispiele zu VBScript/Beispiele in WinCC/Dynamisieren der Controls".

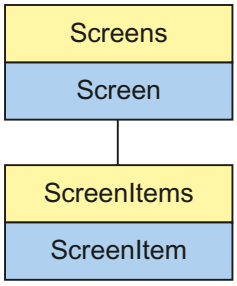
Siehe auch

Controls (Seite 226)

Controls vor WinCC V7

WinCC Alarm Control (vor WinCC V7)

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "WinCC Alarm Control".

Typkennzeichnung in VBS

HMIMessageView

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Control1" um 10 Pixel nach rechts verschoben:

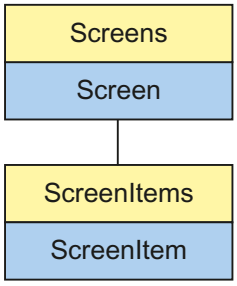
```
'VBS54
Dim objControl
Set objControl = ScreenItems("Control1")
objControl.Left = objControl.Left + 10
```

Siehe auch

ProjectPath-Eigenschaft (Seite 522)
BackColor-Eigenschaft (Seite 316)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Controls (Seite 226)
WindowType-Eigenschaft (Seite 675)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolbarButtons-Eigenschaft (Seite 614)
Titleline-Eigenschaft (Seite 603)
TitleCut-Eigenschaft (vor WinCC V7) (Seite 602)
StatusbarPanes-Eigenschaft (Seite 569)
ServerNames-Eigenschaft (vor WinCC V7) (Seite 548)
SelectionType-Eigenschaft (vor WinCC V7) (Seite 545)
SelectionRectWidth-Eigenschaft (vor WinCC V7) (Seite 544)
SelectionRectColor-Eigenschaft (vor WinCC V7) (Seite 544)
SelectionMode-Eigenschaft (Seite 543)
PersistentRTPermission-Eigenschaft (Seite 511)
PersistentRTCSPermission-Eigenschaft (Seite 510)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Object-Eigenschaft (Seite 489)
MsgFilterSQL-Eigenschaft (vor WinCC V7) (Seite 486)
MsgCtrlFlags-Eigenschaft (Seite 486)
LineTitle-Eigenschaft (Seite 458)
LineHeight-Eigenschaft (Seite 458)
LineFont-Eigenschaft (Seite 457)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
HeaderSort-Eigenschaft (Seite 420)
GridLineVert-Eigenschaft (Seite 419)
GridLineHorz-Eigenschaft (Seite 417)
Font-Eigenschaft (Vor WinCC V7) (Seite 408)
Enabled-Eigenschaft (Seite 386)
ColWidth-Eigenschaft (Seite 369)

WinCC Function Trend Control (vor WinCC V7)

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "WinCC Function Trend Control.

Typkennzeichnung in VBS

HMIFunctionTrendView

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Control1" um 13 Pixel nach rechts verschoben:

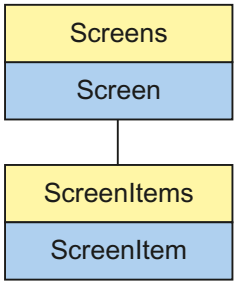
```
'VBS57
Dim objControl
Set objControl = ScreenItems("Control1")
objControl.Left = objControl.Left + 13
```

Siehe auch

Top-Eigenschaft (Seite 618)
ScalingTypeY-Eigenschaft (Seite 537)
Layer-Objekt (Seite 129)
DesiredCurveSourceUAArchive-Eigenschaft (Seite 382)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Controls (Seite 226)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
UpperLimitValue-Eigenschaft (Seite 650)
UpperLimit-Eigenschaft (Seite 649)
UpperLimitColor-Eigenschaft (Seite 649)
Type-Eigenschaft (Seite 642)
ToolbarHotKeys-Eigenschaft (Seite 615)
ToolbarButtons-Eigenschaft (Seite 614)
ToolbarAlignment-Eigenschaft (vor WinCC V7) (Seite 606)
Titleline-Eigenschaft (Seite 603)
TimeZone-Eigenschaft (Seite 601)
TimeAxisX-Eigenschaft (Seite 585)
TagProviderClsid-Eigenschaft (Seite 573)
SourceUAColumnY-Eigenschaft (Seite 563)
SourceUAColumnX-Eigenschaft (Seite 562)
SourceUAArchiveStartID-Eigenschaft (Seite 562)
SourceUAArchive-Eigenschaft (Seite 561)
SourceTimeRange-Eigenschaft (Seite 561)
SourceTagProviderDataY-Eigenschaft (Seite 561)
SourceTagProviderDataX-Eigenschaft (Seite 560)
SourceTagNameY-Eigenschaft (Seite 560)
SourceTagNameX-Eigenschaft (Seite 560)
SourceNumberOfValues-Eigenschaft (Seite 559)
SourceNumberOfUAValues-Eigenschaft (Seite 559)
SourceEndTime-Eigenschaft (Seite 559)
SourceBeginTime-Eigenschaft (Seite 558)
ShowValuesExponentialY-Eigenschaft (Seite 555)
ShowValuesExponentialX-Eigenschaft (Seite 555)
ShowRulerImmediately-Eigenschaft (Seite 551)
ScalingTypeX-Eigenschaft (Seite 536)
RulerPrecisionY-Eigenschaft (Seite 534)

WinCC Online Table Control (vor WinCC V7)

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "WinCC Online Table Control".

Typkennzeichnung in VBS

HMITableView

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Control1" um 15 Pixel nach rechts verschoben:

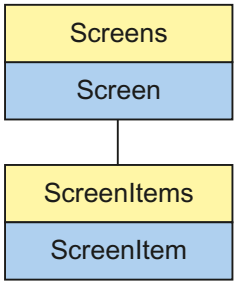
```
'VBS59
Dim objControl
Set objControl = ScreenItems("Control1")
objControl.Left = objControl.Left + 15
```

Siehe auch

TimeOverlap-Eigenschaft (Seite 597)
ItemVisible-Eigenschaft (Seite 433)
PrintBackgroundColor-Eigenschaft (Seite 521)
Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Controls (Seite 226)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Variable-Eigenschaft (Seite 670)
ValueColumnAlignment-Eigenschaft (Seite 663)
UpperLimitValue-Eigenschaft (Seite 650)
UpperLimit-Eigenschaft (Seite 649)
UpperLimitColor-Eigenschaft (Seite 649)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolbarHotKeys-Eigenschaft (Seite 615)
Toolbar-Eigenschaft (Seite 605)
ToolbarButtons-Eigenschaft (Seite 614)
ToolbarAlignment-Eigenschaft (vor WinCC V7) (Seite 606)
Titleline-Eigenschaft (Seite 603)
TimeZone-Eigenschaft (Seite 601)
TimeRangeFactor-Eigenschaft (Seite 598)
TimeRange-Eigenschaft (Seite 598)
TimeRangeBase-Eigenschaft (Seite 598)
TimeOverlapColor-Eigenschaft (Seite 597)
TimeJump-Eigenschaft (Seite 596)
TimeJumpColor-Eigenschaft (Seite 596)
TimeFormat-Eigenschaft (Seite 595)
TimeColumnAlignment-Eigenschaft (Seite 587)
Statusbar-Eigenschaft (Seite 564)
PrintJob-Eigenschaft (Seite 521)
Precisions-Eigenschaft (Seite 519)
PersistentRTPermission-Eigenschaft (Seite 511)
PersistentRT-Eigenschaft (Seite 510)
PersistentRTCSPermission-Eigenschaft (Seite 510)
PersistentRTCS-Eigenschaft (Seite 510)
Parent-Eigenschaft (Seite 506)
Online-Eigenschaft (vor WinCC V7) (Seite 494)

WinCC Online Trend Control (vor WinCC V7)

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "WinCC Online Trend Control".

Typkennzeichnung in VBS

HMITrendView

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Control1" um 16 Pixel nach rechts verschoben:

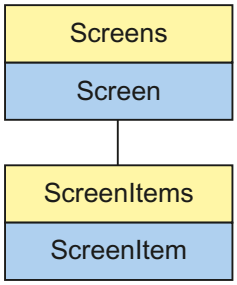
```
'VBS60
Dim objControl
Set objControl = ScreenItems("Control1")
objControl.Left = objControl.Left + 16
```

Siehe auch

Eigenschaften (Seite 295)
TimeAxis-Eigenschaft (Seite 578)
LowerLimitColor-Eigenschaft (Seite 465)
Caption-Eigenschaft (Seite 342)
Activate-Methode (Seite 686)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Controls (Seite 226)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
UpperLimitValue-Eigenschaft (Seite 650)
UpperLimit-Eigenschaft (Seite 649)
UpperLimitColor-Eigenschaft (Seite 649)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolbarHotKeys-Eigenschaft (Seite 615)
Toolbar-Eigenschaft (Seite 605)
ToolbarButtons-Eigenschaft (Seite 614)
ToolbarAlignment-Eigenschaft (vor WinCC V7) (Seite 606)
Titleline-Eigenschaft (Seite 603)
TimeZone-Eigenschaft (Seite 601)
TimeRangeFactor-Eigenschaft (Seite 598)
TimeRange-Eigenschaft (Seite 598)
TimeRangeBase-Eigenschaft (Seite 598)
TimeOverlap-Eigenschaft (Seite 597)
TimeOverlapColor-Eigenschaft (Seite 597)
TimeJump-Eigenschaft (Seite 596)
TimeJumpColor-Eigenschaft (Seite 596)
TimeAxisFormat-Eigenschaft (Seite 581)
TagName-Eigenschaft (Seite 572)
Statusbar-Eigenschaft (Seite 564)
ShowRulerImmediately-Eigenschaft (Seite 551)
ServerData-Eigenschaft (Seite 546)
RulerPrecisions-Eigenschaft (Seite 533)
Replacement-Eigenschaft (Seite 528)
ReplacementColor-Eigenschaft (Seite 528)
RelayCurves-Eigenschaft (Seite 527)
ProviderClsid-Eigenschaft (Seite 522)
PrintJob-Eigenschaft (Seite 521)
Precisions-Eigenschaft (Seite 519)

1.14.3.7 Anwender-Objekt

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Anwender-Objekt".

Typkennzeichnung in VBS

HMIScreenModule

Verwendung

Anwenderspezifische Eigenschaften in einem Anwender-Objekt müssen in VBS über die Attributnamen angesprochen werden. Die Intellisense gilt ausschließlich für das Anwender-Objekt als Ganzes.

Der Attributname ist unter Eigenschaften des nach außen gelegten Eigenschaften zu finden (rechter Mausklick auf die Eigenschaft) und kann dort auch geändert werden.

Im folgenden Beispiel wird das Objekt mit dem Namen "CustomizedObject1" um 10 Pixel nach rechts verschoben:

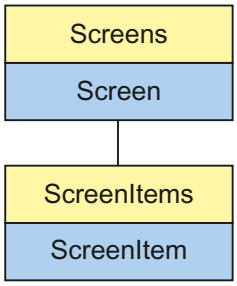
```
'VBS65
Dim objCustomObject
Set objCustomObject = ScreenItems("CustomizedObject1")
objCustomObject.Left = objCustomObject.Left + 10
```

Siehe auch

Activate-Methode (Seite 686)
Eigenschaften (Seite 295)
ScreenItems-Objekt (Auflistung) (Seite 138)
ScreenItem-Objekt (Seite 134)
Objekt-Typen des Objekts ScreenItem (Seite 152)
Width-Eigenschaft (Seite 673)
Visible-Eigenschaft (Seite 671)
Type-Eigenschaft (Seite 642)
Top-Eigenschaft (Seite 618)
ToolTipText-Eigenschaft (Seite 617)
Parent-Eigenschaft (Seite 506)
ObjectName-Eigenschaft (Seite 490)
Left-Eigenschaft (Seite 454)
Layer-Objekt (Seite 129)
Height-Eigenschaft (Seite 421)
Enabled-Eigenschaft (Seite 386)

1.14.3.8 Gruppe

Beschreibung



Objekt-Typ des ScreenItem-Objektes. Repräsentiert das Grafikobjekt "Gruppe".

Typkennzeichnung in VBS

HMIGroup

Verwendung

Im folgenden Beispiel wird das Objekt mit dem Namen "Group1" um 10 Pixel nach rechts verschoben:

```
'VBS66
Dim objGroup
Set objGroup = ScreenItems("Group1")
objGroup.Left = objGroup.Left + 10
```

Siehe auch

Eigenschaften (Seite 295)

Activate-Methode (Seite 686)

ScreenItems-Objekt (Auflistung) (Seite 138)

ScreenItem-Objekt (Seite 134)

Objekt-Typen des Objekts ScreenItem (Seite 152)

Width-Eigenschaft (Seite 673)

Visible-Eigenschaft (Seite 671)

Type-Eigenschaft (Seite 642)

Top-Eigenschaft (Seite 618)

ToolTipText-Eigenschaft (Seite 617)

Parent-Eigenschaft (Seite 506)

ObjectName-Eigenschaft (Seite 490)

Left-Eigenschaft (Seite 454)

Layer-Objekt (Seite 129)

Height-Eigenschaft (Seite 421)

Enabled-Eigenschaft (Seite 386)

1.14.4 Eigenschaften

1.14.4.1 Eigenschaften

Übersicht

Über die Eigenschaften der einzelnen Objekte können Sie Grafikobjekte und Variablen in Runtime gezielt verändern, z.B. auf Mausklick ein Bedienelement freischalten, oder bei Änderung eines Variablenwertes einen Farbumschlag auslösen.

Eigenschaften an Grafikobjekten können Sie über folgende Syntax ansprechen:

```
'VBS191
Dim obj
Set obj = ScreenItems("object1")
obj.property = Wert
```

Im folgenden Beispiel wird das Objekt mit dem Namen "control1" um 10 Pixel nach rechts verschoben:

```
'VBS192
Dim obj
Set obj = ScreenItems("control1")
obj.Left = obj.Left + 10
```

1.14.4.2 A

Aa - Ad

AccessPath-Eigenschaft

Beschreibung

Gibt den Ablagepfad (mit Hierarchieangabe) eines Screen-Objektes (Bild) aus. Die Eigenschaft entspricht dem vollständigen Zugriffsschlüssel auf die Screens-Collections.

STRING (readonly)

Beispiel

Im folgenden Beispiel wird der Pfad des Bildes "ScreenWindow1" ausgegeben:

```
'VBS67
Dim objScreen
Set objScreen = HMIRuntime.Screens("ScreenWindow1")
MsgBox objScreen.AccessPath
```

Siehe auch

ScreenItem-Objekt (Seite 134)

Screens-Objekt (Auflistung) (Seite 143)

Activate-Eigenschaft (vor WinCC V7)

Beschreibung

Die darzustellenden Daten werden nur dann vom Archivserver angefordert, wenn dieses Attribut gesetzt ist. Um die Bildaufschlagszeiten zu verringern, sollte dieses Attribut nicht gesetzt sein und der Wert nur bei Bedarf dynamisch geändert werden.

Schreib-/Lesezugriff

Um die "Activate"-Eigenschaft von der "Activate"-Methode zu unterscheiden, wird die Eigenschaft über "Object" angesprochen.

Beispiel

```
Dim ctrl
Set ctrl = ScreenItems("Control")
ctrl.Object.activate = true
```

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

Activate-Eigenschaft

Activate

Die im Meldefenster darzustellenden Daten werden nur dann vom Melde-Server angefordert, wenn Sie dieses Attribut setzen. Um die Bildaufschlagszeiten zu verringern, sollten Sie dieses Attribut nicht setzen und den Wert bei Bedarf dynamisch ändern.

Das Attribut ist mit dem Namen **Activate** dynamisierbar. Der Datentyp ist BOOLEAN.

ActiveProject-Eigenschaft

Beschreibung

Liefert ein Objekt vom Typ "Project" zurück.

Siehe auch

Path-Eigenschaft (Seite 508)

Name-Eigenschaft (Seite 487)

Ellipsensegment (Seite 158)

HMIRuntime-Objekt (Seite 127)

ActiveScreen-Eigenschaft

Beschreibung

Liefert eine Referenz auf das Bild, welches das Objekt enthält, das aktuell den Fokus hat.

Verwendung

Sie verwenden "ActiveScreen", um in Runtime die Eigenschaften des Bildes anzusprechen, welches das aktuell fokussierte Objekt enthält.

Beispiel

Das folgende Beispiel weist den Namen des aktuellen Bildes der Variablen "strScrName" zu und gibt ihn in einer Meldung aus :

```
'VBS68
Dim strScrName
strScrName = HMIRuntime.ActiveScreen.Objectname
MsgBox strScrName
```

Siehe auch

Screen-Objekt (Seite 140)

HMIRuntime-Objekt (Seite 127)

ActiveScreenItem-Eigenschaft

Beschreibung

Liefert eine Referenz auf das Objekt, welches gerade den Fokus hat.

Verwendung

Sie verwenden "ActiveScreenItem", um in Runtime die Eigenschaften des Objektes anzusprechen, welches den Fokus hat.

Beispiel

Das folgende Beispiel gibt den Namen des Objekts aus, das den Fokus im Bild "ScreenWindow1" hat:

```
'VBS69
Dim objScreen
Set objScreen = HMIRuntime.Screens("ScreenWindow1")
MsgBox objScreen.ActiveScreenItem.ObjectName
```

Siehe auch

ScreenItem-Objekt (Seite 134)

HMIRuntime-Objekt (Seite 127)

Actualize-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert ein Spaltenpaar bzw. eine Kurve. "Actualize" legt fest, ob für dieses Spaltenpaar/diese Kurve eine statische oder eine dynamische Darstellung verwendet werden soll.

- 0: statische Darstellung
- -1: dynamische Darstellung

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

ActualPointLeft-Eigenschaft

Beschreibung

Legt die x-Koordinate des aktuellen Eckpunktes in Bezug auf den Bildursprung (links oben) fest oder gibt sie zurück. Jeder Eckpunkt wird über einen Index identifiziert, der sich aus der Anzahl ("PointCount") der vorhandenen Eckpunkte ableitet.

Eine Änderung des Wertes kann sich auf die Eigenschaften "Width" (Objektbreite) und "Left" (x-Koordinate der Objektposition) auswirken.

Siehe auch

Polygonzug (Seite 170)

Polygon (Seite 168)

ScreenItem-Objekt (Seite 134)

ActualPointTop-Eigenschaft

Beschreibung

Legt die y-Koordinate des aktuellen Eckpunktes in Bezug auf den Bildursprung (links oben) fest oder gibt sie zurück. Jeder Eckpunkt wird über einen Index identifiziert, der sich aus der Anzahl ("PointCount") der vorhandenen Eckpunkte ableitet.

Eine Änderung des Wertes kann sich auf die Eigenschaften "Height" (Objekthöhe) und "Top" (y-Koordinate der Position) auswirken.

Siehe auch

Polygonzug (Seite 170)

Polygon (Seite 168)

ScreenItem-Objekt (Seite 134)

AdaptBorder-Eigenschaft

Beschreibung

TRUE, wenn der Rahmen dynamisch an die Textgröße angepasst werden soll. BOOLEAN Schreib-Lese-Zugriff.

Für Textliste und EA-Feld: Nur Lese-Zugriff.

Siehe auch

Button (Seite 212)

Statischer Text (Seite 178)

Textliste (Seite 208)

Radio-Box (Seite 217)

Check-Box (Seite 215)

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

AdaptPicture-Eigenschaft

Beschreibung

Bestimmt, ob das in einem Bildfenster dargestellte Bild sich in Runtime an die Größe des Bildfensters anpasst oder nicht. Nur Lese-Zugriff.

TRUE, wenn sich das Bild an die Bildfenstergröße anpasst.

FALSE, wenn sich das Bild nicht an die Bildfenstergröße anpasst.

Siehe auch

Bildfenster (Seite 190)

ScreenItem-Objekt (Seite 134)

AdaptSize-Eigenschaft

Beschreibung

Bestimmt, ob das Bildfenster sich dem in ihm dargestellten Bild in Runtime anpasst oder nicht. Nur Lese-Zugriff.

TRUE, wenn sich die Bildfenstergröße an das Bild anpasst.

FALSE, wenn sich die Bildfenstergröße nicht an das Bild anpasst.

Siehe auch

Bildfenster (Seite 190)

ScreenItem-Objekt (Seite 134)

AdjustRuler-Eigenschaft

Beschreibung

Legt fest, ob das Linealfenster bei jedem Aufblenden dem Kurvenfenster angepasst wird.

TRUE, wenn Sie das Linealfenster verschieben und dann aus- und wieder einblenden, wird es wieder an der ursprünglichen Position mit der urspünglichen Größe dargestellt.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

Al - Ap

AlarmID-Eigenschaft

Beschreibung

Liefert die AlarmID des Alarm-Objekts zurück. Die AlarmID ist eindeutig und wird vom System vergeben.

AlarmID (readonly)

Siehe auch

Alarms-Objekt (Auflistung) (Seite 120)

AlarmHigh-Eigenschaft

Beschreibung

Legt den oberen Grenzwert fest, bei dem Alarm ausgelöst wird oder gibt ihn zurück.
Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft "TypeAlarmHigh" fest.
Die Eigenschaft "CheckAlarmHigh" legt fest, ob die Überwachung dieses Grenzwertes aktiviert ist.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

AlarmLogs-Eigenschaft

Beschreibung

Liefert ein Objekt vom Typ "AlarmLogs" zurück.

AlarmLogs (read-only)

Siehe auch

HMIRuntime-Objekt (Seite 127)

AlarmLow-Eigenschaft

Beschreibung

Legt den unteren Grenzwert fest, bei dem Alarm ausgelöst wird oder gibt ihn zurück.
Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft "TypeAlarmLow" fest.
Die Eigenschaft "CheckAlarmLow" legt fest, ob die Überwachung dieses Grenzwertes aktiviert ist.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

Alignment-Eigenschaft

Beschreibung

Legt die Darstellung der Skala (links/rechts oder oben/unten) abhängig von der Lage des BarGraph-Objektes fest oder gibt sie zurück. Die "Scaling"-Eigenschaft muss auf TRUE gesetzt sein, damit die Skala angezeigt wird.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

AlignmentLeft-Eigenschaft

Beschreibung

Legt die horizontale Ausrichtung des Textes fest oder gibt sie zurück. Wertebereich von 0 bis 2.

0 = links

1 = zentriert

2 = rechts

Siehe auch

Sammelanzeige (Seite 205)
Statischer Text (Seite 178)
Textliste (Seite 208)
Radio-Box (Seite 217)
Check-Box (Seite 215)
Button (Seite 212)
EA-Feld (Seite 195)
ScreenItem-Objekt (Seite 134)

AlignmentTop-Eigenschaft

Beschreibung

Legt die vertikale Ausrichtung des Textes fest oder gibt sie zurück. Wertebereich von 0 bis 2.
0 = oben
1 = zentriert
2 = unten

Siehe auch

Sammelanzeige (Seite 205)
Statischer Text (Seite 178)
Textliste (Seite 208)
Radio-Box (Seite 217)
Check-Box (Seite 215)
Button (Seite 212)
EA-Feld (Seite 195)
ScreenItem-Objekt (Seite 134)

AllowPersistence-Eigenschaft

Beschreibung

TRUE, wenn Einstellungen bezüglich der Persistenz möglich sind. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

AllowXAxisColor-Eigenschaft

Beschreibung

TRUE, wenn in Runtime die festgelegte Farbe der gemeinsamen X-Achse angezeigt wird. BOOLEAN Schreib-Lese-Zugriff.

AllServer-Eigenschaft (vor WinCC V7)

Beschreibung

Legt fest, dass die im Meldefenster darzustellenden Daten von allen an einem verteilten System beteiligten Servern, auf denen das Alarm Logging aktiviert ist, angefordert werden. Schreib-Lese-Zugriff.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

AllServer-Eigenschaft

Alle Server - AllServer

Legt fest, ob alle Server gewählt werden, deren Packages geladen wurden und auf denen "Alarm Logging Runtime" in der Anlaufliste aktiviert ist.

Wert	Erklärung
TRUE	Alle Server werden aktiviert.
FALSE	Nur der in "Serverauswahl" eingegebene Server wird aktiviert.

Das Attribut ist mit dem Namen **AllServer** dynamisierbar. Der Datentyp ist BOOLEAN.

Analog-Eigenschaft

Beschreibung

TRUE, wenn die Uhr als Analoguhr dargestellt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

AnchorRuler-Eigenschaft

Beschreibung

TRUE, wenn das Linealfenster fest mit dem Kurvenfenster verbunden ist. BOOLEAN Schreib-Lese-Zugriff.

AngleAlpha-Eigenschaft

Beschreibung

Legt den Tiefenwinkel a für den 3D-Effekt des "3DBarGraph"-Objektes fest oder gibt ihn zurück. Wertebereich in Grad von 0 bis 90.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

AngleBeta-Eigenschaft

Beschreibung

Legt den Tiefenwinkel b für den 3D-Effekt des "3DBarGraph"-Objektes fest oder gibt ihn zurück. Wertebereich in Grad von 0 bis 90.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

AngleMax-Eigenschaft

Beschreibung

Legt den Winkel auf der Skala fest, an dem die Skalenteilung endet, oder gibt ihn zurück. LONG Schreib-Lese-Zugriff.

Anfang und Ende der Skalenteilung werden mit den Attributen "AngleMin" und "AngleMax" in Winkelgraden beschrieben. Es gilt AngleMin< AngleMax.

Der Winkel 0 Grad liegt auf der rechten Seite des horizontalen Durchmessers der Skalenscheibe. Positive Winkelwerte werden im Uhrzeigersinn gezählt.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

AngleMin-Eigenschaft

Beschreibung

Legt den Winkel auf der Skala fest, an dem die Skalenteilung beginnt, oder gibt ihn zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

Application-Eigenschaft

Beschreibung

Gibt die Graphics-Designer-Anwendung zurück, wenn die Application-Eigenschaft ohne Objektkennzeichner verwendet wird. Wird die Application-Eigenschaft mit Objektkennzeichner verwendet, gibt sie ein Application-Objekt zurück, das die Anwendung darstellt, mit der das festgelegte Objekt erstellt wurde. Nur Lese-Zugriff.

Siehe auch

Applikationsfenster (Seite 185)

ScreenItem-Objekt (Seite 134)

ApplyProjectSettings-Eigenschaft

Projekteinstellungen übernehmen - ApplyProjectSettings

Legt fest, ob die Projekteinstellungen aus dem "Alarm Logging" übernommen werden.

Wert	Erklärung
TRUE	Die Option "Projekteinstellungen übernehmen" ist aktiviert. Die im "Alarm Logging" projektierten Meldeblöcke werden mit ihren Eigenschaften in das AlarmControl übernommen. Die Meldeblöcke werden mit diesen Eigenschaften im Meldefenster angezeigt.
FALSE	Die Option "Projekteinstellungen übernehmen" ist nicht aktiviert. Sie können Meldeblöcke hinzufügen bzw. entfernen oder die Eigenschaften ändern.

Das Attribut ist mit dem Namen **ApplyProjectSettings** dynamisierbar. Der Datentyp ist BOOLEAN.

Ar - Ax

Archive-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert ein Spaltenpaar. "Archive" legt das mit diesem Spaltenpaar verbundene Prozesswertarchiv fest. Die Angabe des Namens des Prozesswertarchives erfolgt in der Form: Servername::Archivname

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

ArchiveName-Eigenschaft

Name - ArchiveName

Legt fest, welches Anwenderarchiv oder welche Sicht angezeigt wird. Über die Schaltfläche öffnen Sie den Dialog "Package Browser" zum Auswählen des Archivs oder der Sicht.

Das Attribut ist mit dem Namen **ArchiveName** dynamisierbar. Der Datentyp ist STRING.

ArchiveType-Eigenschaft

Typ - ArchiveType

Gibt an, ob das gewählte Anwenderarchiv ein Archiv oder eine Sicht ist. Das Feld kann nicht bearbeitet werden.

Das Attribut ist mit dem Namen **ArchiveType** dynamisierbar. Der Datentyp ist LONG.

AspectRatio-Eigenschaft

AspectRatio

Legt bei Filmen fest, ob das Seitenverhältnis beibehalten wird.

Das Attribut ist mit dem Namen **AspectRatio** dynamisierbar. Der Datentyp ist BOOLEAN.

Assignments-Eigenschaft

Beschreibung

Eine Auflistung, welche die Zuordnungen zwischen dem Ausgabewert und dem tatsächlich auszugebenden Ausgabetext enthält.
Die Zuordnungen sind abhängig von der eingestellten Listenart. Die Listenart legen Sie mit der ListType-Eigenschaft fest.

Nur Lese-Zugriff.

Siehe auch

Textliste (Seite 208)

ScreenItem-Objekt (Seite 134)

AssumeOnExit-Eigenschaft

Beschreibung

TRUE, wenn der eingegebene Text nach dem Verlassen des Eingabefeldes übernommen wird (z.B. mit der Taste <Tab> oder Mausklick). BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

EA-Feld (Seite 195)

Textliste (Seite 208)

ScreenItem-Objekt (Seite 134)

AssumeOnFull-Eigenschaft

Beschreibung

TRUE, wenn das Eingabefeld nach vollständiger Eingabe (die vorgegebene Anzahl an Zeichen wurde eingegeben) automatisch verlassen und die Eingabe dabei übernommen wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

AutoCompleteColumns-Eigenschaft

Leerspalten anzeigen - AutoCompleteColumns

Legt fest, ob Leerspalten angezeigt werden, wenn das Control breiter ist als die projektierten Spalten.

Wert	Erklärung
TRUE	Die Leerspalten werden angezeigt.
FALSE	Die Leerspalten werden nicht angezeigt.

Das Attribut ist mit dem Namen **AutoCompleteColumns** dynamisierbar. Der Datentyp ist BOOLEAN.

AutoCompleteRows-Eigenschaft

Leerzeilen anzeigen - AutoCompleteRows

Legt fest, ob Leerzeilen angezeigt werden, wenn das Control länger ist als die Anzahl der projektierten Zeilen.

Wert	Erklärung
TRUE	Die Leerzeilen werden angezeigt.
FALSE	Die Leerzeilen werden nicht angezeigt.

Das Attribut ist mit dem Namen **AutoCompleteRows** dynamisierbar. Der Datentyp ist BOOLEAN.

AutoPosition-Eigenschaft

Automatisch positionieren- AutoPosition

Legt fest, ob das RulerControl genau unterhalb des Quell-Controls platziert wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Erklärung
TRUE	Das RulerControl wird genau unterhalb des Quell-Controls platziert.
FALSE	Das RulerControl wird so angezeigt, wie sie die Position des Controls projektiert haben.

Das Attribut ist mit dem Namen **AutoPosition** dynamisierbar. Der Datentyp ist BOOLEAN.

Autorange-Eigenschaft

Beschreibung

TRUE, wenn der Wertebereich der y-Achse automatisch ermittelt oder über die Attribute "BeginValue" und "EndValue" festgelegt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

AutorangeX-Eigenschaft

Beschreibung

TRUE, wenn der Wertebereich der X-Achse automatisch ermittelt wird. FALSE, wenn er über die Attribute "BeginX" und "EndX" festgelegt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

AutorangeY-Eigenschaft

Beschreibung

TRUE, wenn der Wertebereich der Y-Achse automatisch ermittelt wird. FALSE, wenn er über die Attribute "BeginY" und "EndY" festgelegt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

AutoScroll-Eigenschaft (vor WinCC V7)

Beschreibung

Legt fest, wie sich das Meldefenster beim Auftreten neuer Meldungen verhält. BOOLEAN Schreib-Lese-Zugriff:

TRUE: Eine neu auftretende Meldung wird an die im Meldefenster dargestellte Liste angehängt und automatisch selektiert. Der sichtbare Bereich des Meldefensters wird gegebenenfalls verschoben.

FALSE: Eine neu auftretende Meldung wird nicht selektiert. Der sichtbare Bereich des Meldefensters wird nicht verändert.

Die gezielte Selektion von Meldezeilen ist nur möglich, wenn "AutoScroll" nicht aktiviert ist.

Die Eigenschaft "AutoScroll" wird deaktiviert, wenn das Attribut "MsgCtrlFlag" = "-1" gesetzt wird. Dies bedeutet, dass die aktuellste Meldung im Meldefenster oben angezeigt wird.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

AutoScroll-Eigenschaft

Auto Scrolling - AutoScroll

Legt fest, wie sich das Meldefenster beim Auftreten neuer Meldungen verhält.

Die Selektion von Meldezeilen ist nur möglich, wenn "Auto Scrolling" nicht aktiviert ist.

Wert	Erkärung
TRUE	Wenn "Autoscroll" aktiviert ist, wird eine neu auftretende Meldung an die im Meldefenster dargestellte Liste angehängt und automatisch selektiert. Der sichtbare Bereich des Meldefensters wird gegebenenfalls verschoben.
FALSE	Wenn "Autoscroll" nicht aktiviert ist, wird eine neu auftretende Meldung nicht selektiert. Der sichtbare Bereich des Meldefensters wird nicht verändert.

Das Attribut ist mit dem Namen **AutoScroll** dynamisierbar. Der Datentyp ist BOOLEAN.

AutoSelectionColors-Eigenschaft

Automatische Farbgebung Markierung - AutoSelectionColors

Legt fest, ob die Markierungsfarben für Zellen und Zeilen mit den vom System festgelegten Farben dargestellt werden.

Wert	Erklärung
TRUE	Die Systemfarben werden verwendet.
FALSE	Die benutzerdefinierten Farben werden verwendet.

Das Attribut ist mit dem Namen **AutoSelectionColors** dynamisierbar. Der Datentyp ist BOOLEAN.

AutoSelectionRectColor-Eigenschaft

Automatische Farbgebung Auswahlrahmen - AutoSelectionRectColor

Legt fest, ob der Auswahlrahmen mit der vom System festgelegten Farbe dargestellt wird.

Wert	Erklärung
TRUE	Die Systemfarbe wird verwendet.
FALSE	Die benutzerdefinierte Farbe wird verwendet.

Das Attribut ist mit dem Namen **AutoSelectionRectColors** dynamisierbar. Der Datentyp ist BOOLEAN.

AutoShow-Eigenschaft

Automatisch ein-/ausblenden - AutoShow

Legt fest, ob das RulerControl automatisch eingeblendet wird, wenn Sie im Quell-Control die Tastenfunktionen für Lineal, Statistikbereich oder Statistik ausgewählt haben.

Wenn Sie Lineal, Statistikbereich oder Statistik nicht mehr verwenden, wird das RulerControl wieder ausgeblendet.

Wert	Erklärung
TRUE	Das RulerControl wird automatisch eingeblendet.
FALSE	Das RulerControl wird nicht automatisch eingeblendet.

Das Attribut ist mit dem Namen **AutoShow** dynamisierbar. Der Datentyp ist BOOLEAN.

AutoSize-Eigenschaft

Beschreibung

Legt die Größenanpassung des Objekts fest oder gibt sie zurück. Folgende Werte sind einstellbar:

- 0: Keine Größenanpassung.
- 1: Das Bild (Eigenschaften "PictureSelected", "PictureUnselected") wird an die Schaltfläche angepasst.
- 2: Die Schaltfläche wird an das Bild (Eigenschaften "PictureSelected", "PictureUnselected") angepasst.

Siehe auch

WinCC Push Button Control (Seite 271)

ScreenItem-Objekt (Seite 134)

Autostart-Eigenschaft

Autostart

Legt bei Filmen fest, ob die Wiedergabe automatisch beginnt.

Das Attribut ist mit dem Namen **Autostart** dynamisierbar. Der Datentyp ist BOOLEAN.

Average-Eigenschaft

Average

TRUE, wenn ein Mittelwert aus den letzten 10 Werten gebildet wird. Damit sich ein neuer Mittelwert bildet, muss sich ein Wert ändern. Der Mittelwert wird bei einem Bildwechsel zurückgesetzt. Wenn z. B. nach dem Bildwechsel nur ein Wert existiert, bildet sich folgender Mittelwert: (5+0+0+0+0+0+0+0+0+0)/10=0,5.

BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

Axe-Eigenschaft

Beschreibung

Definiert die Lage des 3D-Balkens im Koordinatensystem oder gibt den Wert zurück. Wertebereich von 0 bis 2.

0: Der 3D-Balken wird auf der X-Achse dargestellt.

1: Der 3D-Balken wird auf der Y-Achse dargestellt.

2: Der 3D-Balken wird auf der Z-Achse dargestellt.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

AxisSection-Eigenschaft

Beschreibung

Legt den Abstand zweier langer Achsabschnitte fest oder gibt ihn zurück. Die Angabe des Abstandes erfolgt dabei in Skaleneinheiten und ist abhängig von den projektierten Minimal- und Maximalwerten.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

1.14.4.3 B

Ba

BackBorderWidth-Eigenschaft

Beschreibung

Legt die Breite des 3D-Rahmens in Pixel fest oder gibt sie zurück. Der Wert für die Breite ist abhängig von der Größe des Objektes.

Siehe auch

ScreenItem-Objekt (Seite 134)

Button (Seite 212)

Rundbutton (Seite 219)

Slider (Seite 221)

Sammelanzeige (Seite 205)

BackColor-Eigenschaft

Hintergrund - BackColor

Gibt die Hintergrundfarbe des Controls an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **BackColor** dynamisierbar. Der Datentyp ist LONG.

BackColor-Eigenschaft

Hintergrundfarbe (BackColor)

Legt über den Dialog "Farbauswahl" die Hintergrundfarbe des Symbols fest. Die Hintergrundfarbe wird bei der Hintergrundart "Undurchsichtig" dargestellt.

Das Attribut ist mit dem Namen **BackColor** dynamisierbar. Der Datentyp ist LONG.

BackColor-Eigenschaft

Funktion

Legt die Hintergrundfarbe des Objektes fest oder gibt sie zurück.

Für Objekte mit Füllmuster wird die Hintergrundfarbe nicht angezeigt, wenn das Füllmuster "Transparent" eingestellt ist.

Besonderheit beim WinCC Slider-Control

Die Hintergrundfarbe wirkt sich nur aus, wenn das Objekt, zumindest teilweise, gefüllt ist.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an.

Beispiel:

RGB(200, 150, 100)

Beispiel

Das folgende Beispiel setzt den Hintergrund des Bildes "ScreenWindow1" auf Rot:

```
'VBS70
Dim objScreen
Set objScreen = HMIRuntime.Screens("ScreenWindow1")
objScreen.BackColor = RGB(255, 0, 0)
```

Siehe auch

Fillstyle-Eigenschaft (Seite 398)

FillColor-Eigenschaft (Seite 396)

ScreenItem-Objekt (Seite 134)

BackColor2-Eigenschaft

Beschreibung

Legt die Balkenfarbe für die Anzeige des aktuellen Werts fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

BackColor3-Eigenschaft

Beschreibung

Legt die Farbe des Balkenhintergrunds fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

ScreenItem-Objekt (Seite 134)

Balken (Seite 186)

BackColorBottom-Eigenschaft

Beschreibung

Legt die Farbe für den unteren/rechten Teil des Sliders fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Slider (Seite 221)

ScreenItem-Objekt (Seite 134)

BackColorTop-Eigenschaft

Beschreibung

Legt die Farbe für den oberen/linken Teil des Sliders fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Slider (Seite 221)

ScreenItem-Objekt (Seite 134)

BackFlashColorOff-Eigenschaft

Beschreibung

Legt die Farbe des Objekthintergrundes für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

ScreenItem-Objekt (Seite 134)

BackFlashColorOn-Eigenschaft

Beschreibung

Legt die Farbe des Objekthintergrundes für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

ScreenItem-Objekt (Seite 134)

Background-Eigenschaft

Beschreibung

TRUE, wenn der Hintergrund beim 3DBarGraph-Objekt sichtbar sein soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

BackgroundPicture-Eigenschaft

Beschreibung

Gibt den Bildnamen des Hintergrundbilds für die Skalenscheibe zurück. Nur Lese-Zugriff

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

BackPictureAlignment-Eigenschaft

Beschreibung

Legt die Darstellungsart des Hintergrundbildes im Prozessbild fest oder gibt sie zurück.

LONG Schreib-Lese-Zugriff.

BackPictureName-Eigenschaft

Beschreibung

Legt Pfad und Dateinamen des Hintergrundbildes im Prozessbild fest oder gibt diese zurück. LONG Schreib-Lese-Zugriff.

BackStyle-Eigenschaft

Beschreibung

WinCC Digital/Analog Clock

Legt die Hintergrundart der Analoguhr fest:

- 0: Der rechteckige Hintergrund der Uhr ist mit der angegebenen Hintergrundfarbe ausgefüllt.
- 1: Das runde Zifferblatt der Uhr ist mit der angegebenen Hintergrundfarbe ausgefüllt. Damit lässt sich eine runde Analoguhr darstellen.
- 2: Zifferblatt und rechteckiger Hintergrund sind transparent.

WinCC Gauge Control

Legt die Hintergrundart der Gauge fest:

- 0: Der rechteckige bzw. quadratische Hintergrund der Gauge ist mit der angegebenen Farbe Rahmenfarbe ausgefüllt. Die kreisförmige Skalenscheibe ist mit der angegebenen Farbe Hintergrundfarbe ausgefüllt.
- 1: Der rechteckige bzw. quadratische Hintergrund der Gauge ist transparent. Die kreisförmige Skalenscheibe ist mit der angegebenen Farbe Hintergrundfarbe ausgefüllt. Damit läßt sich eine kreisförmige Gauge darstellen.
- 2: Der rechteckige bzw. quadratische Hintergrund und die Skalenscheibe sind transparent.

WinCC Slider Control

Legt fest, ob der Hintergrund des Objekts transparent erscheint.

- 0: Der Hintergrund des Objekts ist nicht transparent
- 1: Der Hintergrund des Objekts ist transparent

HMI Symbol Library

Legt die Transparenz des Symbolhintergrunds fest. Schreib-Lese-Zugriff.

- 0: Der Hintergrund ist transparent und damit nicht sichtbar.
- 1: Der Hintergrund ist sichtbar, die Farbe des Hintergrunds wird durch das Attribut "Hintergrundfarbe" bestimmt.

Siehe auch

HMI Symbol Library (Seite 248)

WinCC Slider Control (Seite 278)

WinCC Gauge Control (Seite 261)

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

BarBackColor-Eigenschaft

Beschreibung

Legt die Hintergrundfarbe im Bereich des Schiebers fest. Der Bereich erstreckt sich von "RangeMin" bis "RangeMax".

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

BarDepth-Eigenschaft

Beschreibung

Legt die Balkentiefe in Pixel fest oder gibt sie zurück.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

BarFillColor-Eigenschaft

Beschreibung

Legt die Füllfarbe im Bereich des Schiebers fest. Der Bereich erstreckt sich vom "RangeMin" bis zur Position des Schiebers.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

BarHeight-Eigenschaft

Beschreibung

Legt die Balkenhöhe in Pixel fest oder gibt sie zurück.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

BarWidth-Eigenschaft

Beschreibung

Legt die Balkenbreite in Pixel fest oder gibt sie zurück.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

BasePicReferenced-Eigenschaft

Beschreibung

TRUE, wenn das zugeordnete Bild im Objekt Zustandsanzeige gespeichert werden soll. Sonst wird nur der zugehörige Objektverweis gespeichert. Nur Lese-Zugriff.

Siehe auch

Zustandsanzeige (Seite 210)

ScreenItem-Objekt (Seite 134)

BasePicTransColor-Eigenschaft

Beschreibung

Legt fest, welche Farbe des zugeordneten Bitmap-Objekts (.bmp, .dib) auf "transparent" gesetzt werden soll oder gibt die Farbe zurück. LONG Schreib-Lese-Zugriff.
Die Farbe wird nur dann auf "transparent" gesetzt, wenn die Eigenschaft "BasePicUseTransColor" den Wert TRUE hat.

Siehe auch

Zustandsanzeige (Seite 210)

ScreenItem-Objekt (Seite 134)

BasePicture-Eigenschaft

Beschreibung

Gibt das Grundbild für das Objekt Zustandsanzeige zurück. Nur Lese-Zugriff.
Das Bild (*.BMP oder *.DIB) muss sich im Verzeichnis "GraCS" des aktuellen Projekts befinden, damit es eingebunden werden kann.
Die Eigenschaft "BasePicReferenced" legt in diesem Zusammenhang fest, ob das Grundbild zusammen mit dem Objekt Zustandsanzeige gespeichert oder referenziert wird.

Siehe auch

Zustandsanzeige (Seite 210)

ScreenItem-Objekt (Seite 134)

BasePicUseTransColor-Eigenschaft

Beschreibung

TRUE, wenn die projektierte Farbe ("BasePicTransColor"-Eigenschaft) des Bitmap-Objektes auf "transparent" gesetzt werden soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Zustandsanzeige (Seite 210)

ScreenItem-Objekt (Seite 134)

BaseScreenName-Eigenschaft

Funktion

Legt das aktuelle Grundbild fest oder gibt es zurück.

STRING (Schreib-Lese-Zugriff)

Ein Bildwechsel wird über die Anweisung

```
HMIRuntime.BaseScreenName = (<Serverpräfix>::)<Neues Grundbild>
```

durchgeführt.

Beim Auslesen der Eigenschaft "BaseScreenName" wird immer nur der Bildname ohne Serverpräfix zurückgeliefert.

Hinweis

Bildnamen sind aus Kompatibilitätsgründen zu zukünftigen Versionen immer ohne die Dateierweiterung ".PDL" zu schreiben.

Beispiel

Das folgende Beispiel führt einen Bildwechsel auf "bild1.pdl" durch:

```
HMIRuntime.BaseScreenName = "bild1"
```

Siehe auch

ScreenItem-Objekt (Seite 134)

HMIRuntime-Objekt (Seite 127)

BaseY-Eigenschaft

Beschreibung

Legt den vertikalen Abstand des unteren Balkenrands zum oberen Rand des Objektfeldes fest oder gibt ihn zurück.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

BaseX-Eigenschaft

Beschreibung

Legt den horizontalen Abstand in Pixel des rechten Balkenrands zum linken Rand des Objektfeldes fest oder gibt ihn zurück.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Be - Bl

BeginTime-Eigenschaft

Beschreibung

WinCC Online Table Control

Die Eigenschaft "Index" referenziert ein Spaltenpaar. "BeginTime" legt den Startzeitpunkt der Darstellung dieses Spaltenpaares fest. Schreib-Lese-Zugriff.

WinCC Online Trend Control

Die Eigenschaft "Index" referenziert eine Kurve. "BeginTime" legt den Startzeitpunkt der Darstellung dieser Kurve fest. Ob die Angabe ausgewertet wird, ist abhängig von den Eigenschaften "TimeRange" und "CommonX".

Wenn Sie den Zeitbereich dynamisieren, verwenden Sie das Format "jjjj-mm-dd hh:mm:ss".

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

BeginValue-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. "BeginValue" legt die untere Grenze des darzustellenden Wertebereichs dieser Kurve fest. Ob die Angabe ausgewertet wird, ist abhängig von den Eigenschaften "Autorange" und "CommonY".

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

BeginX-Eigenschaft

Beschreibung

Legt die untere Grenze der X-Achse einer mit der Eigenschaft "Index" referenzierten Kurve fest oder gibt sie zurück. Ob die Angabe ausgewertet wird, ist abhängig von den Eigenschaften "AutorangeX" und "CommonX".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

BeginY-Eigenschaft

Beschreibung

Legt die untere Grenze der Y-Achse einer mit der Eigenschaft "Index" referenzierten Kurve fest oder gibt sie zurück. Ob die Angabe ausgewertet wird, ist abhängig von den Eigenschaften "AutorangeY" und "CommonY".

Siehe auch

ScreenItem-Objekt (Seite 134)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

BevelColorDown-Eigenschaft

Beschreibung

Legt die Farbe folgender Rahmenteile bei 3D-Darstellung des Rahmens fest:

- bei vertieftem Rahmen ("BevelStyle" = 1): oberer und linker Rahmenteil
- bei erhabenem Rahmen ("BevelStyle" = 2): unterer und rechter Rahmenteil

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

BevelColorUp-Eigenschaft

Beschreibung

Legt die Farbe folgender Rahmenteile bei 3D-Darstellung des Rahmens fest:

- bei vertieftem Rahmen ("BevelStyle" = 1): unterer und rechter Rahmenteil
- bei erhabenem Rahmen ("BevelStyle" = 2): oberer und linker Rahmenteil

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

BevelInner-Eigenschaft

Beschreibung

Legt die Erscheinungsform des inneren Teils des Objektrahmens fest oder gibt sie zurück. Schreib-Lese-Zugriff.

- 0: innerer Teil nicht vorhanden
- 1: "vertiefte" Erscheinungsform
- 2: "erhabene" Erscheinungsform
- 3: einheitlich grauer Rahmen
- 4 oder höher: einheitlich farbiger Rahmen, Rahmenfarbe = Hintergrundfarbe

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

BevelOuter-Eigenschaft

Beschreibung

Legt die Erscheinungsform des äußeren Teils des Objektrahmens fest oder gibt sie zurück. Schreib-Lese-Zugriff.

- 0: innerer Teil nicht vorhanden
- 1: "vertiefte" Erscheinungsform
- 2: "erhabene" Erscheinungsform
- 3: einheitlich grauer Rahmen
- 4 oder höher: einheitlich farbiger Rahmen, Rahmenfarbe = Hintergrundfarbe

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

BevelWidth-Eigenschaft

Beschreibung

Legt die Rahmenbreite für den inneren Rahmenteil (Innenrahmen) und für den äußeren Rahmenteil (Außenrahmen) in Pixel an oder gibt sie zurück. Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

BitNumber-Eigenschaft

Beschreibung

Legt das Bit fest, dessen Zustand sich ändern muss, um eine Wertänderung auszulösen oder gibt es zurück. Die verwendete Variable muss vom Typ BYTE, WORD oder DWORD sein.

Siehe auch

Textliste (Seite 208)

ScreenItem-Objekt (Seite 134)

BlinkColor-Eigenschaft

Beschreibung

Legt die Farbe des Symbols im Blinkbild fest. LONG Schreib-Lese-Zugriff.

Siehe auch

HMI Symbol Library (Seite 248)

ScreenItem-Objekt (Seite 134)

BlinkMode-Eigenschaft

Blinkmodus (BlinkMode)

Legt den Blinkmodus des Symbols in Runtime fest.

Folgende Einstellungen sind möglich:

Wert	Beschreibung	Erläuterung
0	Kein Blinken	Das Symbol blinkt nicht.
1	Unsichtbar	Das Symbol blinkt in der Hintergrundfarbe.
2	Schattiert	Das Symbol blinkt schattiert in der Vordergrundfarbe.
3	Massiv	Das Symbol blinkt in der Vordergrundfarbe.

Das Attribut ist mit dem Namen **BlinkMode** dynamisierbar. Der Datentyp ist LONG.

BlinkSpeed-Eigenschaft

Blinkgeschwindigkeit (BlinkSpeed)

Legt die Länge des Blinkintervalls des Symbols in Runtime fest.

Folgende Einstellungen sind möglich:

Wert	Beschreibung	Erläuterung
250	Schnell	Die Länge des Blinkintervalls ist 250 ms.
500	Mittel	Die Länge des Blinkintervalls ist 500 ms.
1000	Langsam	Die Länge des Blinkintervalls ist 1000 ms.

Das Attribut ist mit dem Namen **BlinkSpeed** dynamisierbar. Der Datentyp ist LONG.

BlockAlign-Eigensschaft

Ausrichtung Blöcke - BlockAlign

Legt fest, wie die Bezeichnung des Blocks in der Spaltenüberschrift ausgerichtet wird.

Folgende Einstellungen sind möglich:

Wert	Beschreibung	Erklärung
0	links	Die Bezeichnung des Blocks wird linksbündig angezeigt.
1	zentriert	Die Bezeichnung des Blocks wird zentriert angezeigt.
2	rechts	Die Bezeichnung des Blocks wird rechtsbündig angezeigt.

Das Attribut ist mit dem Namen **BlockAlign** dynamisierbar. Der Datentyp ist LONG.

BlockAutoPrecisions-Eigenschaft

Nachkommastellen Automatisch - BlockAutoPrecisions

Legt fest, ob die Anzahl der Nachkommastellen automatisch festgelegt wird.

Wert	Erklärung
TRUE	Anzahl der Nachkommastellen wird automatisch festgelegt. Der Wert im Feld "Nachkommastellen" ist nicht wirksam.
FALSE	Der Wert im Feld "Nachkommastellen" ist wirksam.

Das Attribut ist mit dem Namen **BlockAutoPrecisions** dynamisierbar. Der Datentyp ist BOOLEAN.

BlockCaption-Eigenschaft

Bezeichnung - BlockCaption

Legt für den ausgewählten Block die Bezeichnung der Spaltenüberschrift im Control fest.

Die Bezeichnung ist in allen Runtime-Sprachen wirksam.

Das Attribut ist mit dem Namen **BlockCaption** dynamisierbar. Der Datentyp ist STRING.

BlockCount-Eigenschaft

BlockCount

Gibt die Anzahl der Blöcke an, die als Spalten für das Control zur Verfügung stehen.

Das Attribut ist mit dem Namen **BlockCount** dynamisierbar. Der Datentyp ist LONG.

BlockDateFormat-Eigenschaft

Datumsformat - BlockDateFormat

Legt fest, welches Datumsformat zur Anzeige verwendet wird.

Folgende Datumsformate stehen zur Verfügung:

Wert	Erklärung
Automatisch	Das Datumsformat wird automatisch bestimmt.
dd.MM.yy	Tag.Monat.Jahr, z.B. 24.12.07.
dd.MM.yyyy	Tag.Monat.Jahr, z.B. 24.12.2007.
dd/MM/yy	Tag/Monat/Jahr, z.B. 24/12/07.
dd/MM/yyyy	Tag/Monat/Jahr, z.B. 24/12/2007.

Das Attribut ist mit dem Namen **BlockDateFormat** dynamisierbar. Der Datentyp ist STRING.

BlockExponentialFormat-Eigenschaft

Exponentialdarstellung - BlockExponentialFormat

Legt fest, ob die Werte des ausgewählten Blocks in Exponentialdarstellung angezeigt werden.

Wert	Erklärung
TRUE	Die Werte werden in Exponentialdarstellung angezeigt.
FALSE	Die Werte werden in Dezimaldarstellung angezeigt.

Das Attribut ist mit dem Namen **BlockExponentialFormat** dynamisierbar. Der Datentyp ist BOOLEAN.

BlockHideText-Eigenschaft

Inhalt als Text - BlockHideText

Legt fest, ob der Inhalt des ausgewählten Blocks als Text angezeigt wird.

Wert	Erklärung
TRUE	Der Inhalt wird nicht als Text angezeigt. Die Option ist nicht aktiviert
FALSE	Der Inhalt wird als Text angezeigt. Die Option ist aktiviert

Das Attribut ist mit dem Namen **BlockHideText** dynamisierbar. Der Datentyp ist BOOLEAN.

BlockHideTitleText-Eigenschaft

Überschrift als Text- BlockHideTitleText

Legt fest, ob die Überschrift des ausgewählten Blocks als Text angezeigt wird.

Wert	Erklärung
TRUE	Die Überschrift wird nicht als Text angezeigt. Die Option ist nicht aktiviert.
FALSE	Die Überschrift wird als Text angezeigt. Die Option ist aktiviert.

Das Attribut ist mit dem Namen **BlockHideTitleText** dynamisierbar. Der Datentyp ist BOOLEAN.

BlockId-Eigenschaft

BlockId

Festgelegte Zuordnung von Ident-Nummer und Block im WinCC RulerControl:

Wert	Beschreibung
0	Kein Block
1	Name
2	Index
3	Bezeichnung
4	Anzeige
5	Variablennname Y
6	Variablenname X
7	Y-Wert
8	X-Wert/Zeitstempel
9	Y-Wert (UG)
10	Zeitstempel (UG)
11	Y-Wert (OG)
12	Zeitstempel (OG)
13	Minimum
14	Minimum - Zeitstempel
15	Maximum
16	Maximum - Zeitstempel
17	Durchschnitt
18	Standardabweichung
19	Integral
20	Gewichteter Mittelwert
21	Dauer
22	Werteanzahl

Das Attribut ist mit dem Namen **BlockID** dynamisierbar. Der Datentyp ist LONG.

BlockIndex-Eigenschaft

BlockIndex

Referenziert einen Block. Unter Verwendung des Attributs können Sie einem bestimmten Block die Werte anderer Attribute zuweisen.

Gültige Werte für "BlockIndex" liegen zwischen 0 und "BlockCount" minus 1. Das Attribut "BlockCount" gibt die Anzahl der vorhandenen Blöcke an.

Das Attribut ist mit dem Namen **BlockIndex** dynamisierbar. Der Datentyp ist LONG.

BlockLength-Eigenschaft

Länge in Zeichen - BlockLength

Legt die Breite der Spalte für den gewählten Block fest.

Das Attribut ist mit dem Namen **BlockLength** dynamisierbar. Der Datentyp ist LONG.

BlockName-Eigenschaft

Objektname - BlockName

Zeigt den Namen für den gewählten Block an. Den Namen können Sie nicht ändern.

Das Attribut ist mit dem Namen **BlockName** dynamisierbar. Der Datentyp ist STRING.

BlockPrecisions-Eigenschaft

Nachkommastellen - BlockPrecisions

Legt die Anzahl der Nachkommastellen der Werte der ausgewählten Spalte fest. Sie können den Wert nur eingeben, wenn die Option "Automatisch" nicht aktiviert ist.

Das Attribut ist mit dem Namen **BlockPrecisions** dynamisierbar. Der Datentyp ist SHORT.

BlockShowDate-Eigenschaft

Datum anzeigen - BlockShowDate

Legt fest, ob der Block "Uhrzeit" mit Uhrzeit und Datum in einem Feld angezeigt wird.

Wert	Erklärung
TRUE	Datum und Uhrzeit werden angezeigt. Das Datumsformat wird im Feld "Datumsformat" festgelegt.
FALSE	Uhrzeit wird angezeigt.

Das Attribut ist mit dem Namen **BlockShowDate** dynamisierbar. Der Datentyp ist BOOLEAN.

BlockShowIcon-Eigenschaft

Inhalt als Symbol - BlockShowIcon

Legt fest, ob der Inhalt des ausgewählten Blocks als Symbol angezeigt wird.

Wert	Erklärung
TRUE	Der Inhalt wird als Symbol angezeigt.
FALSE	Der Inhalt wird nicht als Symbol angezeigt.

Das Attribut ist mit dem Namen **BlockShowIcon** dynamisierbar. Der Datentyp ist BOOLEAN.

BlockShowTitleIcon-Eigenschaft

Überschrift als Symbol- BlockShowTitleIcon

Legt fest, ob die Überschrift des ausgewählten Blocks als Symbol angezeigt wird.

Wert	Erklärung
TRUE	Die Überschrift wird als Symbol angezeigt.
FALSE	Die Überschrift wird nicht als Symbol angezeigt.

Das Attribut ist mit dem Namen **BlockShowTitleIcon** dynamisierbar. Der Datentyp ist BOOLEAN.

BlockTimeFormat-Eigenschaft

Zeitformat - BlockTimeFormat

Legt fest, welches Zeitformat zur Anzeige verwendet wird.

Folgende Zeitformate stehen zur Verfügung:

Wert	Erklärung
Automatisch	Das Zeitformat wird automatisch bestimmt.
HH:mm:ss.ms	Stunden:Minuten:Sekunden, z.B. 15:35:44.240.
hh:mm:ss tt	Stunden:Minuten:Sekunden AM/PM, z.B. 03:35:44 PM.
hh:mm:ss.ms tt	Stunden:Minuten:Sekunden.Millisekunden AM/PM, z.B. 03:35:44.240 PM.

Das Attribut ist mit dem Namen **BlockTimeFormat** dynamisierbar. Der Datentyp ist STRING.

BlockUseSourceFormat-Eigenschaft

von Quelle übernehmen - BlockUseSourceFormat

Legt fest, ob die Formatierungen vom verbundenen Control übernommen werden.

Wert	Erklärung
TRUE	Die Formatierungen werden vom verbundenen Control übernommen.
FALSE	Die hier projektierten Formatierungen werden übernommen.

Das Attribut ist mit dem Namen **BlockUseSouceFormat** dynamisierbar. Der Datentyp ist BOOLEAN.

Bo - Bu

BorderBackColor-Eigenschaft

Beschreibung

Legt die Hintergrundfarbe der Linie für das Objekt fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff. Die Hintergrundfarbe ist nur sichtbar, wenn die Eigenschaft "BorderWidth" > 0 gesetzt ist.

Siehe auch

ScreenItem-Objekt (Seite 134)

BorderColor-Eigenschaft

Beschreibung

Legt die Linienfarbe für das Objekt fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

ScreenItem-Objekt (Seite 134)

BorderColor-Eigenschaft

Rahmenfarbe - BorderColor

Gibt die Farbe des Rahmens an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **BorderColor** dynamisierbar. Der Datentyp ist LONG.

BorderColorBottom-Eigenschaft

Beschreibung

Legt die Randfarbe für den unteren/rechten Teil des Objekts fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

ScreenItem-Objekt (Seite 134)

Button (Seite 212)

Rundbutton (Seite 219)

BorderColorTop-Eigenschaft

Beschreibung

Legt die Randfarbe für den oberen/linkenTeil des Objekts fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Button (Seite 212)

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

BorderEndStyle-Eigenschaft

Beschreibung

Legt die Art der Linienenden des Objektes fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Polygonzug (Seite 170)

Linie (Seite 166)

ScreenItem-Objekt (Seite 134)

BorderFlashColorOff-Eigenschaft

Beschreibung

Legt die Linienfarbe des Objektes für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

ScreenItem-Objekt (Seite 134)

BorderFlashColorOn-Eigenschaft

Beschreibung

Legt die Linienfarbe des Objektes für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG-Schreib-Lese-Zugriff.

Siehe auch

ScreenItem-Objekt (Seite 134)

BorderStyle-Eigenschaft

Beschreibung

Legt die Linienart für das Objekt fest oder gibt sie zurück. Wertebereich von 0 bis 4.

0 = durchgezogene Linie

1 = gestrichelte Linie

2 = gepunktete Linie

3 = strichpunktierte Linie

4 = Strich-Punkt-Punkt-Linie

Siehe auch

ScreenItem-Objekt (Seite 134)

BorderWidth-Eigenschaft

Beschreibung

Legt die Linienstärke (in Pixel) für das Objekt fest oder gibt sie zurück.

WinCC Gauge Control:

Legt die Breite des mittleren Rahmenteils in Pixel an oder gibt sie zurück.

Der Objektrahmen ist aus drei Teilen zusammengesetzt. Der mittlere Teil des Objektrahmens wird durch die Eigenschaft "BorderWidth" beschrieben.

Die Farbe des mittleren Rahmenteils ist die Hintergrundfarbe.

Siehe auch

ScreenItem-Objekt (Seite 134)

BorderWidth-Eigenschaft

Rahmenstärke - BorderWidth

Legt die Linienstärke des Rahmens in Pixel fest.

Das Attribut ist mit dem Namen **BorderWidth** dynamisierbar. Der Datentyp ist LONG.

BottomConnectedConnectionPointIndex-Eigenschaft

Beschreibung

Gibt die Indexnummer des unteren Verbinderpunktes an oder setzt sie.

Long Schreib-Lese-Zugriff

Siehe auch

Verbinder (Seite 180)

ScreenItem-Objekt (Seite 134)

BottomConnectedObjectName-Eigenschaft

Beschreibung

Gibt den Objektnamen des Objektes an, das am unteren Verbinderpunkt angedockt ist oder setzt ihn.

Long Schreib-Lese-Zugriff

Siehe auch

Verbinder (Seite 180)

ScreenItem-Objekt (Seite 134)

BoxAlignment-Eigenschaft

Beschreibung

TRUE, wenn die Felder rechtsbündig angeordnet sind. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Radio-Box (Seite 217)
Check-Box (Seite 215)
ScreenItem-Objekt (Seite 134)

BoxCount-Eigenschaft

Beschreibung

Legt die Anzahl der Felder fest oder gibt sie zurück. Wertebereich von 0 bis 31.

Siehe auch

Radio-Box (Seite 217)
Check-Box (Seite 215)
ScreenItem-Objekt (Seite 134)

BoxType-Eigenschaft

Beschreibung

Legt den Feldtyp fest oder gibt ihn zurück. Wertebereich von 0 bis 2:

- 0: Ausgabe
- 1: Eingabe
- 2: E/A-Feld

Siehe auch

Textliste (Seite 208)
EA-Feld (Seite 195)
ScreenItem-Objekt (Seite 134)

ButtonColor-Eigenschaft

Beschreibung

Legt die Farbe des Schiebers fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Slider (Seite 221)

ScreenItem-Objekt (Seite 134)

ButtonCommand-Eigenschaft

Beschreibung

Bei der Änderung des Wertes von ButtonCommand wird an das WinCC Alarm Control eine Nachricht abgesetzt, um die Anzeige im Meldefenster zu beeinflussen.

Wert (hex); Wert (dez); Aufgerufene Funktion:

- 0x00000001; 1; Meldeliste
- 0x00000002; 2; Kurzzeitarchivliste
- 0x00000004; 4; Langzeitarchivliste
- 0x00200000; 2097152; Sperrliste
- 0x00000008; 8; Quittierung zentraler Melder
- 0x00000010; 16; Einzelquittierung
- 0x00000020; 32; Sammelquittierung
- 0x00000040; 64; Autoscroll
- 0x00000080; 128; Selektions-Dialog
- 0x00000100; 256; Sperr-Dialog
- 0x00000200; 512; Druck Meldeprotokoll
- 0x00000800; 2048; Not-Quittierung
- 0x00001000; 4096; Erste Meldung
- 0x00002000; 8192; Letzte Meldung
- 0x00004000; 16384; Nächste Meldung
- 0x00008000; 32768; Vorherige Meldung
- 0x00010000; 65536; Infotext-Dialog
- 0x00020000; 131072; Kommentar-Dialog
- 0x00040000; 262144; Loop in Alarm
- 0x00100000; 1048576; Druck aktuelle Ansicht
- 0x00400000; 4194304; Meldung sperren/freigeben
- 0x00800000; 8388608; Sortier-Dialog
- 0x01000000; 16777216; Zeitbasis-Dialog
- 0x02000000; 33554432; Hitliste
- 0x04000000; 67108864; Liste auszublendender Meldungen

- 0x08000000; 134217728; Meldung einblenden/ausblenden
- 0x10000000; 268435456; Anzeigeoptions-Dialog

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

Button1Width-Eigenschaft

Beschreibung

Legt die Breite des Buttons 1 in Pixel fest oder gibt sie zurück.
Wenn die Eigenschaft SameSize auf TRUE gesetzt ist, erhalten alle Buttons dieselbe Breite.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

Button2Width-Eigenschaft

Beschreibung

Legt die Breite des Buttons 2 in Pixel fest oder gibt sie zurück.
Wenn die Eigenschaft SameSize auf TRUE gesetzt ist, erhalten alle Buttons dieselbe Breite.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

Button3Width-Eigenschaft

Beschreibung

Legt die Breite des Buttons 3 in Pixel fest oder gibt sie zurück.
Wenn die Eigenschaft SameSize auf TRUE gesetzt ist, erhalten alle Buttons dieselbe Breite.

Siehe auch

ScreenItem-Objekt (Seite 134)

Sammelanzeige (Seite 205)

Button4Width-Eigenschaft

Beschreibung

Legt die Breite des Buttons 4 in Pixel fest oder gibt sie zurück.
Wenn die Eigenschaft SameSize auf TRUE gesetzt ist, erhalten alle Buttons dieselbe Breite.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

1.14.4.4 C

Ca - Cl

Caption-Eigenschaft

Beschreibung

Applikations- und Bildfenster

TRUE, wenn das Applikations- oder Bildfenster in Runtime eine Titelleiste hat. Nur Lese-Zugriff.

Die Caption-Eigenschaft muss auf TRUE gesetzt sein, wenn das Applikations- oder Bildfenster Schaltflächen zum Maximieren und Schließen besitzen soll.

Vor WinCC V7 Controls

Legt den Text fest, der als Beschriftung auf der Schaltfläche oder in der Titelleiste (Online Trend Control und Online Table Control) angezeigt wird oder gibt ihn zurück. Schreib-Lese-Zugriff.

Siehe auch

Controls (Seite 226)

Bildfenster (Seite 190)

Applikationsfenster (Seite 185)

ScreenItem-Objekt (Seite 134)

Caption-Eigenschaft

Text - Caption

Legt den Text der Fensterüberschrift fest.

Das Attribut ist mit dem Namen **Caption** dynamisierbar. Der Datentyp ist STRING.

CaptionColor-Eigenschaft

Beschreibung

Legt die Farbe der Instrumentenbeschriftung fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

ScreenItem-Objekt (Seite 134)

WinCC Gauge Control (Seite 261)

CaptionFont-Eigenschaft

Beschreibung

Gibt den Wert für Schriftart, Schriftstil und Schriftgröße sowie die Effekte "Unterstrichen" und "Durchgestrichen" für die Instrumentenbeschriftung zurück. Nur Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

CaptionOffset-Eigenschaft

Beschreibung

Legt den Abstand der Instrumentenbeschriftung in Bezug auf die Oberkante des Objekts fest oder gibt ihn zurück. Die Instrumentenbeschriftung ist nur längs des vertikalen Durchmessers der Skalenscheibe positionierbar. Der Wert des Attributs ist bezogen auf die Höhe des Objekts und wird gemessen von der Oberkante des Objekts zur Unterkante des Schriftzugs. Schreib-Lese-Zugriff.

Der Wertebereich ist 0 bis 1:

0: Die Unterkante des Schriftzugs liegt auf der oberen Begrenzung des Objekts. Der Text ist nicht mehr sichtbar, da er außerhalb des Objekts liegt.

1: Die Unterkante des Schriftzugs liegt auf der unteren Begrenzung des Objekts.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

CaptionText-Eigenschaft

Beschreibung

Legt den Fenstertitel fest, der in Runtime angezeigt wird oder gibt ihn zurück.
Die Caption-Eigenschaft muss dazu auf TRUE gesetzt sein.

Siehe auch

Bildfenster (Seite 190)

ScreenItem-Objekt (Seite 134)

CellCut-Eigenschaft (vor WinCC V7)

Beschreibung

TRUE, wenn die Inhalte der Felder einer Meldezeile bei zu kleiner Spaltenbreite abgekürzt werden. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

CellCut-Eigenschaft

Inhalte abkürzen - CellCut

Legt fest, ob die Inhalte der Zellen abgekürzt werden, wenn die Zellen zu schmal sind.

Wert	Erklärung
TRUE	Die Zelleninhalte werden abgekürzt.
FALSE	Die Zelleninhalte werden nicht abgekürzt.

Das Attribut ist mit dem Namen **CellCut** dynamisierbar. Der Datentyp ist BOOLEAN.

CellSpaceBottom-Eigenschaft

CellSpaceBottom

Legt den Abstand von unten fest, der in den Zellen der Tabelle verwendet wird.

Das Attribut ist mit dem Namen **CellSpaceBottom** dynamisierbar. Der Datentyp ist LONG.

CellSpaceLeft-Eingenschaft

CellSpaceLeft

Legt den Einzug Links fest, der in den Zellen der Tabelle verwendet wird.

Das Attribut ist mit dem Namen **CellSpaceLeft** dynamisierbar. Der Datentyp ist LONG.

CellSpaceRight-Eingenschaft

CellSpaceRight

Legt den Einzug Rechts fest, der in den Zellen der Tabelle verwendet wird.

Das Attribut ist mit dem Namen **CellSpaceRight** dynamisierbar. Der Datentyp ist LONG.

CellSpaceTop-Eingenschaftt

CellSpaceTop

Legt den Abstand von oben fest, der in den Zellen der Tabelle verwendet wird.

Das Attribut ist mit dem Namen **CellSpaceTop** dynamisierbar. Der Datentyp ist LONG.

CenterColor-Eigenschaft

Beschreibung

Legt die Farbe des kreisförmigen Skalenmittelpunkts (Abdeckung der Zeigerachse) fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

CenterScale-Eigenschaft

Beschreibung

Legt den Durchmesser des kreisförmigen Skalenmittelpunkts (Abdeckung der Zeigerachse) in Bezug zum kleineren Wert der Geometrieattribute Breite und Höhe fest oder gibt ihn zurück. Schreib-Lese-Zugriff.

Der Wertebereich ist 0,03 bis 1:

1: Der Durchmesser entspricht dem kleineren Wert der Geometrieattribute "Width" bzw. "Height".

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

CheckAlarmHigh-Eigenschaft

Beschreibung

TRUE, wenn der Grenzwert "AlarmHigh" überwacht wird. BOOLEAN Schreib-Lese-Zugriff. Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften "AlarmHigh", "ColorAlarmHigh" und "TypeAlarmHigh" festgelegt.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

CheckAlarmLow-Eigenschaft

Beschreibung

TRUE, wenn der Grenzwert "AlarmLow" überwacht wird. BOOLEAN Schreib-Lese-Zugriff. Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften "AlarmLow", "ColorAlarmLow" und "TypeAlarmLow" festgelegt.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

CheckLimitHigh4-Eigenschaft

Beschreibung

TRUE, wenn der obere Grenzwert "Reserve 4" überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.
Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften "LimitHigh4", "ColorLimitHigh4" und "TypeLimitHigh4" festgelegt.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

CheckLimitHigh5-Eigenschaft

Beschreibung

TRUE, wenn der obere Grenzwert "Reserve 5" überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.
Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften "LimitHigh5", "ColorLimitHigh5" und "TypeLimitHigh5" festgelegt.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

CheckLimitLow4-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Reserve 4" überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.
Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften "LimitLow4", "ColorLimitLow4" und "TypeLimitLow4" festgelegt.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

CheckLimitLow5-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Reserve 5" überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.
Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften "LimitLow5", "ColorLimitLow5" und "TypeLimitLow5" festgelegt.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

CheckToleranceHigh-Eigenschaft

Beschreibung

TRUE, wenn der Grenzwert "ToleranceHigh" überwacht wird. BOOLEAN Schreib-Lese-Zugriff.
Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften "ToleranceHigh", "ColorToleranceHigh" und "TypeToleranceHigh" festgelegt.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

CheckToleranceLow-Eigenschaft

Beschreibung

TRUE, wenn der Grenzwert "ToleranceLow" überwacht wird. BOOLEAN Schreib-Lese-Zugriff.
Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften "ToleranceLow", "ColorToleranceLow" und "TypeToleranceLow" festgelegt.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

CheckWarningHigh-Eigenschaft

Beschreibung

TRUE, wenn der Grenzwert "WarningHigh" überwacht wird. BOOLEAN Schreib-Lese-Zugriff. Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften "WarningHigh", "ColorWarningHigh" und "TypeWarningHigh" festgelegt.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

CheckWarningLow-Eigenschaft

Beschreibung

TRUE, wenn der Grenzwert "WarningLow" überwacht wird. BOOLEAN Schreib-Lese-Zugriff. Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften "WarningLow", "ColorWarningLow" und "TypeWarningLow" festgelegt.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ClearOnError-Eigenschaft

Beschreibung

TRUE, wenn der Feldeintrag bei einer Fehleingabe automatisch gelöscht wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

ClearOnNew-Eigenschaft

Beschreibung

TRUE, wenn der Feldeintrag gelöscht wird, sobald das EA-Feld den Fokus erhält. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

Closeable-Eigenschaft (vor WinCC V7)

Beschreibung

TRUE, wenn das Fenster im Runtime geschlossen werden kann. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

Closeable-Eigenschaft

Schließbar - Closeable

Legt fest, ob das Control in Runtime schließbar ist.

Wert	Erklärung
TRUE	Das Control ist in Runtime schließbar.
FALSE	Das Control ist nicht in Runtime schließbar.

Das Attribut ist mit dem Namen **Closeable** dynamisierbar. Der Datentyp ist BOOLEAN.

CloseButton-Eigenschaft

Beschreibung

TRUE, wenn das Fenster eine "Schließen"-Schaltfläche besitzt. Nur Lese-Zugriff.

Siehe auch

Bildfenster (Seite 190)

Applikationsfenster (Seite 185)

ScreenItem-Objekt (Seite 134)

Co

CoarseGrid-Eigenschaft

Beschreibung

TRUE wenn die Werteachse mit langen Teilstrichen skaliert wird. Der Abstand zweier langer Teilstriche kann über die Eigenschaft "CoarseGridValue" verändert werden. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

CoarseGridX-Eigenschaft

Beschreibung

TRUE, wenn die X-Achse mit langen Teilstrichen skaliert wird. Der Abstand zweier langer Teilstriche kann über die Eigenschaft "CoarseGridValueX" verändert werden. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

CoarseGridY-Eigenschaft

Beschreibung

TRUE, wenn die Y-Achse mit langen Teilstrichen skaliert wird. Der Abstand zweier langer Teilstriche kann über die Eigenschaft "CoarseGridValueY" verändert werden. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

CoarseGridValue-Eigenschaft

Beschreibung

Legt den Abstand zweier langer Teilstrichen der Skalierung fest. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "CoarseGrid".

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

CoarseGridValueX-Eigenschaft

Beschreibung

Legt den Abstand zweier langer Teilstrichen der Skalierung der X-Achse fest oder gibt ihn zurück. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "CoarseGridX".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

CoarseGridValueY-Eigenschaft

Beschreibung

Legt den Abstand zweier langer Teilstrichen der Skalierung der Y-Achse fest oder gibt ihn zurück. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "CoarseGridY".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

CollectValue-Eigenschaft

Beschreibung

Beinhaltet in Runtime als Eingangswert den jeweils aktuellen Zustand der aktiven Meldeklassen. LONG Schreib-Lese-Zugriff.
Durch Dynamisierung über eine Variable kann beispielsweise der Wert aus den Sammelanzeigen hierarchisch untergeordneter Bilder ermittelt werden.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

ColMove-Eigenschaft

Beschreibung

TRUE, wenn die Spaltenanordnung verändert werden kann. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

Color-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert ein Spaltenpaar bzw. eine Kurve. "Color" legt die in diesem Spaltenpaar verwendete Schriftfarbe bzw. die Farbe der Kurve fest. LONG Schreib-Lese-Zugriff. Die Angabe der Farbe erfolgt als RGB-Wert.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

ColorAlarmHigh-Eigenschaft

Beschreibung

Legt die Balkenfarbe für den Grenzwert "AlarmHigh" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.
Die Eigenschaft "CheckAlarmHigh" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ColorAlarmLow-Eigenschaft

Beschreibung

Legt die Balkenfarbe für den Grenzwert "AlarmLow" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.
Die Eigenschaft "CheckAlarmLow" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ColorBottom-Eigenschaft

Beschreibung

Legt die Farbe für den unteren/rechten Anschlag des Slider-Objektes fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Slider (Seite 221)

ScreenItem-Objekt (Seite 134)

ColorChangeType-Eigenschaft

Beschreibung

TRUE, wenn bei einer Farbänderung (z.B. beim Erreichen eines Grenzwerts) der Farbumschlag segmentweise erfolgen soll. Bei FALSE gilt der Farbumschlag für den gesamten Balken. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ColorLimitHigh4-Eigenschaft

Beschreibung

Legt die Farbe für den oberen Grenzwert "Reserve 4" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.
Die Eigenschaft "CheckLimitHigh4" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ColorLimitHigh5-Eigenschaft

Beschreibung

Legt die Farbe für den oberen Grenzwert "Reserve 5" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.
Die Eigenschaft "CheckLimitHigh5" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ColorLimitLow4-Eigenschaft

Beschreibung

Legt die Farbe für den unteren Grenzwert "Reserve 4" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.
Die Eigenschaft "CheckLimitLow4" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ColorLimitLow5-Eigenschaft

Beschreibung

Legt die Farbe für den unteren Grenzwert "Reserve 5" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.
Die Eigenschaft "CheckLimitLow5" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ColorToleranceHigh-Eigenschaft

Beschreibung

Legt die Farbe für den oberen Grenzwert "ToleranceHigh" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.
Die Eigenschaft "CheckToleranceHigh" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ColorToleranceLow-Eigenschaft

Beschreibung

Legt die Farbe für den unteren Grenzwert "ToleranceLow" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.
Die Eigenschaft "CheckToleranceLow" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ColorTop-Eigenschaft

Beschreibung

Legt die Farbe für den oberen/linken Anschlag des Slider-Objektes fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Slider (Seite 221)

ScreenItem-Objekt (Seite 134)

ColorWarningHigh-Eigenschaft

Beschreibung

Legt die Farbe für den oberen Grenzwert "WarningHigh" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.
Die Eigenschaft "CheckWarningHigh" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ColorWarningLow-Eigenschaft

Beschreibung

Legt die Farbe für den unteren Grenzwert "WarningLow" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.
Die Eigenschaft "CheckWarningLow" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ColTitle-Eigenschaft

Beschreibung

TRUE, wenn die Spalten im Meldefenster eine Titelleiste besitzen sollen. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

ColumnAdd-Eigenschaft

Übernehmen - ColumnAdd

Übernimmt die ausgewählte Spalte aus der Liste der vorhandenen Spalten in die Liste der gewählten Spalten.

Das Attribut ist mit dem Namen **ColumnAdd** dynamisierbar. Der Datentyp ist STRING.

ColumnAlias-Eigenschaft

ColumnAlias

Gibt den im Anwenderarchiv festgelegten Alias für den Namen der Spalte an.

Das Attribut ist mit dem Namen **ColumnAlias** dynamisierbar. Der Datentyp ist STRING.

ColumnAlign-Eigenschaft

Ausrichtung - ColumnAlign

Legt fest, wie die ausgewählte Spalte ausgerichtet wird.

Folgende Einstellungen sind möglich:

Wert	Beschreibung	Erklärung
0	links	Die ausgewählte Spalte wird linksbündig angezeigt.
1	zentriert	Die ausgewählte Spalte wird zentriert angezeigt.
2	rechts	Die ausgewählte Spalte wird rechtsbündig angezeigt.

Das Attribut ist mit dem Namen **ColumnAlign** dynamisierbar. Der Datentyp ist LONG.

ColumnAutoPrecisions-Eigenschaft

Nachkommastellen Automatisch - ColumnAutoPrecisions

Legt fest, ob die Anzahl der Nachkommastellen automatisch festgelegt wird.

Wert	Erklärung
TRUE	Anzahl der Nachkommastellen wird automatisch festgelegt. Der Wert im Feld "Nachkommastellen" ist nicht wirksam.
FALSE	Der Wert im Feld "Nachkommastellen" ist wirksam.

Das Attribut ist mit dem Namen **ColumnAutoPrecisions** dynamisierbar. Der Datentyp ist BOOLEAN.

ColumnCaption-Eigenschaft

Bezeichnung - ColumnCaption

Legt die Bezeichnung für die ausgewählte Spalte fest.

Das Attribut ist mit dem Namen **ColumnCaption** dynamisierbar. Der Datentyp ist STRING.

ColumnCount-Eigenschaft

ColumnCount

Gibt die Anzahl der projektierten Spalten an.

Das Attribut ist mit dem Namen **ColumnCount** dynamisierbar. Der Datentyp ist LONG.

ColumnDateFormat-Eigenschaft

Datumsformat - ColumnDateFormat

Legt fest, welches Datumsformat zur Anzeige verwendet wird.

Folgende Datumsformate stehen zur Verfügung:

Wert	Erklärung
Automatisch	Das Datumsformat wird automatisch bestimmt.
dd.MM.yy	Tag.Monat.Jahr, z.B. 24.12.07.
dd.MM.yyyy	Tag.Monat.Jahr, z.B. 24.12.2007.
dd/MM/yy	Tag/Monat/Jahr, z.B. 24/12/07.
dd/MM/yyyy	Tag/Monat/Jahr, z.B. 24/12/2007.

Das Attribut ist mit dem Namen **ColumnDateFormat** dynamisierbar. Der Datentyp ist STRING.

ColumnDMVarName-Eigenschaft

ColumnDMVarName

Gibt den Namen der Variablen an, die Sie im Anwenderarchiv der Spalte zugeordnet haben.

Das Attribut ist mit dem Namen **ColumnDMVarName** dynamisierbar. Der Datentyp ist STRING.

ColumnExponentialFormat-Eigenschaft

Exponentialdarstellung - ColumnExponentialFormat

Legt fest, ob die Werte der ausgewählten Spalte in Exponentialdarstellung angezeigt werden.

Wert	Erklärung
TRUE	Die Werte werden in Exponentialdarstellung angezeigt.
FALSE	Die Werte werden in Dezimaldarstellung angezeigt.

Das Attribut ist mit dem Namen **ColumnExponentialFormat** dynamisierbar. Der Datentyp ist BOOLEAN.

ColumnFlagNotNull-Eigenschaft

ColumnFlagNotNull

Legt fest, ob das der Spalte zugeordnete Feld des Anwenderarchivs einen Wert haben muss.

Wert	Erklärung
Ja	Der Wert der Spalte muss einen Wert haben.
Nein	Der Wert der Spalte kann einen Wert haben.

Das Attribut ist mit dem Namen **ColumnFlagNotNull** dynamisierbar. Der Datentyp ist BOOLEAN.

ColumnFlagUnique-Eingenschaft

ColumnFlagUnique

Legt fest, ob das der Spalte zugeordnete Feld des Anwenderarchivs einen eindeutigen Wert haben muss. Die Werte in dieser Spalte müssen sich voneinander unterscheiden.

Wert	Erklärung
TRUE	Der Wert der Spalte muss einen eindeutigen Wert haben.
FALSE	Der Wert der Spalte muss keinen eindeutigen Wert haben.

Das Attribut ist mit dem Namen **ColumnFlagUnique**dynamisierbar. Der Datentyp ist BOOLEAN.

ColumnHideText-Eigenschaft

Inhalt als Text - ColumnHideText

Legt fest, ob der Inhalt der ausgewählten Spalte als Text angezeigt wird.

Wert	Erklärung
TRUE	Der Inhalt wird nicht als Text angezeigt. Die Option ist nicht aktiviert.
FALSE	Der Inhalt wird als Text angezeigt. Die Option ist aktiviert.

Das Attribut ist mit dem Namen **ColumnHideText** dynamisierbar. Der Datentyp ist BOOLEAN.

ColumnHideTitleText-Eigenschaft

Überschrift als Text - ColumnHideTitleText

Legt fest, ob die Überschrift der ausgewählten Spalte als Text angezeigt wird.

Wert	Erklärung
TRUE	Die Überschrift wird nicht als Text angezeigt. Die Option ist nicht aktiviert.
FALSE	Die Überschrift wird als Text angezeigt. Die Option ist aktiviert.

Das Attribut ist mit dem Namen **ColumnHideTitleText** dynamisierbar. Der Datentyp ist BOOLEAN.

ColumnIndex-Eingenschaft

ColumnIndex

Referenziert eine Spalte des Controls. Unter Verwendung des Attributs können Sie einer bestimmten Spalte die Werte anderer Eigenschaften zuweisen.

Gültige Werte für "ColumnIndex" liegen zwischen 0 und "ColumnCount" minus 1. Das Attribut "ColumnCount" gibt die Anzahl der vorhandenen Spalten an.

Das Attribut "ColumnIndex" ist über das Attribut **ColumnIndex** dynamisierbar. Der Datentyp ist LONG.

ColumnLeadingZeros-Eigenschaft

Mit führenden Nullen - ColumnLeadingZeros

Legt fest, ob die Werte der ausgewählten Spalte mit führenden Nullen angezeigt wird. Über "Anzahl der Stellen" bzw. "ColumnLeadingZeros" legen Sie die Anzahl der führenden Nullen fest. Die maximale Anzahl beträgt "11". Der Wert "0" bewirkt, dass keine führende Nullen angezeigt werden. Die Option "Mit führenden Nullen" wird deaktiviert.

Das Attribut ist mit dem Namen **ColumnLeadingZeros** dynamisierbar. Der Datentyp ist LONG.

ColumnLength-Eigenschaft

Länge in Zeichen - ColumnLength

Legt die Breite für die gewählte Spalte fest.

Das Attribut ist mit dem Namen **ColumnLength** dynamisierbar. Der Datentyp ist LONG.

ColumnMaxValue-Eigenschaft

ColumnMaxValue

Gibt den im Anwenderarchiv festgelegten Maximalwert der Spalte an.

Das Attribut ist mit dem Namen **ColumnMaxValue** dynamisierbar. Der Datentyp ist STRING.

ColumnMinValue-Eigenschaft

ColumnMinValue

Gibt den im Anwenderarchiv festgelegten Minimalwert der Spalte an.

Das Attribut ist mit dem Namen **ColumnMinValue** dynamisierbar. Der Datentyp ist STRING.

ColumnName-Eigenschaft

ColumnName

Gibt den Namen der über das Attribut "ColumnIndex" referenzierten Spalte an.

Das Attribut ist mit dem Namen **ColumnName** dynamisierbar. Der Datentyp ist STRING.

ColumnPosition-Eigenschaft

ColumnPosition

Zeigt die im Anwenderarchiv festgelegte Position des Feldes an.

Das Attribut ist mit dem Namen **ColumnPosition** dynamisierbar. Der Datentyp ist LONG.

ColumnPrecisions-Eigenschaft

Nachkommastellen - ColumnPrecisions

Legt die Anzahl der Nachkommastellen der Werte der ausgewählten Spalte fest. Sie können den Wert nur eingeben, wenn die Option "Automatisch" nicht aktiviert ist.

Das Attribut ist mit dem Namen **ColumnPrecisions** dynamisierbar. Der Datentyp ist SHORT.

ColumnReadAccess-Eigenschaft

ColumnReadAccess

Gibt die im Anwenderarchiv festgelegte Berechtigung für den lesenden Zugriff auf die Spalte an. Die Zahl entspricht der Nummer, die im Editor "User Administartor" der Berechtigung zugeordnet ist.

Das Attribut ist mit dem Namen **ColumnReadAccess** dynamisierbar. Der Datentyp ist LONG.

ColumnReadonly-Eigenschaft

Schreibgeschützt - ColumnReadonly

Legt fest, ob die ausgewählte Spalte schreibgeschützt ist.

Wert	Erklärung
TRUE	Die Spalte ist schreibgeschützt.
FALSE	Die Spalte ist nicht schreibgeschützt. Sie können die Werte der Spalte in Runtime ändern, wenn Sie auf der Registerkarte "Allgemeines" die Option "Ändern" aktivieren.

Das Attribut ist mit dem Namen **ColumnReadonly** dynamisierbar. Der Datentyp ist BOOLEAN.

ColumnRemove-Eigenschaft

Entfernen - ColumnRemove

Entfernt die ausgewählte Spalte aus der Liste der gewählten Spalten und fügt ihn in die Liste der vorhandenen Spalten ein.

Das Attribut ist mit dem Namen **ColumnRemove** dynamisierbar. Der Datentyp ist STRING.

ColumnRepos-Eigenschaft

Auf/Ab - ColumnRepos

Ändert die Reihenfolge der Spalten. "Auf" und "Ab" bewegen die ausgewählte Spalte in der Liste nach oben oder unten. Dadurch wird die Spalte in der Tabelle weiter vorne oder hinten platziert.

Das Attribut ist mit dem Namen **ColumnRepos** dynamisierbar. Der Datentyp ist LONG.

ColumnResize-Eigenschaft

Breite änderbar - ColumnResize

Legt fest, ob Sie die Breite der Spalten ändern können.

Wert	Erklärung
TRUE	Sie können die Breite der Spalten ändern.
FALSE	Sie können die Breite der Spalten nicht ändern.

Das Attribut ist mit dem Namen **ColumnResize** dynamisierbar. Der Datentyp ist BOOLEAN.

ColumnSrollbar-Eigenschaften

Spalten-Rollbalken - ColumnSrollbar

Legt fest, ob Spalten-Rollbalken angezeigt werden.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	nein	Die Spalten-Rollbalken werden nicht angezeigt.
1	bei Bedarf	Die Spalten-Rollbalken werden angezeigt, wenn der Platzbedarf des Controls in vertikaler Richtung größer ist als der zur Verfügung stehende Anzeigebereich.
2	immer	Die Spalten-Rollbalken werden immer angezeigt.

Das Attribut ist mit dem Namen **ColumnScrollbar** dynamisierbar. Der Datentyp ist LONG.

ColumnShowDate-Eigenschaft

Datum anzeigen - ColumnShowDate

Legt fest, ob der Block "Uhrzeit" mit Uhrzeit und Datum in einem Feld angezeigt wird.

Wert	Erklärung
TRUE	Datum und Uhrzeit werden angezeigt. Das Datumsformat wird im Feld "Datumsformat" festgelegt.
FALSE	Die Uhrzeit wird angezeigt.

Das Attribut ist mit dem Namen **ColumnShowDate** dynamisierbar. Der Datentyp ist BOOLEAN.

ColumnShowIcon-Eigenschaft

Inhalt als Symbol - ColumnShowIcon

Legt fest, ob der Inhalt der ausgewählten Spalte als Symbol angezeigt wird.

Wert	Erklärung
TRUE	Der Inhalt wird als Symbol angezeigt.
FALSE	Der Inhalt wird nicht als Symbol angezeigt.

Das Attribut ist mit dem Namen **ColumnShowIcon** dynamisierbar. Der Datentyp ist BOOLEAN.

ColumnShowTitleIcon-Eigenschaft

Überschrift als Symbol - ColumnShowTitleIcon

Legt fest, ob die Überschrift der ausgewählten Spalte als Symbol angezeigt wird.

Wert	Erklärung
TRUE	Die Überschrift wird als Symbol angezeigt.
FALSE	Die Überschrift wird nicht als Symbol angezeigt.

Das Attribut ist mit dem Namen **ColumnShowTitleIcon** dynamisierbar. Der Datentyp ist BOOLEAN.

ColumnSort-Eigenschaft

ColumnSort

Legt fest, wie die im Attribut "ColumnIndex" referenzierte Spalte des Anwenderarchivs sortiert wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	nein	Keine Sortierung
1	aufsteigend	Aufsteigende Sortierung vom kleinsten zum größten Wert.
2	absteigend	Absteigende Sortierung vom größten zum kleinsten Wert.

Das Attribut ist mit dem Namen **ColumnSort** dynamisierbar. Der Datentyp ist LONG.

ColumnSortIndex-Eigenschaft

ColumnSortIndex

Gibt die Sortierreihenfolge der im "ColumnIndex" referenzierten Spalte an. Wenn Sie den Wert auf "0" setzen, wird das Sortierkriterium in "ColumnSort" entfernt.

Das Attribut ist mit dem Namen **ColumnSortIndex** dynamisierbar. Der Datentyp ist LONG.

ColumnStartValue-Eigenschaft

ColumnStartValue

Gibt den im Anwenderarchiv festgelegten Startwert der Spalte an.

Das Attribut ist mit dem Namen **ColumnStartValue** dynamisierbar. Der Datentyp ist STRING.

ColumnStringLength-Eigenschaft

ColumnStringLength

Zeigt die im Anwenderarchiv festgelegte Länge der Zeichenkette der Spalte an.

Das Attribut ist mit dem Namen **ColumnStringLength** dynamisierbar. Der Datentyp ist LONG.

ColumnTimeFormat-Eigenschaft

Zeitformat - ColumnTimeFormat

Legt fest, welches Zeitformat zur Anzeige verwendet wird.

Folgende Zeitformate stehen zur Verfügung:

Wert	Erklärung
Automatisch	Das Zeitformat wird automatisch bestimmt.
HH:mm:ss.ms	Stunden:Minuten:Sekunden, z.B. 15:35:44.240.
hh:mm:ss tt	Stunden:Minuten:Sekunden AM/PM, z.B. 03:35:44 PM.
hh:mm:ss.ms tt	Stunden:Minuten:Sekunden.Millisekunden AM/PM, z.B. 03:35:44.240 PM.

Das Attribut ist mit dem Namen **ColumnTimeFormat** dynamisierbar. Der Datentyp ist STRING.

ColumnTitleAlign-Eigenschaft

Ausrichtung Spaltenüberschrift - ColumnTitleAlign

Legt fest, wie die Spaltenüberschriften ausgerichtet werden.

Folgende Einstellungen stehen zur Auswahl:

Wert	Beschreibung	Erklärung
0	links	Die Spaltenüberschriften werden linksbündig angezeigt.
1	zentriert	Die Spaltenüberschriften werden zentriert angezeigt.
2	rechts	Die Spaltenüberschriften werden rechtsbündig angezeigt.
3	wie der Tabelleninhalt	Die Spaltenüberschriften werden wie der zugehörige Inhalt der Spalte ausgerichtet.

Das Attribut ist mit dem Namen **ColumnTitleAlign** dynamisierbar. Der Datentyp ist LONG.

ColumnTitles-Eigenschaft

Anzeigen Spaltenüberschrift - ColumnTitles

Legt fest, ob die Spaltenüberschrift angezeigt wird.

Wert	Erklärung
TRUE	Die Spaltenüberschrift wird angezeigt.
FALSE	Die Spaltenüberschrift wird nicht angezeigt.

Das Attribut ist mit dem Namen **ColumnTitles** dynamisierbar. Der Datentyp ist BOOLEAN.

ColumnType-Eigenschaft

Typ - ColumnType

Gibt den im Anwenderarchiv festgelegten Datentyp der ausgewählten Spalte an.

Das Attribut ist mit dem Namen **ColumnType** dynamisierbar. Der Datentyp ist LONG.

ColumnVisible-Eigenschaft

ColumnVisible

Legt fest, ob die über das Attribut "ColumnIndex" referenzierte Spalte angezeigt wird.

Wert	Erklärung
TRUE	Die Spalte wird angezeigt.
FALSE	Die Spalte wird nicht angezeigt

Das Attribut ist mit dem Namen **ColumnVisible** dynamisierbar. Der Datentyp ist BOOLEAN.

ColumnWriteAccess-Eigenschaft

ColumnWriteAccess

Gibt die im Anwenderarchiv festgelegte Berechtigung für den schreibenden Zugriff auf die Spalte an. Die Zahl entspricht der Nummer, die im Editor "User Administrator" der Berechtigung zugeordnet ist.

Das Attribut ist mit dem Namen **ColumnWriteAccess** dynamisierbar. Der Datentyp ist LONG.

ColWidth-Eigenschaft

Beschreibung

TRUE, wenn die Breite der Spalten des Meldefensters veränderbar sein soll. Die Breite der Spalten kann allerdings nur verändert werden, wenn die Eigenschaft "AutoScroll" nicht aktiviert ist. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

Command-Eigenschaft

Beschreibung

TRUE, wenn die Aktualisierung der im Control dargestellten Werte erzwungen werden soll.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

Comment-Eigenschaft

Beschreibung

Liest oder setzt den Kommentar des Alarm-Objekts.

Siehe auch

Alarms-Objekt (Auflistung) (Seite 120)

CommonTime-Eigenschaft

Beschreibung

TRUE, wenn im Tabellenfenster eine gemeinsame Zeitspalte verwendet wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

CommonX-Eigenschaft

Beschreibung

TRUE, wenn die Kurven des Kurvenfensters mit einer gemeinsamen X-Achse dargestellt werden. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

CommonY-Eigenschaft

Beschreibung

TRUE, wenn die Kurven des Kurvenfensters mit einer gemeinsamen Y-Achse dargestellt werden. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

ScreenItem-Objekt (Seite 134)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ComputerName-Eigenschaft

Beschreibung

Gibt den Namen des Computers zurück, auf dem das Alarm-Objekt ausgelöst wurde.

ComputerName (readonly)

Siehe auch

Alarms-Objekt (Auflistung) (Seite 120)

Context-Eigenschaft

Beschreibung

Liest oder setzt das Serverpräfix des Alarm-Objekts.

Siehe auch

Alarms-Objekt (Auflistung) (Seite 120)

ConnectTrendWindows-Eigenschaft

Verbundene Kurvenfenster - ConnectTrendWindows

Legt fest, ob die projektierten Kurvenfenster verbunden werden. Voraussetzung ist, dass Sie mehrere Kurvenfenster projektiert haben.

Die verbundenen Kurvenfenster haben folgende Eigenschaften:

- Sie können eine gemeinsame X-Achse haben.
- Sie haben einen Rollbalken.
- Sie haben ein Lineal.
- Die Zoomfunktionen für ein Kurvenfenster wirken sich auf die verbundenen Kurvenfenster aus.

Wert	Beschreibung
TRUE	Alle projektierten Kurvenfenster werden verbunden.
FALSE	Die Kurvenfenster werden separat dargestellt.

Das Attribut ist mit dem Namen **ConnectTrendWindows** dynamisierbar. Der Datentyp ist BOOLEAN.

ContinousChange-Eigenschaft

Beschreibung

Bestimmt die Art der Übergabe des mit dem Schieber eingestellten Werts (Eigenschaft "Position") in Runtime:

- FALSE: Der Wert der Eigenschaft "Position" wird erst beim Loslassen der Maustaste übergeben.
- TRUE: Der Wert der Eigenschaft "Position" wird sofort bei Änderung der Schieberposition übergeben.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

Count-Eigenschaft

Beschreibung

Liefert die Anzahl der Elemente einer Auflistung zurück.

INTEGER (Nur-Lese-Zugriff).

Beispiel

Das Beispiel zeigt, wie die Anzahl der Objekte der DataSet-Auflistung ausgeben wird.

```
'VBS165
HMIRuntime.Trace "Count: " & HMIRuntime.DataSet.Count & vbNewLine
```

Das folgende Beispiel fügt der TagSet-Auflistung zwei Variablen hinzu und gibt die Count-Eigenschaft als Trace aus.

```
'VBS177
Dim group
Set group = HMIRuntime.Tags.CreateTagSet
group.Add "Motor1"
group.Add "Motor2"
HMIRuntime.Trace "Count: " & group.Count & vbNewLine
```

Siehe auch

CreateTagSet-Methode (Seite 691)

TagSet-Objekt (Auflistung) (Seite 151)

ScreenItems-Objekt (Auflistung) (Seite 138)

Screens-Objekt (Auflistung) (Seite 143)

Layers-Objekt (Auflistung) (Seite 130)

DataSet-Objekt (Auflistung) (Seite 125)

ProcessValues-Objekt (Auflistung) (Seite 133)

Cu

CurrentContext-Eigenschaft

Beschreibung

Bei einem Bildfenster wird der Server ausgelesen, von dem das Bild kommt, in dem das Skript enthalten ist.

Die Eigenschaft "CurrentContext" kann unterschiedliche Ergebnisse liefern: Wenn z.B. in einem lokalen Grundbild ein Bildfenster liegt, das ein Bild eines Servers darstellt, werden zwei Fälle unterschieden:

- Die Eigenschaft "CurrentContext" wird in einer Aktion des Bildfensterbildes verwendet: Als Ergebnis wird der symbolische Rechnername des Servers (Package Eigenschaft) erweitert um zwei Doppelpunkte, beispielsweise "WinCCProject_MyComputer::" zurückgegeben.
- Die Eigenschaft "CurrentContext" wird in einer Aktion des Grundbildes verwendet: Als Ergebnis wird eine leere Zeichenkette zurückgegeben.

Siehe auch

HMIRuntime-Objekt (Seite 127)

Cursor-Eigenschaft

Beschreibung

Steuert das Aussehen des Cursors in Runtime, wenn er sich über dem Symbol befindet.

- 0: Der Cursor hat die Gestalt eines Pfeils und verändert sein Aussehen nicht, wenn er über dem Symbol positioniert wird.
- 1: Der Cursor hat die Gestalt eine 3D-Pfeils, begleitet von einem grünen Blitzsymbol. Damit wird in Runtime signalisiert, dass das betreffende Objekt bedienbar ist.

Siehe auch

ScreenItem-Objekt (Seite 134)

HMI Symbol Library (Seite 248)

Cursor-Eigenschaft

Mauszeiger (Cursor)

Legt fest, ob der Mauszeiger in Runtime auf dem Symbol angezeigt wird.

Wert	Erkärung
TRUE	Der Mauszeiger wird in Runtime eingeblendet, wenn der Zeiger auf dem Symbol positioniert wird.
FALSE	Der Mauszeiger wird in Runtime nicht eingeblendet, wenn der Zeiger auf dem Symbol positioniert wird.

Das Attribut ist mit dem Namen **Cursor** dynamisierbar. Der Datentyp ist BOOLEAN.

CursorControl-Eigenschaft

Beschreibung

TRUE, wenn bei aktiviertem Alpha-Cursor-Modus der Cursor nach dem Verlassen des Feldes auf das nächste Feld der TAB-Reihenfolge springt. BOOLEAN Schreib-Lese-Zugriff.

Die Eigenschaft "CursorMode" muss dazu auf TRUE gesetzt sein.

Siehe auch

Textliste (Seite 208)

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

CurveForm-Eigenschaft

Beschreibung

WinCC Function Trend Control

Legt fest, wie die Messpunkte einer über die Eigenschaft "Index" referenzierten Kurve verbunden werden sollen. Schreib-Lese-Zugriff.

WinCC Online Trend Control

Die Eigenschaft "Index" referenziert eine Kurve. "CurveForm" legt fest, wie die Messpunkte verbunden werden sollen.

- 0x00000012: Darstellung der Messpunkte.
- 0x00000014: Messpunkte werden linear verbunden.

- 0x00000011: Messpunkte werden über eine Treppenkurve verbunden.
- 0x00000021: Die Fläche unter der linear verbundenen Kurve wird gefüllt.
- 0x00000022: Die Fläche unter der Treppenkurve wird gefüllt.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

CursorMode-Eigenschaft

Beschreibung

Wenn der "CursorMode" auf "ja" gesetzt ist, können Sie alle Meldungen des Umlaufarchivs in der Langzeitarchivliste seitenweise anzeigen. Über die Eigenschaft "CursorModePrefetch" bestimmen Sie die Anzahl der Meldungen, die pro Seite angezeigt werden.

Die Option "Autoscroll" muss deaktiviert sein, um zwischen den Seiten wechseln zu können. Schreib-Lese-Zugriff.

CursorModePrefetch-Eigenschaft

Beschreibung

Stellt die Anzahl der Meldungen ein, die Sie aus allen Meldungen des Umlaufarchivs in der Langzeitarchivliste seitenweise anzeigen wollen.

Die Objekteigenschaft "CursorMode" muss auf "ja" gesetzt sein.

Schreib-Lese-Zugriff.

1.14.4.5 D

Da

DangerColor-Eigenschaft

Beschreibung

Legt die Farbe des Gefahrenbereichs auf der Skala fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

Danger-Eigenschaft

Beschreibung

Legt den Skalenwert fest, ab dem der "Gefahrenbereich" beginnt, oder gibt ihn zurück. Der Bereich erstreckt sich ab dem Wert "Gefahr" bis zum Ende der Skala. Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

DataFormat-Eigenschaft

Beschreibung

Gibt den Datentyp des Objektes IOField zurück. Nur Lese-Zugriff.

Wertebereich von 0 bis 3.

0: Binär

1: Dezimal

2: String

3: Hexadezimal

Siehe auch

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

DataIndex-Eigenschaft

Beschreibung

Gibt den aktuellen Index der Daten der aktuellen Kurve zurück.

Hinweis

Die Eigenschaft wird nur für die Controls vor WinCC V7 unterstützt.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

DataLogs-Eigenschaft

Beschreibung

Liefert ein Objekt vom Typ "DataLogs" zurück.

DataLogs (read-only)

Siehe auch

DataLogs-Objekt (Seite 124)

HMIRuntime-Objekt (Seite 127)

DataSet-Eigenschaft

Beschreibung

Liefert ein Objekt vom Typ "DataSet" zurück.

DataSet (read-only)

Siehe auch

DataSet-Objekt (Auflistung) (Seite 125)

HMIRuntime-Objekt (Seite 127)

DataX-Eigenschaft

Beschreibung

Fügt einen einzelnen Datensatz ein und muss vor dem Aufruf von "InsertData" gesetzt werden.

Hinweis

Die Eigenschaft wird nur für die Controls vor WinCC V7 unterstützt.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

DataXY-Eigenschaft

Beschreibung

Fügt mehrere Datensätze als Array mit Wertepaaren ein und muss vor dem Aufruf von "InsertData" gesetzt werden.

Die Daten im Array werden übernommen, wenn "DataX" vom Typ VT_EMPTY ist. Ansonsten wird im Attribut "InsertData" das einzelne Wertepaar verwendet, welches sich aus "DataX" und "DataY" ergibt.

Hinweis

Die Eigenschaft wird nur für die Controls vor WinCC V7 unterstützt.

Siehe auch

Beispiel: So rufen Sie Methoden eines ActiveX-Controls auf (Seite 810)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

DataY-Eigenschaft

Beschreibung

Fügt einen einzelnen Datensatz ein und muss vor dem Aufruf von "InsertData" gesetzt werden.

Hinweis

Die Eigenschaft wird nur für die Controls vor WinCC V7 unterstützt.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

De - Do

DefaultMsgFilterSQL-Eigenschaft

DefaultMsgFilterSQL

Legt für eine feste Selektion der Meldungen ein SQL-Statement fest.

Wenn Sie über das Attribut "MsgFilterSQL" zusätzliche benutzerdefinierte Selektionen festlegen, werden die SQL-Statements von "DefaultMsgFilterSQL" und "MsgFilterSQL" mit "AND" verbunden.

Das Attribut ist mit dem Namen **DefaultMsgFilterSQL** dynamisierbar. Der Datentyp ist STRING.

DefaultPrecision-Eigenschaft

Beschreibung

Dieses Attribut legt die Anzahl der Nachkommastellen als Standardwert fest, mit denen die Angabe der Skalierungswerte erfolgt. Schreib-Lese-Zugriff.

DefaultRulerPrecision-Eigenschaft

Beschreibung

Dieses Attribut legt die Anzahl der Nachkommastellen als Standardwert fest, mit denen ein Messwert angezeigt wird, wenn Sie ihn über die Funktion "Anzeige Wert an dieser Stelle" ermitteln. Schreib-Lese-Zugriff.

DefaultSort-Eigenschaft

Standardsortierung - DefaultSort

Legt die Standardsortierung in den Tabellenspalten fest.

Folgende Einstellungen sind möglich:

Wert	Beschreibung	Erläuterung
0	aufsteigend	Die Liste wird von der untersten Zeile ausgehend aktualisiert.
1	absteigend	Die Liste wird von der obersten Zeile ausgehend aktualisiert.

Das Attribut ist mit dem Namen **DefaultSort** dynamisierbar. Der Datentyp ist LONG.

DefaultSort2-Eigenschaft

DefaultSort2

Legt die Sortierung in den Tabellenspalten fest, wenn Sie nicht die Standardsortierung in der Reihenfolge "Datum/Uhrzeit/Nummer" verwenden wollen. Stattdessen haben Sie in der Objekteigenschaft "DefaultSort2Column" einen Meldeblock angegeben, der eine Sortierung nach "Meldeblock/Datum/Uhrzeit/Nummer" durchführt.

Folgende Einstellungen sind möglich:

Wert	Beschreibung	Erläuterung
0	aufsteigend	Die Liste wird von der untersten Zeile ausgehend aktualisiert.
1	absteigend	Die Liste wird von der obersten Zeile ausgehend aktualisiert.

Das Attribut ist mit dem Namen **DefaultSort2** dynamisierbar. Der Datentyp ist LONG.

DefaultSort2Column-Eigenschaft

DefaultSort2Column

Legt die Sortierung in den Tabellenspalten fest, wenn Sie nicht die Standardsortierung in der Reihenfolge "Datum/Uhrzeit/Nummer" verwenden wollen.

Geben Sie einen Meldeblock mit seinem Objektnamen an.

Die Tabellenspalten werden dann in der Reihenfolge "Meldeblock/Datum/Uhrzeit/Nummer" sortiert.

Das Attribut ist mit dem Namen **DefaultSort2Column** dynamisierbar. Der Datentyp ist STRING.

DeleteData-Eigenschaft

Beschreibung

Löscht Daten des Datenpuffers der aktuellen Kurve.

TRUE: alle Daten der Kurve werden gelöscht.

FALSE: das Wertepaar an der Position "DataIndex" wird gelöscht.

Hinweis

Die Eigenschaft wird nur für die Controls vor WinCC V7 unterstützt.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

Delta-Eigenschaft

Beschreibung

Legt die Wertdifferenz zwischen zwei Hauptskalenstrichen fest oder gibt sie zurück. Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

DesiredCurveColor-Eigenschaft

Beschreibung

Legt die Farbe einer Sollkurve fest, die einer über die Eigenschaft "Index" referenzierten Kurve zugehört. Die Angabe der Farbe erfolgt als RGB-Wert. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "DesiredCurveVisible".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

DesiredCurveCurveForm-Eigenschaft

Beschreibung

Legt die Darstellungsform einer Sollkurve fest, die einer über die Eigenschaft "Index" referenzierten Kurve zugehört. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "DesiredCurveVisible".

0x00000011: Messpunkte werden über eine Treppenkurve mit durchgezogener Linie verbunden

0x00000012: Darstellung der Messpunkte

0x00000014: Messpunkte werden linear mit einer durchgezogenen Linie verbunden

0x00000021: Die Fläche unter der linear verbundenen Kurve wird gefüllt

0x00000022: Die Fläche unter der Treppenkurve wird gefüllt

0x00000031: Messpunkte werden über eine Treppenkurve mit gestrichelter Linie verbunden

0x00000032: Messpunkte werden linear mit einer gestrichelten Linie verbunden

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

DesiredCurveSourceNumberOfUAValues-Eigenschaft

Beschreibung

Legt die Anzahl der Wertepaare einer Sollkurve fest, die einer über die Eigenschaft "Index" referenzierten Kurve zugehört. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "DesiredCurveVisible".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

DesiredCurveSourceUAArchive-Eigenschaft

Beschreibung

Legt den Namen des Anwenderarchives fest, aus dem die Werte der einer über "Index" referenzierten Kurve zugehörigen Sollkurve gelesen werden. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "DesiredCurveVisible".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

DesiredCurveSourceUAArchiveStartID-Eigenschaft

Beschreibung

Legt für die Werte der einer über "Index" referenzierten Kurve zugehörigen Sollkurve die Anfangsposition fest, ab der die Werte aus dem Anwenderarchiv gelesen werden. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "DesiredCurveVisible".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

DesiredCurveSourceUAColumnX-Eigenschaft

Beschreibung

Legt die Spalte im Anwenderarchiv fest, aus der die X-Werte der einer über "Index" referenzierten Kurve zugehörigen Sollkurve gelesen werden. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "DesiredCurveVisible".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

DesiredCurveSourceUAColumnY-Eigenschaft

Beschreibung

Legt die Spalte im Anwenderarchiv fest, aus der die Y-Werte der einer über "Index" referenzierten Kurve zugehörigen Sollkurve gelesen werden. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "DesiredCurveVisible".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

DesiredCurveVisible-Eigenschaft

Beschreibung

TRUE, wenn die zu einer über "Index" referenzierten Kurve gehörende Sollkurve dargestellt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

Direction-Eigenschaft

Beschreibung

Legt die Balkenrichtung bzw. die Lage des Slider-Objektes fest oder gibt sie zurück. BOOLEAN Schreib-Lese-Zugriff. Wertebereich von 0 bis 3.

0 = oben

1 = unten

2 = links

3 = rechts

Siehe auch

Slider (Seite 221)

Balken (Seite 186)

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

DisplayOptions-Eigenschaft

Meldungen anzeigen - DisplayOptions

Legt fest, welche Meldungen angezeigt werden.

Folgende Auswahlmöglichkeiten bestehen:

Wert	Bezeichnung
0	Alle Meldungen
1	Nur eingeblendete Meldungen
2	Nur ausgeblendete Meldungen

Das Attribut ist mit dem Namen **DisplayOptions** dynamisierbar. Der Datentyp ist LONG.

DisplayOptions-Eigenschaft (vor WinCC V7)

Beschreibung

Legt fest, ob ein Button mit einer Grafik, einem Text oder beidem belegt wird.

0 Grafik oder Text: Wenn eine Grafik vorhanden ist, wird der Button mit der Grafik belegt, sonst mit einem Text.

1 Grafik und Text

2 nur Text

3 nur Grafik

DoubleClickAction-Eigenschaft

Aktion bei Doppelklick - DoubleClickAction

Legt die Aktion fest, die bei Doppelklick auf eine Meldezeile in Runtime ausgeführt wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	keine	Keine Aktion wird ausgeführt.
1	Loop in Alarm	Die Funktion "Loop in Alarm" wird aufgerufen.
2	Kommentar-Dialog öffnen	Die Tastenfunktion "Kommentar-Dialog" wird aufgerufen.
3	Infotext-Dialog öffnen	Die Tastenfunktion "Infotext-Dialog" wird aufgerufen.
4	Spalten abhängig	Die ausgeführte Aktion richtet sich nach der Spalte, in die Sie doppelgeklickt haben.

Das Attribut ist mit dem Namen **DoubleClickAction** dynamisierbar. Der Datentyp ist LONG.

1.14.4.6 E

Edit-Eigenschaft

Beschreibung

Aktiviert den Editiermodus für eine Zelle, sofern in der entsprechenden Spalte die Eigenschaft "Editable" auf TRUE gesetzt ist.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

Editable-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert ein Spaltenpaar. "Editable" legt fest, ob dieses Spaltenpaar editierbar sein soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

EditAtOnce-Eigenschaft

Beschreibung

TRUE, wenn beim Anspringen des Feldes mit der Taste <Tab> die Eingabe sofort und ohne weitere Aktion erfolgen kann. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Textliste (Seite 208)

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

Enabled-Eigenschaft

Funktion

Gibt ein Objekt zur Bedienfreigabe frei oder sperrt es bzw. gibt den entsprechenden Wert aus. TRUE: Bedienung freigegeben, FALSE: Bedienung gesperrt.

BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Das folgende Beispiel sperrt alle Objekte des Bildes "NewPDL1":

```
'VBS71
Dim objScreen
Dim objScrItem
Dim lngIndex
Dim strName
lngIndex = 1
Set objScreen = HMIRuntime.Screens("NewPDL1")
For lngIndex = 1 To objScreen.ScreenItems.Count
strName = objScreen.ScreenItems.Item(lngIndex).ObjectName    'Read names of objects
Set objScrItem = objScreen.ScreenItems(strName)
objScrItem.Enabled=False    'Lock object
Next
```

Siehe auch

Screen-Objekt (Seite 140)

ScreenItem-Objekt (Seite 134)

EnableDelete-Eigenschaft

Löschen - EnableDelete

Legt fest, ob Sie Daten des Anwenderarchivs in Runtime löschen können.

Wert	Erklärung
TRUE	Sie können Daten des Anwenderarchivs in Runtime löschen.
FALSE	Sie können keine Daten des Anwenderarchivs in Runtime löschen.

Das Attribut ist mit dem Namen **EnableDelete** dynamisierbar. Der Datentyp ist BOOLEAN.

EnableEdit-Eigenschaft

Ändern - EnableEdit

Legt fest, ob Sie die angezeigten Daten in Runtime ändern können.

Wert	Erklärung
TRUE	Sie können die Daten in Runtime ändern.
FALSE	Sie können die Daten in Runtime nicht ändern.

Das Attribut ist mit dem Namen **EnableEdit** dynamisierbar. Der Datentyp ist BOOLEAN.

EnableInsert-Eigenschaft

Hinzufügen - EnableInsert

Legt fest, ob Sie Daten zum Anwenderarchiv in Runtime hinzufügen können.

Wert	Erklärung
TRUE	Sie können Daten zum Anwenderarchiv in Runtime hinzufügen.
FALSE	Sie können keine Daten zum Anwenderarchiv in Runtime hinzufügen.

Das Attribut ist mit dem Namen **EnableInsert** dynamisierbar. Der Datentyp ist BOOLEAN.

EnablePopupMenu-Eigenschaft

EnablePopupMenu

Legt fest, ob im Control das Kontextmenü eingeschaltet wird.

Das Attribut ist mit dem Namen **EnablePopupMenu** dynamisierbar. Der Datentyp ist BOOLEAN.

EndAngle-Eigenschaft

Beschreibung

Legt das Ende des Objektes fest oder gibt es zurück. Die Angabe erfolgt im Uhrzeigersinn in Grad, beginnend bei 12:00 Uhr.

Siehe auch

Kreissegment (Seite 164)

Kreisbogen (Seite 162)

Ellipsensegment (Seite 158)

Ellipsenbogen (Seite 156)

ScreenItem-Objekt (Seite 134)

EndTime-Eigenschaft

Beschreibung

Online Table Control

Das Attribut "Index" referenziert ein Spaltenpaar. "EndTime" legt den Endzeitpunkt der Darstellung dieses Spaltenpaares fest. Ob die Angabe ausgewertet wird, ist abhängig von den Eigenschaften "TimeRange" und "CommonTime". Schreib-Lese-Zugriff.

Online Trend Control

Das Attribut "Index" referenziert eine Kurve. "EndTime" legt den Endzeitpunkt der Darstellung dieser Kurve fest. Ob die Angabe ausgewertet wird, ist abhängig von den Eigenschaften "Autorange", "TimeRange" und "CommonX".

Wenn Sie den Zeitbereich dynamisieren, verwenden Sie das Format "jjjj-mm-dd hh:mm:ss".

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

EndValue-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. "EndValue" legt die obere Grenze des darzustellenden Wertebereichs dieser Kurve fest. Ob die Angabe ausgewertet wird, ist abhängig von den Eigenschaften "Autorange" und "CommonY".

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

EndX-Eigenschaft

Beschreibung

Legt die obere Grenze der X-Achse einer mit "Index" referenzierten Kurve fest. Ob die Angabe ausgewertet wird, ist abhängig von den Eigenschaften "AutorangeX" und "CommonX".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

EndY-Eigenschaft

Beschreibung

Legt die obere Grenze der Y-Achse einer mit "Index" referenzierten Kurve fest. Ob die Angabe ausgewertet wird, ist abhängig von den Eigenschaften "AutorangeY" und "CommonY".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

ErrorDescription-Eigenschaft

Funktion

Fehlerbeschreibung der Eigenschaft "LastError". Die Fehlerbeschreibung erfolgt nur in englischer Sprache.

STRING (readonly)

Folgende Fehlermeldungen sind definiert:

Ausgabe	Beschreibung
" "	OK
"Operation Failed"	Durchführungsfehler
"Variable not found"	Variablenfehler

Ausgabe	Beschreibung
"Server down"	Server nicht verfügbar
"An error occured for one or several tags"	Multi Tag Error (Fehler bei einer oder mehreren Variablen)

Damit ErrorDescription einen Wert zurückgibt, muss zuvor ein Read durchgeführt werden.

Tritt beim Lesen oder Schreiben mehrerer Variablen über das TagSet-Objekt ein Fehler auf, so wird der Fehler "Multi Tag Error" gesetzt. Um zu ermitteln, bei welcher Variable ein Fehler auftrat und welcher Art er war, muss die ErrorDescription-Eigenschaft jeder Variablen ausgewertet werden.

Beispiel

Das folgende Beispiel gibt die Fehlerbeschreibung für die Variable "Tag1" aus:

```
'VBS72
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
objtag.Read
MsgBox objTag.ErrorDescription
```

Das folgende Beispiel fügt der TagSet-Auflistung zwei Variablen hinzu und gibt die ErrorDescription-Eigenschaft als Trace aus.

```
'VBS179
Dim group
Set group = HMIRuntime.Tags.CreateTagSet
group.Add "Motor1"
group.Add "Motor2"
HMIRuntime.Trace "ErrorDescription: " & group.ErrorDescription & vbNewLine
```

Auf die ErrorDescription-Eigenschaft einer in der Auflistung enthaltenen Variable kann wie folgt zugegriffen werden:

```
HMIRuntime.Trace "ErrorDescription: " & group("Motor1").ErrorDescription & vbNewLine
```

Siehe auch

LastError-Eigenschaft (Seite 435)

QualityCode-Eigenschaft (Seite 523)

TagSet-Objekt (Auflistung) (Seite 151)

Tag-Objekt (Seite 146)

Exponent-Eigenschaft

Beschreibung

TRUE, wenn die Zahlendarstellung mit Exponenten (z.B."1,00e+000")erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ExportDirectoryChangeable-Eigenschaft

Verzeichnis änderbar - ExportDirectoryChangeable

Legt fest, ob das Verzeichnis für den Datenexport in Runtime geändert werden kann.

Wert	Erklärung
TRUE	Das Verzeichnis für den Datenexport kann in Runtime geändert werden.
FALSE	Das Verzeichnis für den Datenexport kann nicht in Runtime geändert werden.

Das Attribut ist mit dem Namen **ExportDirectoryChangeable** dynamisierbar. Der Datentyp ist BOOLEAN.

ExportDirectoryname-Eigenschaft

Verzeichnis - ExportDirectoryname

Legt das Verzeichnis fest, in das die Runtime-Daten exportiert werden.

Über die Auswahlschaltfläche können Sie das gewünschte Verzeichnis auswählen oder anlegen.

Das Attribut ist mit dem Namen **ExportDirectoryname** dynamisierbar. Der Datentyp ist STRING.

ExportFileExtension-Eigenschaft

ExportFileExtension

Legt die Dateierweiterung der Exportdatei fest.

Bisher wird nur die Datenerweiterung "csv" unterstützt.

Das Attribut ist mit dem Namen **ExportFileExtension** dynamisierbar. Der Datentyp ist STRING.

ExportFilename-Eigenschaft

Dateiname - ExportFilename

Legt den Dateinamen der Datei fest, in welche die Runtime-Daten exportiert werden.

Das Attribut ist mit dem Namen **ExportFilename** dynamisierbar. Der Datentyp ist STRING.

ExportFilenameChangeable-Eigenschaft

Dateiname änderbar - ExportFilenameChangeable

Legt fest, ob der Dateiname der Exportdatei in Runtime geändert werden kann.

Wert	Erklärung
TRUE	Der Dateiname der Exportdatei kann in Runtime geändert werden.
FALSE	Der Dateiname der Exportdatei kann nicht in Runtime geändert werden.

Das Attribut ist mit dem Namen **ExportFilenameChangeable** dynamisierbar. Der Datentyp ist BOOLEAN.

ExportFormatGuid-Eigenschaft

ExportFormatGuid

Festgelegte Zuordnung von Ident-Nummer und Export-Provider.

Das Attribut ist mit dem Namen **ExportFormatGuid** dynamisierbar. Der Datentyp ist STRING.

ExportFormatName-Eigenschaft

Format - ExportFormatName

Legt das Dateiformat für den Export fest.

Zurzeit steht nur das "csv"-Dateiformat für den Export zur Verfügung.

Das Attribut ist mit dem Namen **ExportFormatName** dynamisierbar. Der Datentyp ist STRING.

Siehe auch

So exportieren Sie Runtime-Daten

ExportParameters-Eigenschaft

ExportParameters

Legt die Parameter des ausgewählten Formats über den Eigenschaften-Dialog fest.

Das Attribut ist mit dem Namen **ExportParameters** dynamisierbar. Der Datentyp ist VARIANT.

ExportSelection-Eigenschaft

Umfang des Datenexports - ExportSelection

Legt fest, welche Runtime-Daten des Controls exportiert werden.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	alles	Alle Runtime-Daten des Controls werden exportiert.
1	Auswahl	Die ausgewählten Runtime-Daten des Controls werden exportiert.

Das Attribut ist mit dem Namen **ExportSelection** dynamisierbar. Der Datentyp ist LONG.

ExportShowDialog-Eigemschaft

Dialog anzeigen - ExportShowDialog

Legt fest, ob der Dialog zum Datenexport in Runtime angezeigt wird.

Wert	Erklärung
TRUE	Der Dialog wird in Runtime angezeigt.
FALSE	Der Dialog wird nicht in Runtime angezeigt.

Das Attribut ist mit dem Namen **ExportShowDialog** dynamisierbar. Der Datentyp ist BOOLEAN.

ExportXML-Eigenschaft

ExportXML

Wird nur intern verwendet.

Das Attribut ist mit dem Namen **ExportXML** dynamisierbar.

ExtendedOperation-Eigenschaft

Beschreibung

TRUE, wenn der Schieberegler auf den zugehörigen Endwert (Minimalwert/Maximalwert) gestellt wird. Dies geschieht bei Mausklick auf den Bereich außerhalb der aktuellen Reglereinstellung. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Slider (Seite 221)

ScreenItem-Objekt (Seite 134)

ExtendedZoomingEnable-Eigenschaft

Beschreibung

Aktiviert / deaktiviert die ExtendedZooming-Eigenschaft eines Bildes.

Durch ExtendedZooming kann die Ansicht eines Prozessbildes in Runtime durch Einsatz des Mausrades vergrößert oder verkleinert werden.

BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Aktiviert ExtendedZooming für das Bild NewPDL1.

```
'VBS155
Dim objScreen
Set objScreen = HMIRuntime.Screens("NewPDL1")
objScreen.ExtendedZoomingEnable = 1
```

Siehe auch

Screen-Objekt (Seite 140)

1.14.4.7 F

Fe - Fl

FeatureFullscreen-Eigenschaft

FeatureFullscreen

Legt fest, ob im Control die Funktion "Vollbild" verfügbar ist.

Das Attribut ist mit dem Namen **FeatureFullscreen** dynamisierbar. Der Datentyp ist BOOLEAN.

FeaturePause-Eigenschaft

FeaturePause

Legt fest, ob im Control die Funktion "Pause" verfügbar ist.

Das Attribut ist mit dem Namen **FeaturePause** dynamisierbar. Der Datentyp ist BOOLEAN.

FeaturePlay-Eigenschaft

FeaturePlay

Legt fest, ob im Control die Funktion "Wiedergabe" verfügbar ist.

Das Attribut ist mit dem Namen **FeaturePlay** dynamisierbar. Der Datentyp ist BOOLEAN.

FeatureStepBackward-Eigenschaft

FeatureStepBackward

Legt fest, ob im Control die Funktion "Rückwärts springen" verfügbar ist.

Das Attribut ist mit dem Namen **FeatureStepBackward** dynamisierbar. Der Datentyp ist BOOLEAN.

FeatureStepForward-Eigenschaft

FeatureStepForward

Legt fest, ob im Control die Funktion "Vorwärts springen" verfügbar ist.

Das Attribut ist mit dem Namen **FeatureStepForward** dynamisierbar. Der Datentyp ist BOOLEAN.

FeatureStop-Eigenschaft

FeatureStop

Legt fest, ob im Control die Funktion "Stop" verfügbar ist.

Das Attribut ist mit dem Namen **FeatureStop** dynamisierbar. Der Datentyp ist BOOLEAN.

FeatureVolume-Eigenschaft

FeatureVolume

Legt fest, ob im Control die Funktion "Lautstärke" verfügbar ist.

Das Attribut ist mit dem Namen **FeatureVolume** dynamisierbar. Der Datentyp ist BOOLEAN.

FileName-Eigenschaft

FileName

Legt die Datei fest, deren Inhalt Sie anzeigen bzw. abspielen wollen.

Das Attribut ist mit dem Namen **FileName** dynamisierbar. Der Datentyp ist STRING.

FillColor-Eigenschaft

Beschreibung

Legt die Farbe des Füllmusters für das Objekt fest oder gibt sie zurück.

LONG (Schreib-Lese-Zugriff)

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an.

Beispiel:

RGB(200, 150, 100)

Beispiel

Das folgende Beispiel setzt die Füllfarbe des Bildes "ScreenWindow1" auf Blau:

```
'VBS73
Dim objScreen
Set objScreen = HMIRuntime.Screens("ScreenWindow1")
objScreen.FillStyle = 131075
```

```
objScreen.FillColor = RGB(0, 0, 255)
```

Siehe auch

Fillstyle-Eigenschaft (Seite 398)

BackColor-Eigenschaft (Seite 316)

ScreenItem-Objekt (Seite 134)

Filling-Eigenschaft

Beschreibung

TRUE, wenn das Objekt mit geschlossener Rahmenlinie gefüllt werden kann (also z.B. den Füllstand eines Tanks darstellt). BOOLEAN Schreib-Lese-Zugriff.
Den Füllstand des Objektes setzen Sie mit der Eigenschaft "FillingIndex".

Siehe auch

ScreenItem-Objekt (Seite 134)

FillingDirection-Eigenschaften

Das Attribut "Füllrichtung" legt fest, in welche Richtung das Objekt mit geschlossener Rahmenlonie gefüllt werden soll.

Unten nach oben	Das Objekt wird von unten nach oben gefüllt.
Oben nach unten	Das Objekt wird von oben nach unten gefüllt.
Links nach rechts	Das Objekt wird von links nach rechts gefüllt.
Rechts nach links	Das Objekt wird von rechts nach links gefüllt.

Das Attribut ist mit dem Namen FillingDirection dynamisierbar. Der Datentyp ist LONG.

FillingIndex-Eigenschaft

Beschreibung

Legt den %-Wert (bezogen auf die Höhe des Objekts) fest, zu dem das Objekt mit geschlossener Rahmenlinie gefüllt wird.
Der Füllstand wird mit der aktuellen Hintergrundfarbe dargestellt. Der nicht gefüllte Hintergrund ist transparent.

Siehe auch

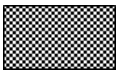
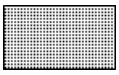
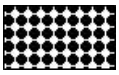
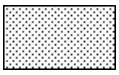
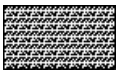
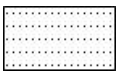
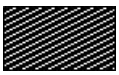
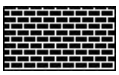
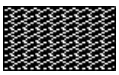
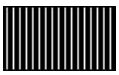
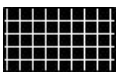
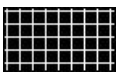
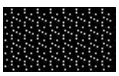
ScreenItem-Objekt (Seite 134)

Fillstyle-Eigenschaft

Beschreibung

Legt das Füllmuster für das Objekt fest oder gibt es zurück.

LONG (Schreib-Lese-Zugriff)

Füllmuster	Wert	Füllmuster	Wert	Füllmuster	Wert
< Transparent >	65536				
< Massiv >	0				
	1048576		196611		196627
	1048577		196612		196628
	1048578		196613		196629
	1048579		196614		196630
	1048832		196615		196631
	1048833		196616		196632
	1048834		196617		196633
	1048835		196618		196634
	131072		196619		196635
	131073		196620		196636
	131074		196621		196637
	131075		196622		196638
	131076		196623		196639
	196608		196624		196640

Füllmuster	Wert	Füllmuster	Wert	Füllmuster	Wert
	196609		196625		196641
	196610		196626		196642

Beispiel

Das folgende Beispiel setzt das Füllmuster des Bildes "Bildfenster1" transparent:

```
'VBS190
Dim obj
Set obj = ScreenItems("Rectangle1")
obj.FillStyle = 65536
```

Siehe auch

FillColor-Eigenschaft (Seite 396)

BackColor-Eigenschaft (Seite 316)

Screen-Objekt (Seite 140)

FillStyle2-Eigenschaft

Beschreibung

Legt das Füllmuster des Balkens fest oder gibt es zurück.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

FillStyleAlignment-Eigenschaft

Beschreibung

Legt die Ausrichtung des Füllmusters für das Prozessbild fest.

Normal	Das Füllmuster bezieht sich auf das Prozessbild. In Runtime wird es beim Aufziehen des Bildes nicht skaliert.
gestreckt (Fenster)	Das Füllmuster bezieht sich auf das Fenster im Graphics Designer. In Runtime wird es beim Aufziehen des Bildes skaliert.

FilterSQL-Eigenschaft

FilterSQL

Legt für eine Selektion der Daten des Anwenderarchivs ein SQL-Statement fest.

Das Attribut ist mit dem Namen **FilterSQL** dynamisierbar. Der Datentyp ist STRING.

FineGrid-Eigenschaft

Beschreibung

TRUE, wenn die Werteachse mit kurzen Teilstrichen skaliert wird. Der Abstand zweier kurzer Teilstriche kann über die Eigenschaft "FineGridValue" verändert werden. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

FineGridValue-Eigenschaft

Beschreibung

Legt den Abstand zweier kurzer Teilstriche der Skalierung fest. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "FineGrid".

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

FineGridValueX-Eigenschaft

Beschreibung

Legt den Abstand zweier kurzer Teilstriche der X-Achsen-Skalierung fest. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "FineGridX".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

FineGridValueY-Eigenschaft

Beschreibung

Legt den Abstand zweier kurzer Teilstriche der Y-Achsen-Skalierung fest. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "FineGridX".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

FineGridX-Eigenschaft

Beschreibung

TRUE, wenn die X-Achse mit kurzen Teilstrichen skaliert wird. Der Abstand zweier kurzer Teilstriche kann über die Eigenschaft "FineGridValueX" verändert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

FineGridY-Eigenschaft

Beschreibung

TRUE, wenn die Y-Achse mit kurzen Teilstrichen skaliert wird. Der Abstand zweier kurzer Teilstriche kann über die Eigenschaft "FineGridValueY" verändert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

FlashBackColor-Eigenschaft

Beschreibung

TRUE, wenn das Blinken des Hintergrunds des Objektes aktiviert ist. BOOLEAN Schreib-Lese-Zugriff

Siehe auch

ScreenItem-Objekt (Seite 134)

FlashBorderColor-Eigenschaft

Beschreibung

TRUE, wenn das Blinken der Linie des Objektes aktiviert ist. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

ScreenItem-Objekt (Seite 134)

FlashFlashPicture-Eigenschaft

Beschreibung

TRUE, wenn das Blinken des Blinkbildes aktiviert ist. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Zustandsanzeige (Seite 210)
ScreenItem-Objekt (Seite 134)

FlashForeColor-Eigenschaft

Beschreibung

TRUE, wenn das Blinken des Textes aktiviert ist. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

EA-Feld (Seite 195)
Statischer Text (Seite 178)
Textliste (Seite 208)
Radio-Box (Seite 217)
Check-Box (Seite 215)
Button (Seite 212)
ScreenItem-Objekt (Seite 134)

FlashPicReferenced-Eigenschaft

Beschreibung

TRUE, wenn das zugeordnete Blinkbild gespeichert werden soll. Sonst wird nur der zugehörige Objektverweis gespeichert. Nur Lese-Zugriff.

Siehe auch

Zustandsanzeige (Seite 210)

ScreenItem-Objekt (Seite 134)

FlashPicTransColor-Eigenschaft

Beschreibung

Legt fest, welche Farbe des dem Blinkbild zugeordneten Bitmap-Objekts (.bmp, .dib) auf "transparent" gesetzt werden soll oder gibt die Farbe zurück. LONG Schreib-Lese-Zugriff. Die Farbe wird nur dann auf "transparent" gesetzt, wenn die Eigenschaft "FlashPicUseTransColor" den Wert TRUE hat.

Siehe auch

ScreenItem-Objekt (Seite 134)

Zustandsanzeige (Seite 210)

FlashPicture-Eigenschaft

Beschreibung

Gibt das Blinkbild zurück. Nur Lese-Zugriff.
Das Bild (*.BMP oder *.DIB) muss sich im Verzeichnis "GraCS" des aktuellen Projekts befinden, damit es eingebunden werden kann.
Die Eigenschaft "FlashPicReferenced" legt in diesem Zusammenhang fest, ob das Blinkbild zusammen mit dem Objekt gespeichert oder referenziert wird.

Siehe auch

Zustandsanzeige (Seite 210)

ScreenItem-Objekt (Seite 134)

FlashPicUseTransColor-Eigenschaft

Beschreibung

TRUE, wenn die projektierte Farbe ("FlashPicTransColor"-Eigenschaft) des dem Blinkbild zugeordneten Bitmap-Objektes auf "transparent" gesetzt werden soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Zustandsanzeige (Seite 210)

ScreenItem-Objekt (Seite 134)

FlashRate-Eigenschaft

Beschreibung

Legt die Blinkfrequenz fest oder gibt sie zurück. Wertebereich von 0 bis 2.

0 = langsam

1 = mittel

2 = schnell

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

FlashRateBackColor-Eigenschaft

Beschreibung

Legt die Blinkfrequenz für den Objekthintergrund fest oder gibt sie zurück. Wertebereich von 0 bis 2.

0 = langsam

1 = mittel

2 = schnell

Siehe auch

ScreenItem-Objekt (Seite 134)

FlashRateBorderColor-Eigenschaft

Beschreibung

Legt die Blinkfrequenz für die Linie des Objektes fest oder gibt sie zurück. Wertebereich von 0 bis 2.

0 = langsam

1 = mittel

2 = schnell

Siehe auch

ScreenItem-Objekt (Seite 134)

FlashRateFlashPic-Eigenschaft

Beschreibung

Legt die Blinkfrequenz für das Blinkbild fest oder gibt sie zurück. Wertebereich von 0 bis 2.

0 = langsam

1 = mittel

2 = schnell

Siehe auch

Zustandsanzeige (Seite 210)

ScreenItem-Objekt (Seite 134)

FlashRateForeColor-Eigenschaft

Beschreibung

Legt die Blinkfrequenz für die Beschriftung des Objektes fest oder gibt sie zurück. Wertebereich von 0 bis 2.

0 = langsam

1 = mittel

2 = schnell

Siehe auch

Statischer Text (Seite 178)

Textliste (Seite 208)

Radio-Box (Seite 217)

Check-Box (Seite 215)

Button (Seite 212)

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

Flip-Eigenschaft

Spiegelung (Flip)

Legt die Spiegelung des Symbols in Runtime fest.

Folgende Einstellungen sind möglich:

Wert	Beschreibung	Erläuterung
0	Kein	Das Symbol wird nicht gespiegelt.
1	Horizontal	Das Symbol wird an der horizontalen Mittelachse gespiegelt.
2	Vertikal	Das Symbol wird an der vertikalen Mittelachse gespiegelt.
3	Beides	Das Symbol wird an der horizontalen und vertikalen Mittelachse gespiegelt.

Das Attribut ist mit dem Namen **Flip** dynamisierbar. Der Datentyp ist LONG.

Flip-Eigenschaft

Beschreibung

Spiegelt das Symbol an der vertikalen und/oder an der horizontalen Mittelachse des Symbols.

- None - 0: Das Symbol wird nicht gespiegelt.
- Horizontal - 1: Das Symbol wird an der vertikalen Mittelachse gespiegelt.
- Vertical - 2: Das Symbol wird an der horizontalen Mittelachse gespiegelt.
- Both - 3: Das Symbol wird sowohl an der horizontalen als auch an der vertikalen Mittelachse gespiegelt.

Siehe auch

HMI Symbol Library (Seite 248)

ScreenItem-Objekt (Seite 134)

Fo - Fr

FocusColor-Eigenschaft

Beschreibung

Wenn der Focus in Runtime auf dem Control liegt, werden die Beschriftung und der Positionstext mit einem Rahmen gekennzeichnet. FocusColor legt die Farbe des Rahmens fest.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

FocusRect-Eigenschaft

Beschreibung

TRUE, wenn in Runtime die Schaltfläche mit einem Selektionsrahmen versehen wird, sobald sie den Fokus erhält. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Push Button Control (Seite 271)

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

FocusWidth-Eigenschaft

Beschreibung

Wenn der Focus in Runtime auf dem Control liegt, werden die Beschriftung und der Positionstext mit einem Rahmen gekennzeichnet. FocusWidth legt die Breite des Rahmens fest, Wertebereich von 1-10 Pixel. Long Schreib-Lese-Zugriff.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

Font-Eigenschaft

Name - Font

Legt die Schriftart fest.

Das Attribut ist nicht dynamisierbar.

Font-Eigenschaft (Vor WinCC V7)

Beschreibung

Legt die Schriftart fest oder gibt sie zurück. Schreib-Lese-Zugriff.

Das Font-Objekt hat die Untereigenschaften

- Size (Schriftgröße)
- Bold (Fett ja/nein)
- Name (Schriftart)
- Italic (Kursiv ja/nein)
- Underline (Unterstrichen ja/nein)
- StrikeThrough (Durchgestrichen ja/nein)

Werden zwei Font-Eigenschaften direkt zugewiesen, wird nur die Default-Eigenschaft "Name" übernommen.

Beispiel

```
'VBS74
Dim objControl1
Dim objControl2
Set objControl1 = ScreenItems("Control1")
Set objControl2 = ScreenItems("Control2")
objControl2.Font = objControl1.Font ' take over only the type of font
```

Siehe auch

WinCC Slider Control (Seite 278)

WinCC Push Button Control (Seite 271)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

WinCC Digital Analog Clock (Seite 255)

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

FontBold-Eigenschaft

Beschreibung

TRUE, wenn der Text im Objekt das Schriftattribut "fett" erhält. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Push Button Control (Seite 271)

Sammelanzeige (Seite 205)

Textliste (Seite 208)

Radio-Box (Seite 217)

Check-Box (Seite 215)

Button (Seite 212)

EA-Feld (Seite 195)

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

FontItalic-Eigenschaft

Beschreibung

TRUE, wenn der Text im Objekt das Schriftattribut "kursiv" erhält. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Push Button Control (Seite 271)
Sammelanzeige (Seite 205)
Statischer Text (Seite 178)
Textliste (Seite 208)
Radio-Box (Seite 217)
Check-Box (Seite 215)
Button (Seite 212)
EA-Feld (Seite 195)
ScreenItem-Objekt (Seite 134)

FontName-Eigenschaft

Beschreibung

Legt die Schriftart des Textes im Objekt fest oder gibt sie zurück.
Dabei stehen Ihnen alle in Windows installierten Schriftarten zur Verfügung.

Siehe auch

WinCC Push Button Control (Seite 271)
Sammelanzeige (Seite 205)
Statischer Text (Seite 178)
Textliste (Seite 208)
Radio-Box (Seite 217)
Check-Box (Seite 215)
Button (Seite 212)
EA-Feld (Seite 195)
Balken (Seite 186)
ScreenItem-Objekt (Seite 134)

FontPosition-Eigenschaft

Beschreibung

Gibt die Schriftart für die Anzeige der Schieberposition im unteren Teil des Objekts zurück.
Dabei stehen Ihnen alle in Windows installierten Schriftarten zur Verfügung. Nur Lese-Zugriff.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

FontSize-Eigenschaft

Beschreibung

Legt die Schriftgröße des Textes im Objekt in Punkt fest oder gibt sie zurück.

Siehe auch

WinCC Push Button Control (Seite 271)

Sammelanzeige (Seite 205)

Statischer Text (Seite 178)

Textliste (Seite 208)

Radio-Box (Seite 217)

Check-Box (Seite 215)

Button (Seite 212)

EA-Feld (Seite 195)

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

FontStrikeThru-Eigenschaft

Beschreibung

TRUE, wenn der Text im Objekt das Schriftattribut "durchgestrichen" erhält. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Push Button Control (Seite 271)

ScreenItem-Objekt (Seite 134)

FontUnderline-Eigenschaft

Beschreibung

TRUE, wenn der Text im Objekt das Schriftattribut "unterstrichen" erhält. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Push Button Control (Seite 271)
Sammelanzeige (Seite 205)
Statischer Text (Seite 178)
Textliste (Seite 208)
Radio-Box (Seite 217)
Check-Box (Seite 215)
Button (Seite 212)
EA-Feld (Seite 195)
ScreenItem-Objekt (Seite 134)

ForeColor-Eigenschaft

Beschreibung

Legt die Farbe des Textes im Objekt fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Slider Control (Seite 278)
WinCC Push Button Control (Seite 271)
WinCC Digital Analog Clock (Seite 255)
HMI Symbol Library (Seite 248)
Statischer Text (Seite 178)
Textliste (Seite 208)
Radio-Box (Seite 217)
Check-Box (Seite 215)
Button (Seite 212)
EA-Feld (Seite 195)
ScreenItem-Objekt (Seite 134)

ForeColor-Eigenschaft

Vordergrundfarbe (ForeColor)

Legt über den Dialog "Farbauswahl" die Vordergrundfarbe des Symbols fest. Im Vordergrundmodus "Schattiert" und "Massiv" wird das Symbol in der Vordergrundfarbe angezeigt.

Das Attribut ist mit dem Namen **ForeColor** dynamisierbar. Der Datentyp ist LONG.

ForeFlashColorOff-Eigenschaft

Beschreibung

Legt die Farbe des Textes für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Textliste (Seite 208)

Statischer Text (Seite 178)

Radio-Box (Seite 217)

Check-Box (Seite 215)

Button (Seite 212)

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

ForeFlashColorOn-Eigenschaft

Beschreibung

Legt die Farbe des Textes für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Statischer Text (Seite 178)

Textliste (Seite 208)

Radio-Box (Seite 217)

Check-Box (Seite 215)

Button (Seite 212)

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

FrameColor-Eigenschaft

Beschreibung

Legt die Farbe des rechteckigen bzw. quadratischen Bereichs fest, auf dem die Skalenscheibe liegt, oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

FrameColorDown-Eigenschaft

Beschreibung

Legt die Farbe für den rechten, unteren Teil des 3D-Rahmens der Schaltfläche fest (Schaltfläche gedrückt) oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Push Button Control (Seite 271)

ScreenItem-Objekt (Seite 134)

FrameColorUp-Eigenschaft

Beschreibung

Legt die Farbe für den linken, oberen Teil des 3D-Rahmens der Schaltfläche fest (Schaltfläche nicht gedrückt) oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Push Button Control (Seite 271)

ScreenItem-Objekt (Seite 134)

FramePicture-Eigenschaft

Beschreibung

Gibt den Bildnamen des Hintergrundbilds für die Skalenscheibe zurück. Nur Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

FrameScale-Eigenschaft

Beschreibung

Legt den Durchmesser der Skalenscheibe in Bezug auf den kleineren Wert der Geometrieattribute Breite und Höhe fest oder gibt ihn zurück. Schreib-Lese-Zugriff.

Der Wertebereich ist (Skalenabstand - Skalenbreite) bis 1.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

FrameWidth-Eigenschaft

Beschreibung

Legt die Rahmenbreite der Schaltfläche in Pixel fest oder gibt sie zurück. Schreib-Lese-Zugriff.

Siehe auch

WinCC Push Button Control (Seite 271)

ScreenItem-Objekt (Seite 134)

FreezeProviderConnections-Eigenschaft

Beschreibung

Ermöglicht das Ändern der Eigenschaften zur Datenanbindung ("ProviderType", "Source"...), ohne dass die Änderung sofort wirksam wird. Beim Verändern von z.B. "SourceTagNameX" können unzulässige Kombinationen mit "SourceTagNameY" entstehen.

Deshalb muss vor dem Ändern eines Attributs zur Datenanbindung "FreezeProviderConnections" auf TRUE gesetzt werden. Nach der Änderung aller Eigenschaften zur Datenanbindung wird "FreezeProviderConnection" auf FALSE gesetzt, und die Änderungen werden wirksam.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

1.14.4.8 G

GlobalColorScheme-Eigenschaft

Beschreibung

Legt fest, ob die im globalen Farbschema des aktuellen Designs definierten Farben für dieses Objekt verwendet werden.

TRUE, wenn das Objekt mit den Farben aus dem für diesen Objekttyp festgelegten globalen Farbschema dargestellt wird.

FALSE, wenn das Objekt mit den Farben entsprechend den Einstellungen im Objekt dargestellt wird.

BOOLEAN Schreib-Lese-Zugriff.

GlobalShadow-Eigenschaft

Beschreibung

Legt fest, ob das Objekt mit der im aktiven Design festgelegten Schattierung dargestellt wird.

TRUE, wenn das Objekt mit der für diesen Objekttyp festgelegte globale Schattierung dargestellt wird.

FALSE, wenn kein Schatten dargestellt wird.

BOOLEAN Schreib-Lese-Zugriff.

GraphDirection-Eigenschaft (vor WinCC V7)

Beschreibung

Legt fest, an welchem Rand des Kurvenfensters die aktuellen Werte angezeigt werden sollen. Schreib-Lese-Zugriff.

0: Positive Werte werden nach rechts und nach oben abgetragen.

-1: Positive Werte werden nach links und nach oben abgetragen.

-2: Positive Werte werden nach rechts und nach oben abgetragen.

-3: Positive Werte werden nach rechts und nach unten abgetragen.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

GraphDirection-Eigenschaft

Schreiberrichtung - GraphDirection

Legt die Richtung fest, in die die Werte auf der Achse aktualisiert werden.

Wert	Beschreibung	Erläuterung
0	von rechts	Die aktualisierten Werte werden rechts im Control dargestellt..
1	von links	Die aktualisierten Werte werden links im Control dargestellt.
2	von oben	Die aktualisierten Werte werden oben im Control dargestellt.
3	von unten	Die aktualisierten Werte werden unten im Control dargestellt.

Wenn Sie für die Schreiberrichtung die Einstellungen "von oben" oder "von unten" wählen, müssen Sie innerhalb des Kurvenfensters True-Type-Schriftarten verwenden. Nur so ist gewährleistet, dass die Beschriftung der vertikalen Achse gut lesbar ist.

Das Attribut ist mit dem Namen **GraphDirection** dynamisierbar. Der Datentyp ist LONG.

GridLineColor-Eingenschaft

Farbe der Trennlinie / Inhalt - GridLineColor

Gibt die Farbe der Trennlinien im Tabelleninhalt an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **GridLineColor** dynamisierbar. Der Datentyp ist LONG.

GridLineHorz-Eigenschaft

Beschreibung

TRUE, wenn die Spalten des Meldefensters durch horizontale Trennlinien getrennt werden. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

GridLines-Eigenschaft

Beschreibung

TRUE, wenn das Kurvenfenster mit zur x-Achse parallelen Rasterlinien dargestellt wird. Der Abstand zweier Rasterlinien kann über die Eigenschaft "GridLineValue" verändert werden. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

GridlinesValueX-Eigenschaft

Beschreibung

Legt den Abstand zweier Rasterlinien der X-Achse fest oder gibt ihn zurück. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "GridLinesX".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

GridlinesValueY-Eigenschaft

Beschreibung

Legt den Abstand zweier Rasterlinien der Y-Achse fest oder gibt ihn zurück. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "GridLinesY".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

GridlinesX-Eigenschaft

Beschreibung

TRUE, wenn das Kurvenfenster mit zur X-Achse parallelen Rasterlinien dargestellt wird. Der Abstand zweier Rasterlinien kann über die Eigenschaft "GridLinesValueX" verändert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

GridlinesY-Eigenschaft

Beschreibung

TRUE, wenn das Kurvenfenster mit zur Y-Achse parallelen Rasterlinien dargestellt wird. Der Abstand zweier Rasterlinien kann über die Eigenschaft "GridLinesValueX" verändert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

GridLineValue-Eigenschaft

Beschreibung

Legt den Abstand zweier Rasterlinien fest. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "GridLines".

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

GridLineVert-Eigenschaft

Beschreibung

TRUE, wenn die Spalten des Meldefensters durch vertikale Trennlinien getrennt werden. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

GridLineWidth-Eigenschaft

Stärke Trennlinien - GridLineWidth

Legt die Stärke der Trennlinien in Pixel fest.

Das Attribut ist mit dem Namen **GridLineWidth** dynamisierbar. Der Datentyp ist LONG.

1.14.4.9 H

Ha - Hi

HandFillColor-Eigenschaft

Beschreibung

Legt die Füllfarbe aller Zeiger der Analoguhr fest oder gibt sie zurück. Damit die Zeiger mit der eingestellten Füllfarbe dargestellt werden, muss die Eigenschaft "Handtype" auf "0" (deckend) eingestellt sein. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

Handtype-Eigenschaft

Beschreibung

Legt die Darstellung der Zeiger fest:

- 0: Die Zeiger werden ausgefüllt in Zeigerfüllfarbe und mit Rand in Vordergrundfarbe dargestellt.
- 1: Die Zeiger erscheinen transparent und werden durch einen Rand in Vordergrundfarbe dargestellt.

Siehe auch

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

HeaderSort-Eigenschaft

Beschreibung

Legt fest, ob das Sortieren der Meldeblöcke über die Spaltenüberschrift der Meldeblöcke möglich ist.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

Height-Eigenschaft

Beschreibung

Legt die Höhe des Objektes in Pixel fest oder gibt sie zurück.

LONG (Schreib-Lese-Zugriff)

Beispiel

Das folgende Beispiel halbiert die Höhe aller Objekte des Bildes "NewPDL1", deren Name mit "Circle" beginnt:

```
'VBS75
Dim objScreen
Dim objCircle
Dim lngIndex
Dim strName
lngIndex = 1
Set objScreen = HMIRuntime.Screens("NewPDL1")
For lngIndex = 1 To objScreen.ScreenItems.Count
'
'Searching all circles
strName = objScreen.ScreenItems.Item(lngIndex).ObjectName
If "Circle" = Left(strName, 6) Then
'
'to halve the height of the circles
Set objCircle = objScreen.ScreenItems(strName)
objCircle.Height = objCircle.Height / 2
End If
Next
```

Siehe auch

Width-Eigenschaft (Seite 673)

Objekt-Typen des Objekts ScreenItem (Seite 152)

ScreenItem-Objekt (Seite 134)

HiddenInput-Eigenschaft

Beschreibung

TRUE, wenn der Eingabewert während der Eingabe nicht angezeigt wird. Für jedes Zeichen wird ein * angezeigt. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

HideTagNames-Eigenschaft

Beschreibung

TRUE, wenn der Archiv- und Variablennamen in der Kurve über die rechte Maustaste, in der Statuszeile und in der Tabelle zur Darstellung der Koordinaten nicht angezeigt wird. BOOLEAN Schreib-Lese-Zugriff.

HitlistColumnAdd-Eigenschaft

HitlistColumnAdd

Übernimmt den ausgewählten Meldeblock aus der Liste der vorhandenen Meldeblöcke in die Liste der gewählten Meldeblöcke.

Das Attribut ist mit dem Namen **HitlistColumnAdd** dynamisierbar. Der Datentyp ist STRING.

HitlistColumnCount-Eigenschaft

HitlistColumnCount

Gibt die Anzahl der gewählten Meldeblöcke an, die in der Hitliste in Runtime angezeigt werden.

Das Attribut ist mit dem Namen **HitlistColumnCount** dynamisierbar. Der Datentyp ist LONG.

HitlistColumnIndex-Eigenschaft

HitlistColumnIndex

Referenziert einen für die Hitliste gewählten Meldeblock. Unter Verwendung des Attributs können Sie einem bestimmten Meldeblock der Hitliste die Werte anderer Attribute zuweisen.

Gültige Werte für "HitlistColumnIndex" liegen zwischen 0 und "HitlistColumnCount" minus 1. Das Attribut "HitlistColumnCount" gibt die Anzahl der für die Hitliste gewählten Meldeblöcke an.

Das Attribut "HitlistColumnIndex" ist über das Attribut **HitlistColumnRepos** dynamisierbar. Der Datentyp ist LONG.

HitlistColumnName-Eigenschaft

HitlistColumnName

Zeigt den Namen des Meldeblocks der Hitliste an, der mit dem Attribut "HitlistColumnIndex" referenziert wird. Den Namen können Sie nicht ändern.

Das Attribut ist mit dem Namen **HitlistColumnName** dynamisierbar. Der Datentyp ist STRING.

HitlistColumnRemove-Eigenschaft

HitlistColumnRemove

Entfernt den markierten Meldeblock aus der Liste der gewählten Meldeblöcke und fügt ihn in die Liste der vorhandenen Meldeblöcke ein.

Das Attribut ist mit dem Namen **HitlistColumnRemove** dynamisierbar. Der Datentyp ist STRING.

HitlistColumnRepos

Auf/Ab - MessageColumnRepos/HitlistColumnRepos

Ändert die Reihenfolge der Meldeblöcke. "Auf" und "Ab" bewegen den ausgewählten Meldeblock in der Liste nach oben oder unten. Dadurch wird in Runtime der Meldeblock im Control weiter vorne oder hinten platziert.

Das Attribut für die Hitliste ist mit dem Namen **HitlistColumnRepos** dynamisierbar.

Das Attribut für die Meldeliste ist mit dem Namen **MessageColumnRepos** dynamisierbar.

Der Datentyp ist LONG.

HitlistColumnSort-Eigenschaft

HitlistColumnSort

Legt fest, wie der im "HitlistColumnIndex" referenzierte Meldeblock für die Hitliste sortiert wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	keine	Keine Sortierung
1	aufsteigend	Aufsteigende Sortierung vom kleinsten zum größten Wert.
2	absteigend	Absteigende Sortierung vom größten zum kleinsten Wert.

Das Attribut ist mit dem Namen **HitlistColumnSort** dynamisierbar. Der Datentyp ist LONG.

HitlistColumnSortIndex-Eigenschaft

HitlistColumnSortIndex

Gibt die Sortierreihenfolge des im "HitlistColumnIndex" referenzierten Meldeblocks der Hitliste an. Wenn Sie den Wert auf "0" setzen, wird das Sortierkriterium in "HitlistColumnSort" entfernt.

Das Attribut ist mit dem Namen **HitlistColumnSortIndex** dynamisierbar. Der Datentyp ist LONG.

HitlistColumnVisible

Gewählte Meldeblöcke - MessageColumnVisible/HitlistColumnVisible

Ausgewählte Meldeblöcke der Meldeliste bzw. Hitliste, die in Runtime angezeigt werden. Legt fest, ob der im "MessageColumnIndex" bzw. "HitlistColumnIndex" referenzierte Meldeblock angezeigt wird.

Das Attribut für die Meldeliste ist mit dem Namen **MessageColumnVisible** dynamisierbar.

Das Attribut für die Hitliste ist mit dem Namen **HitlistColumnVisible** dynamisierbar.

Der Datentyp ist BOOLEAN.

HitlistDefaultSort-Eigenschaft

HitlistDefaultSort

Legt die Standardsortierung in den Tabellenspalten der Hitliste fest.

Folgende Einstellungen sind möglich:

Wert	Beschreibung	Erläuterung
0	aufsteigend	Die Liste wird aufsteigend nach Häufigkeit sortiert.
1	absteigend	Die Liste wird absteigend nach Häufigkeit sortiert.

Das Attribut ist mit dem Namen **HitlistDefaultSort** dynamisierbar. Der Datentyp ist LONG.

HitListMaxSourceItems-Eigenschaft

Maximale Anzahl Datensätze - HitListMaxSourceItems

Legt die maximale Anzahl Datensätze für die Statistik fest.

Das Attribut ist mit dem Namen **HitListMaxSourceItems** dynamisierbar. Der Datentyp ist LONG.

HitListMaxSourceItemsWarn-Eigenschaft

Warnhinweis - HitListMaxSourceItemsWarn

Legt fest, ob ein Warnhinweis erscheint, wenn das zulässige Maximum der Datensätze erreicht ist.

Wert	Erklärung
TRUE	Ein Warnhinweis erscheint, wenn das zulässige Maximum der Datensätze erreicht ist.
FALSE	Kein Warnhinweis erscheint, wenn das zulässige Maximum der Datensätze erreicht ist.

Das Attribut ist mit dem Namen **HitListMaxSourceItemsWarn** dynamisierbar. Der Datentyp ist BOOLEAN.

HitListRelTime-Eigenschaft

Zeitbereich für die Statistik - HitListRelTime

Legt fest, ob ein Zeitbereich für die Statistik verwendet wird.

Wert	Erklärung
TRUE	Wenn kein Zeitbereich in der Selektion angegeben ist, wird der angegebene Zeitbereich für die Statistik verwendet.
FALSE	Der angegebene Zeitbereich wird nicht verwendet.

Das Attribut ist mit dem Namen **HitListRelTime** dynamisierbar. Der Datentyp ist BOOLEAN.

HitListRelTimeFactor-Eigenschaft

Zeitbereich - HitListRelTimeFactor

Legt den Faktor zur Bestimmung des Zeitbereichs fest. Nur ganzzahlige Faktoren sind zulässig.

Das Attribut ist mit dem Namen **HitListRelTimeFactor** dynamisierbar. Der Datentyp ist LONG.

HitListRelTimeFactorType-Eigenschaft

Zeitbereich - HitListRelTimeFactorType

Legt die Zeiteinheit zur Bestimmung des Zeitbereichs fest.

Folgende Zeiteinheiten stehen zur Verfügung:

Wert	Beschreibung
0	Minute
1	Stunde

Wert	Beschreibung
2	Tag
3	Woche
4	Monat

Das Attribut ist mit dem Namen **HitListMaxRelTimeFactorType** dynamisierbar. Der Datentyp ist LONG.

Ho - Hy

HorizontalGridLines-Eigenschaft

Horizontal - HorizontalGridLines

Legt fest, ob horizontale Trennlinien angezeigt werden.

Wert	Erklärung
TRUE	Horizontale Trennlinien werden angezeigt.
FALSE	Horizontale Trennlinien werden nicht angezeigt.

Das Attribut ist mit dem Namen **HorizontalGridLines** dynamisierbar. Der Datentyp ist BOOLEAN.

Hotkey-Eigenschaft

Beschreibung

Gibt beim Objekt Button die Funktionstaste für die Mausbedienung zurück.

Nur Lese-Zugriff.

Siehe auch

Button (Seite 212)

ScreenItem-Objekt (Seite 134)

HourNeedleHeight-Eigenschaft

Beschreibung

Legt die Länge des Stundenzeigers der Analoguhr fest oder gibt sie zurück. Die Angabe der Länge erfolgt in Prozent, bezogen auf die halbe Länge der kürzeren Seite des rechteckigen Hintergrunds. Schreib-Lese-Zugriff.

Beispiel:
Die kürzere Seite des rechteckigen Hintergrunds sei 100 Pixel lang.

Die Stundenzeigerlänge sei mit 50 angegeben.
Daraus ergibt sich die Länge des Stundenzeigers zu (100 Pixel / 2) * 0,5 = 25 Pixel.

Siehe auch

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

HourNeedleWidth-Eigenschaft

Beschreibung

Legt die Breite des Stundenzeigers der Analoguhr fest oder gibt sie zurück. Die Angabe der Breite erfolgt in Prozent, bezogen auf die doppelte Länge des Stundenzeigers. Schreib-Lese-Zugriff.

Beispiel:
Die Länge des Stundenzeigers sei 25 Pixel.
Die Stundenzeigerbreite sei mit 10 angegeben.
Daraus ergibt sich die Breite des Stundenzeigers zu 25 Pixel * 2 * 0,1 = 5 Pixel.

Siehe auch

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

Hysteresis-Eigenschaft

Beschreibung

TRUE, wenn die Anzeige mit Hysterese erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

HysteresisRange-Eigenschaft

Beschreibung

Legt die Hysterese in % des Anzeigewertes fest oder gibt sie zurück.
Die Eigenschaft Hysteresis muss den Wert TRUE haben, damit die Hysterese berechnet werden kann.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

1.14.4.10 I

IconSpace-Eigenschaft

IconSpace

Legt den Abstand zwischen Symbol und Text in den Zellen der Tabelle fest. Der Wert ist wirksam, wenn Symbol und Text angezeigt werden.

Das Attribut ist mit dem Namen **IconSpace** dynamisierbar. Der Datentyp ist LONG.

IndependentWindow-Eigenschaft

Beschreibung

Legt fest, ob die Darstellung des Bildfensters in Runtime von dem Prozessbild abhängt, in dem das Bildfenster projektiert wurde.

TRUE, wenn Größe und Position des Bildfensters unabhängig vom Prozessbild und nur durch das Attribut "Fenstermodus" festgelegt sind.

FALSE, wenn sich Größe und Position des Bildfensters mit der Verschiebung oder Skalierung des Prozessbildes ändern.

Index-Eigenschaft

Beschreibung

Check-Box, Radio-Box

Legt die Nummer (0 bis 31) des Feldes fest, dessen Text Sie definieren wollen, oder gibt sie zurück.

Kombinationsfeld, Listenfeld

Legt die Nummer (0 bis 31) der Zeile fest, deren Text Sie definieren wollen, oder gibt sie zurück.

Polygon, Polygonzug, Rohrpolygon

Legt die Nummer des Eckpunkts fest oder gibt sie zurück, dessen Positionskoordinaten Sie ändern oder anzeigen lassen wollen.

WinCC Online Trend Control, WinCC Online Table Control, WinCC Function Trend Control

Die Eigenschaft "Index" wird von anderen Eigenschaften ausgewertet, um deren Einstellungen einer bestimmten Kurve bzw. einem bestimmten Spaltenpaar zuordnen zu können. Gültige Werte für den Index bewegen sich im Bereich von 0 bis (NumItems - 1). Die Eigenschaft "NumItems" enthält die Anzahl der anzuzeigenden Kurven/Spaltenpaare. Der Index muss immer gesetzt werden, bevor Sie die Eigenschaften einer Kurve/Spalte in Runtime ändern.

Zustandsanzeige

Legt den Zustand (0 bis 255) fest oder gibt ihn zurück. Für jeden Zustandswert können Sie ein Grund- und ein Blinkbild angeben.

Siehe auch

Zustandsanzeige (Seite 210)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

Polygonzug (Seite 170)

Polygon (Seite 168)

Radio-Box (Seite 217)

Check-Box (Seite 215)

ScreenItem-Objekt (Seite 134)

InnerBevelOffset-Eigenschaft

Beschreibung

Legt den Abstand zwischen Innenrahmen und Außenrahmen fest.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

InnerBevelStyle-Eigenschaft

Beschreibung

Legt den 3D-Effekt für den Innenrahmen des Objekts fest.

- 0: Kein Rahmen.
- 1: Der Rahmen wird vertieft dargestellt.

- 2: Der Rahmen wird erhaben dargestellt.
- 3: Der Rahmen wird einfarbig, ohne 3D-Effekt dargestellt. Die Rahmenfarbe wird durch die Eigenschaft "BevelColorDown" bestimmt.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

InnerBevelWidth-Eigenschaft

Beschreibung

Legt die Breite des Innenrahmens in Pixel fest.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

InputValue-Eigenschaft

Beschreibung

Definiert den im EA-Feld vom Benutzer eingegebenen Wert. Der Wert wird beim Setzen der Eigenschaft nicht im EA-Feld angezeigt.

Wenn Sie wollen, dass der eingegebene Wert nach Bestätigung mit der Taste <Return> im EA-Feld angezeigt wird, projektieren Sie eine Direktverbindung zwischen den Eigenschaften "Eingabewert" und "Ausgabewert". Die Direktverbindung ist nur sinnvoll, wenn am Ausgabewert keine Variablenverbindung projektiert ist, der Benutzer aber trotzdem den eingegebenen Wert abfragen möchte, z.B. über einen Skript.

LONG Schreib-Lese-Zugriff.

Siehe auch

Beispiel: So rufen Sie Methoden eines ActiveX-Controls auf (Seite 810)

InsertData-Eigenschaft

Beschreibung

Fügt Daten für die aktuelle Kurve ein.

TRUE: "DataIndex" wird ignoriert und die Daten werden im Datenpuffer hinten angefügt.

FALSE: Die Daten werden an die Position "DataIndex" des Datenbuffers eingefügt.

Bei jeder Operation mit "Insert Data" wird das Kurvenfenster neu gezeichnet.

Hinweis

Die Eigenschaft wird nur für die Controls vor WinCC V7 unterstützt.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

Instance-Eigenschaft

Beschreibung

Gibt eine Instanz des Alarm-Objekts zurück.

Siehe auch

Alarms-Objekt (Auflistung) (Seite 120)

ItemBorderBackColor-Eigenschaft

Beschreibung

Legt die Hintergrundfarbe der Trennlinien in der Auswahlliste des Objektes Textliste fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff. Die Hintergrundfarbe ist nur sichtbar, wenn die Eigenschaft ItemBorderStyle > 0 gesetzt ist.

Siehe auch

Textliste (Seite 208)

ScreenItem-Objekt (Seite 134)

ItemBorderColor-Eigenschaft

Beschreibung

Legt die Farbe der Trennlinien in der Auswahlliste des Objektes Textliste fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Textliste (Seite 208)

ScreenItem-Objekt (Seite 134)

ItemBorderStyle-Eigenschaft

Beschreibung

Legt die Trennlinienart der Auswahlliste des Objektes Textliste fest oder gibt sie zurück. Wertebereich von 0 bis 4.

0 = durchgezogene Linie

1 = gestrichelte Linie

2 = gepunktete Linie

3 = strichpunktierte Linie

4 = Strich-Punkt-Punkt-Linie

Siehe auch

Textliste (Seite 208)

ScreenItem-Objekt (Seite 134)

ItemBorderWidth-Eigenschaft

Beschreibung

Legt die Trennlinienstärke der Auswahlliste des Objektes Textliste in Pixel fest oder gibt sie zurück.

Siehe auch

Textliste (Seite 208)

ScreenItem-Objekt (Seite 134)

ItemProviderClsid-Eigenschaft

Beschreibung

"ItemProviderClsid" legt fest, ob die über "Index" referenzierte Kurve im Kurven-Control entweder mit einer Archivvariable oder mit einer Online-Variable verbunden ist.

Achtung: Wenn Sie die Eigenschaft "ProviderClsid" mit einem Wert versehen, überschreiben Sie die kurvenspezifische Eigenschaft "ItemProviderClsid".

- {416A09D2-8B5A-11D2-8B81-006097A45D48}: Die Kurve ist mit einer Archivvariablen verbunden.
- {A3F69593-8AB0-11D2-A440-00A0C9DBB64E}: Die Kurve ist mit einerOnline-Variablen verbunden.

Wenn die Kurven mit Archiv- und Online-Variablen versorgt werden, gibt die Eigenschaft "ProviderClsid" den Wert "{00000000-0000-0000-0000-000000000000}" zurück.

ItemVisible-Eigenschaft

Beschreibung

TRUE, wenn eine über die Eigenschaft "Index" referenzierte Kurve bzw. ein über "Index" referenziertes Spaltenpaar sichtbar ist. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

1.14.4.11 L

Lab - Las

Label-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. Abhängig vom Wert der Eigenschaft "TimeAxis" wird über Label die Bezeichnung der Zeitachse bzw. der Werteachse festgelegt.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

LabelColor-Eigenschaft

Beschreibung

Legt die Farbe der Skalenbeschriftung fest.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

LabelX-Eigenschaft

Beschreibung

Legt, abhängig vom Wert von "TimeAxisX", die Bezeichnung der X-Achse einer über "Index" referenzierten Kurve fest oder gibt sie zurück. Schreib-Lese-Zugriff.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

LabelY-Eigenschaft

Beschreibung

Legt, abhängig vom Wert von "TimeAxisY", die Bezeichnung der Y-Achse einer über "Index" referenzierten Kurve fest oder gibt sie zurück. Schreib-Lese-Zugriff.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

LanguageSwitch-Eigenschaft

Beschreibung

Gibt den Wert zurück, der festlegt, wo die sprachabhängigen Zuordnungstexte abgelegt werden. Nur Lese-Zugriff.

TRUE, wenn die Texte in der Textbibliothek verwaltet werden. Übersetzungen in andere Sprachen erfolgen in der Textbibliothek.

FALSE, wenn die Texte direkt im Objekt verwaltet werden. Übersetzungen in andere Sprachen können mittels Text Distributor vorgenommen werden.

Siehe auch

Textliste (Seite 208)

ScreenItem-Objekt (Seite 134)

Language-Eigenschaft

Beschreibung

Setzt die aktuelle Runtime-Sprache oder liest sie aus.

Die Runtime-Sprache geben Sie in VBS mit einer Länder-Kennung an, z.B. 1031 für Deutsch - Standard, 1033 für Englisch - USA etc. Eine Übersicht aller Länder-Kennungen finden Sie in den Grundlagen von VBScript unter dem Thema "Gebietsschema-ID (LCID)-Diagramm".

INTEGER (Schreib-Lese-Zugriff)

Beispiel

Das folgende Beispiel setzt die Datensprache auf Deutsch:

```
'VBS76
HMIRuntime.Language = 1031
```

Siehe auch

HMIRuntime-Objekt (Seite 127)

LastError-Eigenschaft

Beschreibung

Gibt einen Fehlercode über den Erfolg der letzten Operation zurück, z.B. Informationen zum Schreib- oder Lesevorgang einer Variablen. Mit der Eigenschaft "QualityCode" kann eine Aussage über die Qualität des gelieferten Wertes gemacht werden. Eine Beschreibung des Fehlers erhält man über die Eigenschaft "ErrorDescription".

LONG (readonly)

Folgende Fehlercodes sind definiert:

Code in Hexadezimalschreibweise	Beschreibung
0x00000000	OK
0x80040001	Durchführungsfehler
0x80040002	Variablenfehler
0x80040003	Server nicht verfügbar
0x80040004	Multi Tag Error (Fehler bei einer oder mehreren Variablen)

Damit LastError einen Wert zurückgibt, muss zuvor ein Read durchgeführt werden.

Tritt beim Lesen oder Schreiben mehrerer Variablen über das TagSet-Objekt ein Fehler auf, so wird der Fehler "Multi Tag Error" gesetzt. Um zu ermitteln, bei welcher Variable ein Fehler

auftrat und welcher Art er war, muss die LastError-Eigenschaft jeder Variablen ausgewertet werden.

Beispiel

Das folgende Beispiel gibt den Fehlercode für die Variable "Tag1" aus:

```
'VBS77
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
objTag.Read
MsgBox objTag.LastError
```

Das folgende Beispiel fügt der TagSet-Auflistung zwei Variablen hinzu und gibt die LastError-Eigenschaft als Trace aus.

```
'VBS178
Dim group
Set group = HMIRuntime.Tags.CreateTagSet
group.Add "Motor1"
group.Add "Motor2"
HMIRuntime.Trace "LastError: " & group.LastError & vbNewLine
```

Auf die LastError-Eigenschaft einer in der Auflistung enthaltenen Variable kann wie folgt zugegriffen werden:

```
HMIRuntime.Trace "LastError: " & group("Motor1").LastError & vbNewLine
```

Siehe auch

TagSet-Objekt (Auflistung) (Seite 151)

QualityCode-Eigenschaft (Seite 523)

ErrorDescription-Eigenschaft (Seite 389)

Tag-Objekt (Seite 146)

Layer

Layer-Eigenschaft

Beschreibung

Gibt die Ebene (Layer) im Bild zurück, in der sich ein Objekt befindet. Insgesamt stehen 32 Ebenen zur Verfügung, wobei Ebene "0" die unterste und Ebene "31" die oberste Ebene ist.

Innerhalb einer Ebene liegen die zuerst projektierten Objekte im Hintergrund.

LONG (readonly)

Hinweis

Die Layer-Eigenschaft, gibt die Ebene aus, in der sich das Objekt befindet. Die Ebene "0" wird als Ebene"0" ausgegeben.

Beim Zugriff werden in VBS die Ebenen von 1 an gezählt. Deshalb muss die Ebene "1" mit layers(2) angesprochen werden.

Beispiel

Das folgende Beispiel gibt für alle Objekte des Bildes "NewPDL1" Name und Ebene aus:

```
'VBS78
Dim objScreen
Dim objScrItem
Dim lngAnswer
Dim lngIndex
Dim strName
lngIndex = 1
Set objScreen = HMIRuntime.Screens("NewPDL1")
For lngIndex = 1 To objScreen.ScreenItems.Count
strName = objScreen.ScreenItems.Item(lngIndex).ObjectName
Set objScrItem = objScreen.ScreenItems(strName)
lngAnswer = MsgBox(strName & " is in layer  " & objScrItem.Layer,vbOKCancel)
If vbCancel = lngAnswer Then Exit For
Next
```

Siehe auch

ScreenItem-Objekt (Seite 134)

Layer00Checked-Eigenschaft

Beschreibung

TRUE, wenn die Grenze 0 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff. Grenzwert und Darstellung werden mit den Eigenschaften Layer00Value und Layer00Color festgelegt.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer01Checked-Eigenschaft

Beschreibung

TRUE, wenn die Grenze 1 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff. Grenzwert und Darstellung werden mit den Eigenschaften Layer01Value und Layer01Color festgelegt.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer02Checked-Eigenschaft

Beschreibung

TRUE, wenn die Grenze 2 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff. Grenzwert und Darstellung werden mit den Eigenschaften Layer02Value und Layer02Color festgelegt.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer03Checked-Eigenschaft

Beschreibung

TRUE, wenn die Grenze 3 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff. Grenzwert und Darstellung werden mit den Eigenschaften Layer03Value und Layer03Color festgelegt.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer04Checked-Eigenschaft

Beschreibung

TRUE, wenn die Grenze 4 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff. Grenzwert und Darstellung werden mit den Eigenschaften Layer04Value und Layer04Color festgelegt.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer05Checked-Eigenschaft

Beschreibung

TRUE, wenn die Grenze 5 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff. Grenzwert und Darstellung werden mit den Eigenschaften Layer05Value und Layer05Color festgelegt.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer06Checked-Eigenschaft

Beschreibung

TRUE, wenn die Grenze 6 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff. Grenzwert und Darstellung werden mit den Eigenschaften Layer06Value und Layer06Color festgelegt.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer07Checked-Eigenschaft

Beschreibung

TRUE, wenn die Grenze 7 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff. Grenzwert und Darstellung werden mit den Eigenschaften Layer07Value und Layer07Color festgelegt.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer08Checked-Eigenschaft

Beschreibung

TRUE, wenn die Grenze 8 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff. Grenzwert und Darstellung werden mit den Eigenschaften Layer08Value und Layer08Color festgelegt.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer09Checked-Eigenschaft

Beschreibung

TRUE, wenn die Grenze 9 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff. Grenzwert und Darstellung werden mit den Eigenschaften Layer09Value und Layer09Color festgelegt.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer10Checked-Eigenschaft

Beschreibung

TRUE, wenn die Grenze 10 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff. Grenzwert und Darstellung werden mit den Eigenschaften Layer10Value und Layer10Color festgelegt.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer00Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 0 fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff. Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer00Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer01Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 1 fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff. Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer01Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer02Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 2 fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff. Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer02Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer03Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 3 fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff. Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer03Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer04Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 4 fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff. Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer04Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer05Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 5 fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff. Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer05Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer06Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 6 fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff. Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer06Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer07Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 7 fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff. Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer07Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer08Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 8 fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff. Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer08Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer09Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 9 fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff. Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer09Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer10Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 10 fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff. Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer10Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer00FillColor-Eigenschaft

Balkenfüllfarbe 0 (Layer00FillColor)

Das Attribut "Balkenfüllfarbe 0" definiert die Farbe, mit welcher der Balken bezüglich der "Grenze 0" gefüllt wird.

Das Attribut "Balkenfüllfarbe 0" ist dynamisierbar mit dem Namen "Layer00FillColor".

Layer01FillColor-Eigenschaft

Balkenfüllfarbe 1 (Layer01FillColor)

Das Attribut "Balkenfüllfarbe 1" definiert die Farbe, mit welcher der Balken bezüglich der "Grenze 1" gefüllt wird.

Das Attribut "Balkenfüllfarbe 1" ist dynamisierbar mit dem Namen "Layer01FillColor".

Layer02FillColor-Eigenschaft

Balkenfüllfarbe 2 (Layer02FillColor)

Das Attribut "Balkenfüllfarbe 2" definiert die Farbe, mit welcher der Balken bezüglich der "Grenze 2" gefüllt wird.

Das Attribut "Balkenfüllfarbe 2" ist dynamisierbar mit dem Namen "Layer02FillColor".

Layer03FillColor-Eigenschaft

Balkenfüllfarbe 3 (Layer03FillColor)

Das Attribut "Balkenfüllfarbe 3" definiert die Farbe, mit welcher der Balken bezüglich der "Grenze 3" gefüllt wird.

Das Attribut "Balkenfüllfarbe 3" ist dynamisierbar mit dem Namen "Layer03FillColor".

Layer04FillColor-Eigenschaft

Balkenfüllfarbe 4 (Layer04FillColor)

Das Attribut "Balkenfüllfarbe 4" definiert die Farbe, mit welcher der Balken bezüglich der "Grenze 4" gefüllt wird.

Das Attribut "Balkenfüllfarbe 4" ist dynamisierbar mit dem Namen "Layer04FillColor".

Layer05FillColor-Eigenschaft

Balkenfüllfarbe 5 (Layer05FillColor)

Das Attribut "Balkenfüllfarbe 5" definiert die Farbe, mit welcher der Balken bezüglich der "Grenze 5" gefüllt wird.

Das Attribut "Balkenfüllfarbe 5" ist dynamisierbar mit dem Namen "Layer05FillColor".

Layer06FillColor-Eigenschaft

Balkenfüllfarbe 6 (Layer06FillColor)

Das Attribut "Balkenfüllfarbe 6" definiert die Farbe, mit welcher der Balken bezüglich der "Grenze 6" gefüllt wird.

Das Attribut "Balkenfüllfarbe 6" ist dynamisierbar mit dem Namen "Layer06FillColor".

Layer07FillColor-Eigenschaft

Balkenfüllfarbe 7 (Layer07FillColor)

Das Attribut "Balkenfüllfarbe 7" definiert die Farbe, mit welcher der Balken bezüglich der "Grenze 7" gefüllt wird.

Das Attribut "Balkenfüllfarbe 7" ist dynamisierbar mit dem Namen "Layer07FillColor".

Layer08FillColor-Eigenschaft

Balkenfüllfarbe 8 (Layer08FillColor)

Das Attribut "Balkenfüllfarbe 8" definiert die Farbe, mit welcher der Balken bezüglich der "Grenze 8" gefüllt wird.

Das Attribut "Balkenfüllfarbe 8" ist dynamisierbar mit dem Namen "Layer08FillColor".

Layer09FillColor-Eigenschaft

Balkenfüllfarbe 9 (Layer09FillColor)

Das Attribut "Balkenfüllfarbe 9" definiert die Farbe, mit welcher der Balken bezüglich der "Grenze 9" gefüllt wird.

Das Attribut "Balkenfüllfarbe 9" ist dynamisierbar mit dem Namen "Layer09FillColor".

Layer10FillColor-Eigenschaft

Balkenfüllfarbe 10 (Layer10FillColor)

Das Attribut "Balkenfüllfarbe 10" definiert die Farbe, mit welcher der Balken bezüglich der "Grenze 10" gefüllt wird.

Das Attribut "Balkenfüllfarbe 10" ist dynamisierbar mit dem Namen "Layer10FillColor".

Layer00FillStyle-Eigenschaft

Balkenfüllmuster 0 (Layer00FillStyle)

Das Attribut "Balkenfüllmuster 0" legt das Muster des Balkens bezüglich der "Grenze 0" fest. Damit das Füllmuster sichtbar wird, muss die "Balkenfüllfarbe 0" von der "Balkenfarbe 0" abweichen.

Es stehen 50 Füllmuster zur Auswahl. Das Füllmuster 0 "Massiv" füllt das Objekt mit der eingestellten Hintergrundfarbe aus. Das Füllmuster 1 "Transparent" bewirkt, dass weder ein Hintergrund noch ein Füllmuster angezeigt werden.

Das Attribut "Balkenfüllmuster 0" ist dynamisierbar mit dem Namen "Layer00FillStyle".

Layer01FillStyle-Eigenschaft

Balkenfüllmuster 1 (Layer01FillStyle)

Das Attribut "Balkenfüllmuster 1" legt das Muster des Balkens bezüglich der "Grenze 1" fest. Damit das Füllmuster sichtbar wird, muss die "Balkenfüllfarbe 1" von der "Balkenfarbe 1" abweichen.

Es stehen 50 Füllmuster zur Auswahl. Das Füllmuster 0 "Massiv" füllt das Objekt mit der eingestellten Hintergrundfarbe aus. Das Füllmuster 1 "Transparent" bewirkt, dass weder ein Hintergrund noch ein Füllmuster angezeigt werden.

Das Attribut "Balkenfüllmuster 1" ist dynamisierbar mit dem Namen "Layer01FillStyle".

Layer02FillStyle-Eigenschaft

Balkenfüllmuster 2 (Layer02FillStyle)

Das Attribut "Balkenfüllmuster 2" legt das Muster des Balkens bezüglich der "Grenze 2" fest. Damit das Füllmuster sichtbar wird, muss die "Balkenfüllfarbe 2" von der "Balkenfarbe 2" abweichen.

Es stehen 50 Füllmuster zur Auswahl. Das Füllmuster 0 "Massiv" füllt das Objekt mit der eingestellten Hintergrundfarbe aus. Das Füllmuster 1 "Transparent" bewirkt, dass weder ein Hintergrund noch ein Füllmuster angezeigt werden.

Das Attribut "Balkenfüllmuster 2" ist dynamisierbar mit dem Namen "Layer02FillStyle".

Layer03FillStyle-Eigenschaft

Balkenfüllmuster 3 (Layer03FillStyle)

Das Attribut "Balkenfüllmuster 3" legt das Muster des Balkens bezüglich der "Grenze 3" fest. Damit das Füllmuster sichtbar wird, muss die "Balkenfüllfarbe 3" von der "Balkenfarbe 3" abweichen.

Es stehen 50 Füllmuster zur Auswahl. Das Füllmuster 0 "Massiv" füllt das Objekt mit der eingestellten Hintergrundfarbe aus. Das Füllmuster 1 "Transparent" bewirkt, dass weder ein Hintergrund noch ein Füllmuster angezeigt werden.

Das Attribut "Balkenfüllmuster 3" ist dynamisierbar mit dem Namen "Layer03FillStyle".

Layer04FillStyle-Eigenschaft

Balkenfüllmuster 4 (Layer04FillStyle)

Das Attribut "Balkenfüllmuster 4" legt das Muster des Balkens bezüglich der "Grenze 4" fest. Damit das Füllmuster sichtbar wird, muss die "Balkenfüllfarbe 4" von der "Balkenfarbe 4" abweichen.

Es stehen 50 Füllmuster zur Auswahl. Das Füllmuster 0 "Massiv" füllt das Objekt mit der eingestellten Hintergrundfarbe aus. Das Füllmuster 1 "Transparent" bewirkt, dass weder ein Hintergrund noch ein Füllmuster angezeigt werden.

Das Attribut "Balkenfüllmuster 4" ist dynamisierbar mit dem Namen "Layer04FillStyle".

Layer05FillStyle-Eigenschaft

Balkenfüllmuster 5 (Layer05FillStyle)

Das Attribut "Balkenfüllmuster 5" legt das Muster des Balkens bezüglich der "Grenze 5" fest. Damit das Füllmuster sichtbar wird, muss die "Balkenfüllfarbe 5" von der "Balkenfarbe 5" abweichen.

Es stehen 50 Füllmuster zur Auswahl. Das Füllmuster 0 "Massiv" füllt das Objekt mit der eingestellten Hintergrundfarbe aus. Das Füllmuster 1 "Transparent" bewirkt, dass weder ein Hintergrund noch ein Füllmuster angezeigt werden.

Das Attribut "Balkenfüllmuster 5" ist dynamisierbar mit dem Namen "Layer05FillStyle".

Layer06FillStyle-Eigenschaft

Balkenfüllmuster 6 (Layer06FillStyle)

Das Attribut "Balkenfüllmuster 6" legt das Muster des Balkens bezüglich der "Grenze 6" fest. Damit das Füllmuster sichtbar wird, muss die "Balkenfüllfarbe 6" von der "Balkenfarbe 6" abweichen.

Es stehen 50 Füllmuster zur Auswahl. Das Füllmuster 0 "Massiv" füllt das Objekt mit der eingestellten Hintergrundfarbe aus. Das Füllmuster 1 "Transparent" bewirkt, dass weder ein Hintergrund noch ein Füllmuster angezeigt werden.

Das Attribut "Balkenfüllmuster 6" ist dynamisierbar mit dem Namen "Layer06FillStyle".

Layer07FillStyle-Eigenschaft

Balkenfüllmuster 7 (Layer07FillStyle)

Das Attribut "Balkenfüllmuster 7" legt das Muster des Balkens bezüglich der "Grenze 7" fest. Damit das Füllmuster sichtbar wird, muss die "Balkenfüllfarbe 7" von der "Balkenfarbe 7" abweichen.

Es stehen 50 Füllmuster zur Auswahl. Das Füllmuster 0 "Massiv" füllt das Objekt mit der eingestellten Hintergrundfarbe aus. Das Füllmuster 1 "Transparent" bewirkt, dass weder ein Hintergrund noch ein Füllmuster angezeigt werden.

Das Attribut "Balkenfüllmuster 7" ist dynamisierbar mit dem Namen "Layer07FillStyle".

Layer08FillStyle-Eigenschaft

Balkenfüllmuster 8 (Layer08FillStyle)

Das Attribut "Balkenfüllmuster 8" legt das Muster des Balkens bezüglich der "Grenze 8" fest. Damit das Füllmuster sichtbar wird, muss die "Balkenfüllfarbe 8" von der "Balkenfarbe 8" abweichen.

Es stehen 50 Füllmuster zur Auswahl. Das Füllmuster 0 "Massiv" füllt das Objekt mit der eingestellten Hintergrundfarbe aus. Das Füllmuster 1 "Transparent" bewirkt, dass weder ein Hintergrund noch ein Füllmuster angezeigt werden.

Das Attribut "Balkenfüllmuster 8" ist dynamisierbar mit dem Namen "Layer08FillStyle".

Layer09FillStyle-Eigenschaft

Balkenfüllmuster 9 (Layer09FillStyle)

Das Attribut "Balkenfüllmuster 9" legt das Muster des Balkens bezüglich der "Grenze 9" fest. Damit das Füllmuster sichtbar wird, muss die "Balkenfüllfarbe 9" von der "Balkenfarbe 9" abweichen.

Es stehen 50 Füllmuster zur Auswahl. Das Füllmuster 0 "Massiv" füllt das Objekt mit der eingestellten Hintergrundfarbe aus. Das Füllmuster 1 "Transparent" bewirkt, dass weder ein Hintergrund noch ein Füllmuster angezeigt werden.

Das Attribut "Balkenfüllmuster 9" ist dynamisierbar mit dem Namen "Layer09FillStyle".

Layer10FillStyle-Eigenschaft

Balkenfüllmuster 10 (Layer10FillStyle)

Das Attribut "Balkenfüllmuster 10" legt das Muster des Balkens bezüglich der "Grenze 10" fest. Damit das Füllmuster sichtbar wird, muss die "Balkenfüllfarbe 10" von der "Balkenfarbe 10" abweichen.

Es stehen 50 Füllmuster zur Auswahl. Das Füllmuster 0 "Massiv" füllt das Objekt mit der eingestellten Hintergrundfarbe aus. Das Füllmuster 1 "Transparent" bewirkt, dass weder ein Hintergrund noch ein Füllmuster angezeigt werden.

Das Attribut "Balkenfüllmuster 10" ist dynamisierbar mit dem Namen "Layer10FillStyle".

Layer00Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 0" fest oder gibt ihn zurück.
Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer00Checked den Wert TRUE hat.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer01Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 1" fest oder gibt ihn zurück.
Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer01Checked den Wert TRUE hat.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer02Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 2" fest oder gibt ihn zurück.
Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer02Checked den Wert TRUE hat.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer03Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 3" fest oder gibt ihn zurück.
Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer03Checked den Wert TRUE hat.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer04Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 4" fest oder gibt ihn zurück.
Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer04Checked den Wert TRUE hat.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer05Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 5" fest oder gibt ihn zurück.
Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer05Checked den Wert TRUE hat.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer06Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 6" fest oder gibt ihn zurück.
Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer06Checked den Wert TRUE hat.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer07Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 7" fest oder gibt ihn zurück.
Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer07Checked den Wert TRUE hat.

Siehe auch

ScreenItem-Objekt (Seite 134)

3D-Balken (Seite 181)

Layer08Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 8" fest oder gibt ihn zurück.
Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer08Checked den Wert TRUE hat.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer09Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 9" fest oder gibt ihn zurück.
Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer09Checked den Wert TRUE hat.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Layer10Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 10" fest oder gibt ihn zurück.
Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer10Checked den Wert TRUE hat.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

LayerDeclutteringEnable-Eigenschaft

Beschreibung

Liefert die LayerDecluttering-Eigenschaft eines Bildes zurück.

LayerDecluttering ermöglicht das Ein- und Ausblenden von Ebenen abhängig vom eingestellten Minimal- und Maximalzoom.

BOOLEAN Nur-Lese-Zugriff.

Beispiel:

Das Beispiel gibt die LayerDecluttering-Eigenschaft des Bildes NewPDL1 als Trace aus.

```
'VBS156
Dim objScreen
Set objScreen = HMIRuntime.Screens("NewPDL1")
HMIRuntime.Trace "Enable: " & objScreen.LayerDeclutteringEnable & vbNewLine
```

Siehe auch

Screen-Objekt (Seite 140)

Layers-Eigenschaft

Beschreibung

Liefert ein Objekt vom Typ "Layers" zurück.

Layers (read-only)

Siehe auch

Layers-Objekt (Auflistung) (Seite 130)

HMIRuntime-Objekt (Seite 127)

Le - Li

Left-Eigenschaft

Beschreibung

Legt die x-Koordinate eines Objektes (gemessen vom linken, oberen Bildrand) in Pixel fest oder gibt sie zurück. Die x-Koordinate bezieht sich auf die Ecke links oben des objektumfassenden Rechteckes.

LONG (Schreib-Lese-Zugriff)

Beispiel

Das folgende Beispiel verschiebt alle Objekte des Bildes "NewPDL1" um 5 Pixel nach links:

```
'VBS79
Dim objScreen
Dim objScrItem
Dim lngIndex
Dim strName
lngIndex = 1
Set objScreen = HMIRuntime.Screens("NewPDL1")
For lngIndex = 1 To objScreen.ScreenItems.Count
strName = objScreen.ScreenItems.Item(lngIndex).ObjectName
Set objScrItem = objScreen.ScreenItems(strName)
objScrItem.Left = objScrItem.Left - 5
Next
```

Siehe auch

Top-Eigenschaft (Seite 618)

ScreenItem-Objekt (Seite 134)

LeftComma-Eigenschaft

Beschreibung

Legt die Anzahl der Vorkommastellen (0 bis 20) fest oder gibt sie zurück.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

LightEffect-Eigenschaft

Beschreibung

TRUE, wenn der Lichteffekt eingeschaltet ist. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

LimitHigh4-Eigenschaft

Beschreibung

Legt den oberen Grenzwert für "Reserve 4" fest oder gibt ihn zurück.
Die Eigenschaft CheckLimitHigh4 muss auf TRUE gesetzt sein, damit der Grenzwert "Reserve 4" überwacht werden kann.
Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft TypeLimitHigh4 fest.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

LimitHigh5-Eigenschaft

Beschreibung

Legt den oberen Grenzwert für "Reserve 5" fest oder gibt ihn zurück.
Die Eigenschaft CheckLimitHigh5 muss auf TRUE gesetzt sein, damit der Grenzwert "Reserve 5" überwacht werden kann.
Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft TypeLimitHigh5 fest.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

LimitLow4-Eigenschaft

Beschreibung

Legt den unteren Grenzwert für "Reserve 4" fest oder gibt ihn zurück.
Die Eigenschaft CheckLimitLow4 muss auf TRUE gesetzt sein, damit der Grenzwert "Reserve 4" überwacht werden kann.
Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft TypeLimitLow4 fest.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

LimitLow5-Eigenschaft

Beschreibung

Legt den unteren Grenzwert für "Reserve 5" fest oder gibt ihn zurück.
Die Eigenschaft CheckLimitLow5 muss auf TRUE gesetzt sein, damit der Grenzwert "Reserve 5" überwacht werden kann.
Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft TypeLimitLow5 fest.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

LimitMax-Eigenschaft

Beschreibung

Legt den oberen Grenzwert als Absolutwert abhängig vom Datenformat fest oder gibt ihn zurück.
Überschreitet der anzuzeigende Wert den oberen Grenzwert, wird er durch eine Folge von *** als nicht darstellbar gekennzeichnet.

Siehe auch

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

LimitMin-Eigenschaft

Beschreibung

Legt den unteren Grenzwert als Absolutwert abhängig vom Datenformat fest oder gibt ihn zurück.
Überschreitet der anzuzeigende Wert den oberen Grenzwert, wird er durch eine Folge von *** als nicht darstellbar gekennzeichnet.

Siehe auch

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

LineColor-.Eigenschaft

Farbe Fenster-Trennlinien - LineColor

Gibt die Farbe für die Fenstertrennlinien an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **LineColor** dynamisierbar. Der Datentyp ist LONG.

LineFont-Eigenschaft

Beschreibung

TRUE, wenn die Schriftgröße automatisch der Zeilenhöhe angepasst wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

LineHeight-Eigenschaft

Beschreibung

TRUE, wenn die Zeilenhöhe verändert werden kann. BOOLEAN Schreib-Lese-Zugriff.

Die Eigenschaft "LineHeight" ist nur dann deaktiviert, wenn bei den beiden Eigenschaften "LineHeight" und "LineFont" "FALSE" ist.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

LineJoinStyle-Eigenschaft

Beschreibung

Legt fest, auf welche Weise die Ecken bei einem Rohrpolygon dargestellt werden.

Eckig	Die Rohre sind an den Eckpunkten ohne Abrundungen miteinander verbunden
Rund	Die Rohre sind an den Eckpunkten außen abgerundet.

LineTitle-Eigenschaft

Beschreibung

TRUE, wenn das Meldefenster eine Spalte mit einer fortlaufenden Nummerierung der anstehenden Meldungen enthält. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

LineWidth-Eigenschaft (vor WinCC V7)

Beschreibung

Legt die Strichstärke der über "Index" referenzierten Kurve fest. Wertebereich von 0 bis 10.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

LineWidth-Eigenschaft

Stärke Fenster-Trennlinien - LineWidth

Legt die Stärke der Fenstertrennlinien in Pixel fest.

Das Attribut ist mit dem Namen **LineWidth** dynamisierbar. Der Datentyp ist LONG.

ListType-Eigenschaft

Beschreibung

Gibt beim Objekt Textliste den dargestellten Datentyp zurück. Nur Lese-Zugriff.

Wertebereich von 0 bis 2.

0 = Dezimal

1 = Binär

2 = Bit

Siehe auch

Textliste (Seite 208)

ScreenItem-Objekt (Seite 134)

Lo

LoadDataImmediately-Eigenschaft

Daten aus Archiv laden - LoadDataImmediately

Legt fest, ob bei einem Bildaufschlag die Variablenwerte für den darzustellenden Zeitbereich aus den Archiven geladen werden.

Wert	Erklärung
TRUE	Archivierte Werte werden bei einem Bildaufschlag geladen.
FALSE	Nur aktuell anfallende Werte werden bei einem Bildaufschlag geladen.

Das Attribut ist mit dem Namen **LoadDataImmediately** dynamisierbar. Der Datentyp ist BOOLEAN.

LoadDataImmediately-Eigenschaft (vor WinCC V7)

Beschreibung

TRUE, wenn bei einem Bildaufschlag die Variablenwerte für den darzustellenden Zeitbereich aus den Archiven geladen werden. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

LocaleID-Eigenschaft

Beschreibung

Stellt die Sprache ein, in der das Control angezeigt wird, zum Beispiel 1031 für Deutsch. Schreib-Lese-Zugriff.

Die Liste mit den Sprachkennungen finden Sie in der WinCC-Dokumentation (Index > Language Code).

Siehe auch

WinCC Slider Control (Seite 278)

WinCC Gauge Control (Seite 261)

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

LocaleSpecificSettings-Eigenschaft

Beschreibung

TRUE, wenn für jede Runtimesprache eine Schriftart zugewiesen und formatiert werden kann. BOOLEAN Schreib-Lese-Zugriff.

LockBackColor-Eigenschaft

Beschreibung

Legt die Hintergrundfarbe der Schaltfläche für eine gesperrte Mess-Stelle fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.
Die Eigenschaft LockStatus muss den Wert TRUE haben, damit die Hintergrundfarbe angezeigt werden kann.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

LockStatus-Eigenschaft

Beschreibung

TRUE, wenn eine gesperrte Mess-Stelle angezeigt werden soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

LockText-Eigenschaft

Beschreibung

Legt die Beschriftung der Schaltflächen für eine gesperrte Mess-Stelle fest.
Die Eigenschaft LockStatus muss den Wert TRUE haben, damit die Beschriftung angezeigt werden kann.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

LockTextColor-Eigenschaft

Beschreibung

Legt die Farbe der Schaltflächenbeschriftung für eine gesperrte Mess-Stelle fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.
Die Eigenschaft LockStatus muss den Wert TRUE haben, damit die Hintergrundfarbe angezeigt werden kann.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

Logging-Eigenschaft

Beschreibung

Liefert ein Objekt vom Typ "Logging" zurück.

Logging (read-only)

Siehe auch

HMIRuntime-Objekt (Seite 127)

Logging-Objekt (Seite 131)

LongStrokesBold-Eigenschaft

Beschreibung

TRUE, wenn bei der Darstellung der Skala die langen Abschnitte fett angezeigt werden sollen. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

LongStrokesOnly-Eigenschaft

Beschreibung

TRUE, wenn bei der Darstellung der Skala nur die langen Abschnitte angezeigt werden sollen. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

LongStrokesSize-Eigenschaft

Beschreibung

Legt die Länge der Achsabschnitte in Pixel fest oder gibt sie zurück.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

LongStrokesTextEach-Eigenschaft

Beschreibung

Gibt den Wert zurück, der festlegt, welche Abschnitte bei der Darstellung der Skala beschriftet werden sollen (1 = jeder Abschnitt, 2 = jeder zweite Abschnitt, usw.). Nur Lese-Zugriff

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

LongTimeArchiveConsistency-Eigenschaft

LongTimeArchiveConsistency

Wenn "LongTimeArchiveConsistency" auf "nein" gesetzt ist, werden 1000 Meldungen in der Langzeitarchivliste auf dem Einzelplatz, auf dem Server oder auf dem Client pro Server bzw. pro redundantem Serverpaar angezeigt.

Wenn Sie "LongTimeArchiveConsistency" auf "ja" setzen, werden die 1000 jüngsten Meldungen auf dem Client von allen Servern oder redundantem Serverpaar in der Langzeitarchivliste angezeigt.

Das Attribut ist mit dem Namen **LongTimeArchiveConsistency** dynamisierbar. Der Datentyp ist BOOLEAN.

LongTimeArchiveConsistency-Eigenschaft (vor WinCC V7)

Beschreibung

Wenn "LongTimeArchiveConsistency" auf "nein" gesetzt ist, werden 1000 Meldungen in der Langzeitarchivliste auf dem Einzelplatz, auf dem Server oder auf dem Client pro Server bzw. pro redundantem Serverpaar angezeigt.

Wenn Sie "LongTimeArchiveConsistency" auf "ja" setzen, werden die 1000 jüngsten Meldungen auf dem Client von allen Servern bzw. redundantem Serverpaar in der Langzeitarchivliste angezeigt.

Schreib-Lese-Zugriff.

LowerLimit-Eigenschaft

Beschreibung

WinCC Online Trend Control/WinCC Function Trend Control

TRUE, wenn die Angabe von "LowerLimitColor" verwendet wird um die Variablenwerte (einer über "Index" referenzierten Kurve) zu kennzeichnen, die unterhalb des Wertes von "LowerLimitValue" liegen. BOOLEAN Schreib-Lese-Zugriff.

WinCC Online Table Control

Der Wert dieses Attributs kann nicht verändert werden. Lese-Zugriff.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

LowerLimitColor-Eigenschaft

Beschreibung

WinCC Online Trend Control/WinCC Function Trend Control

Legt die Farbe fest, die verwendet wird, um die Variablenwerte (einer über "Index" referenzierten Kurve) zu kennzeichnen, die unterhalb des Wertes von "LowerLimitValue" liegen. Ob die Angabe ausgewertet wird ist abhängig von der Eigenschaft "LowerLimit". Die Angabe der Farbe erfolgt als RGB-Wert. LONG Schreib-Lese-Zugriff.

Online Table Control

Der Wert dieses Attributs kann nicht verändert werden. Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

LowerLimitTagName-Eigenschaft

Beschreibung

Legt die Untergrenze des Kurvenbereichs fest, die automatisch aus den in PCS 7 projektierten Eigenschaften von Variablen übernommen wird. Schreib-Lese-Zugriff.

LowerLimitValue-Eigenschaft

Beschreibung

WinCC Online Trend Control/WinCC Function Trend Control

Variablenwerte (einer über "Index" referenzierten Kurve), die den Wert von "LowerLimitValue" unterschreiten, werden mit der in "LowerLimitColor" festgelegten Farbe gekennzeichnet. Ob die Angabe ausgewertet wird ist abhängig vom Attribut "LowerLimit".

Online Table Control

Der Wert dieses Attributs kann nicht verändert werden. Lese-Zugriff.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

1.14.4.12 M

Ma - Mc

Marker-Eigenschaft

Beschreibung

TRUE, wenn die Grenzwerte als Skalenwert angezeigt werden sollen. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

Max-Eigenschaft

Beschreibung

Legt den Absolutwert bei voller Wertanzeige fest oder gibt ihn zurück. Ist die Skalenanzeige aktiv, so wird dieser Wert angezeigt.

Siehe auch

Balken (Seite 186)

Slider (Seite 221)

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

MaximizeButton-Eigenschaft

Beschreibung

TRUE, wenn das Objekt in Runtime maximiert werden kann. Nur Lese-Zugriff.

Siehe auch

Bildfenster (Seite 190)

Applikationsfenster (Seite 185)

ScreenItem-Objekt (Seite 134)

MCGUBackColorOff-Eigenschaft

Beschreibung

Legt für den Zustand "Gegangen Unquittiert" die Farbe des Hintergrunds für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCGUBackColorOn-Eigenschaft

Beschreibung

Legt für den Zustand "Gegangen Unquittiert" die Farbe des Hintergrunds für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCGUBackFlash-Eigenschaft

Beschreibung

TRUE, wenn der Hintergrund beim unquittierten Gehen einer Meldung blinken soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCGUTextColorOff-Eigenschaft

Beschreibung

Legt für den Zustand "Gegangen Unquittiert" die Farbe des Textes für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCGUTextColorOn-Eigenschaft

Beschreibung

Legt für den Zustand "Gegangen Unquittiert" die Hintergrundfarbe des Textes für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCGUTextFlash-Eigenschaft

Beschreibung

TRUE, wenn die Schrift beim unquittierten Gehen einer Meldung blinken soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCKOBackColorOff-Eigenschaft

Beschreibung

Legt für den Zustand "Gekommen" die Farbe des Hintergrunds für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCKOBackColorOn-Eigenschaft

Beschreibung

Legt für den Zustand "Gekommen" die Farbe des Hintergrunds für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCKOBackFlash-Eigenschaft

Beschreibung

TRUE, wenn der Hintergrund beim Kommen einer Meldung blinken soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCKOTextColorOff-Eigenschaft

Beschreibung

Legt für den Zustand "Gekommen" die Farbe des Textes für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCKOTextColorOn-Eigenschaft

Beschreibung

Legt für den Zustand "Gekommen" die Farbe des Hintergrunds des Textes für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCKOTextFlash-Eigenschaft

Beschreibung

TRUE, wenn die Schrift beim Kommen einer Meldung blinken soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCKQBackColorOff-Eigenschaft

Beschreibung

Legt für den Zustand "Gegangen Quittiert" die Farbe des Hintergrunds für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCKQBackColorOn-Eigenschaft

Beschreibung

Legt für den Zustand "Gegangen Quittiert" die Farbe des Hintergrunds für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCKQBackFlash-Eigenschaft

Beschreibung

TRUE, wenn der Hintergrund beim quittierten Gehen einer Meldung blinken soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCKQTextColorOff-Eigenschaft

Beschreibung

Legt für den Zustand "Gegangen Quittiert" die Farbe des Textes für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCKQTextColorOn-Eigenschaft

Beschreibung

Legt für den Zustand "Gegangen Quittiert" die Farbe des Hintergrunds des Textes für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCKQTextFlash-Eigenschaft

Beschreibung

TRUE, wenn die Schrift beim quittierten Gehen einer Meldung blinken soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MCText-Eigenschaft

Beschreibung

Legt die Beschriftung für die jeweilige Meldeklasse fest oder gibt sie zurück.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

Me

MeasurePoints-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. "MeasurePoints"legt die Anzahl der darzustellenden Messpunkte fest. Die Angabe wird nur ausgewertet, wenn die Eigenschaft "TimeAxis" den Wert "-1" besitzt.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

MenuToolBarConfig-Eigenschaft

Beschreibung

Lädt die angegebene Konfigurationsdatei mit benutzerdefiniertem Menü und Symbolleisten oder gibt den Namen der Konfigurationsdatei zurück. STRING-Schreib-Lese-Zugriff.

Siehe auch

Bildfenster (Seite 190)

HMIRuntime-Objekt (Seite 127)

MessageBlockAlign-Eigenschaft

Ausrichtung Meldeblöcke - MessageBlockAlign

Legt fest, wie die Inhalte des ausgewählten Meldeblocks in der Tabelle ausgerichtet werden.

Um die Ausrichtung ändern zu können, muss die Option "Projekteinstellungen übernehmen" deaktiviert bzw. "ApplyProjectSettings" auf "FALSE" gesetzt sein.

Folgende Einstellungen sind möglich:

Wert	Beschreibung	Erklärung
0	links	Die Inhalte des ausgewählten Meldeblocks werden linksbündig dargestellt.
1	zentriert	Die Inhalte des ausgewählten Meldeblocks werden zentriert dargestellt.
2	rechts	Die Inhalte des ausgewählten Meldeblocks werden rechtsbündig dargestellt.

Das Attribut ist mit dem Namen **MessageBlockAlign** dynamisierbar. Der Datentyp ist LONG.

MessageBlockAutoPrecisions-Egienschaft

Nachkommastellen Automatisch - MessageBlockAutoPrecisions

Legt fest, ob die Anzahl der Nachkommastellen automatisch bestimmt wird.

Wert	Erklärung
TRUE	Die Anzahl der Nachkommastellen wird automatisch festgelegt. Der Wert im Feld "Nachkommastellen" ist nicht wirksam.
FALSE	Der Wert im Feld "Nachkommastellen" ist wirksam.

Das Attribut ist mit dem Namen **MessageBlockAutoPrecisions** dynamisierbar. Der Datentyp ist BOOLEAN.

MessageBlockCaption-Eigenschaft

Bezeichnung - MessageBlockCaption

Legt für den ausgewählten Meldeblock die Bezeichnung der Spaltenüberschrift im Meldefenster fest. Die angegebene Bezeichnung ist in allen Runtime-Sprachen wirksam.

Um die Bezeichnung ändern zu können, muss die Option "Projekteinstellungen übernehmen" deaktiviert bzw. "ApplyProjectSettings" auf "FALSE" gesetzt sein.

Das Attribut ist mit dem Namen **MessageBlockCaption** dynamisierbar. Der Datentyp ist STRING.

MessageBlockCount-Eigenschaft

MessageBlockCount

Gibt die Anzahl der vorhandenen Meldblöcke an, die für die Meldeliste und Hitliste zur Verfügung stehen.

Das Attribut ist mit dem Namen **MessageBlockCount** dynamisierbar. Der Datentyp ist LONG.

MessageBlockDateFormat-Eigenschaft

Datumsformat - MessageBlockDateFormat

Legt fest, welches Datumsformat zur Anzeige der Meldungen verwendet wird.

Um das Datumsformat ändern zu können, muss die Option "Projekteinstellungen übernehmen" deaktiviert bzw. "ApplyProjectSettings" auf "FALSE" gesetzt sein.

Folgende Datumsformate stehen zur Verfügung:

Wert	Erklärung
Automatisch	Das Datumsformat wird automatisch bestimmt.
dd.MM.yy	Tag.Monat.Jahr, z.B. 24.12.07.
dd.MM.yyyy	Tag.Monat.Jahr, z.B. 24.12.2007.
dd/MM/yy	Tag/Monat/Jahr, z.B. 24/12/07.
dd/MM/yyyy	Tag/Monat/Jahr, z.B. 24/12/2007.

Das Attribut ist mit dem Namen **MessageBlockDateFormat** dynamisierbar. Der Datentyp ist STRING.

MessageBlockExponentialFormat-Eigenschaft

Exponentialdarstellung - MessageBlockExponentialFormat

Legt fest, ob die Werte des ausgewählten Meldeblocks in Exponentialdarstellung angezeigt werden.

Wert	Erklärung
TRUE	Die Werte werden in Exponentialdarstellung angezeigt.
FALSE	Die Werte werden in Dezimaldarstellung angezeigt.

Das Attribut ist mit dem Namen **MessageBlockExponentialFormat** dynamisierbar. Der Datentyp ist BOOLEAN.

MessageBlockFlashMode-Eigenschaft

Blinkmodus - MessageBlockFlashMode

Legt fest, wie der Inhalt des ausgewählten Meldeblocks beim Erscheinen einer Meldung in Runtime blinkt. Die Option "Blinken ein" muss aktiviert sein.

Um die Einstellung ändern zu können, muss die Option "Projekteinstellungen übernehmen" deaktiviert bzw. "ApplyProjectSettings" auf "FALSE" gesetzt sein.

Wert	Beschreibung	Erklärung
0	Standard	Beim Blinken wechselt die Textfarbe zwischen der Standardfarbe und der Blinkfarbe
1	Wechsel Hintergrund-/ Textfarbe	Beim Blinken wechseln sich Hintergrundfarbe und Textfarbe der Meldung ab. Die Meldungsfarben konfigurieren Sie im Alarmlogging-Editor bei der Meldeart.
2	Wechsel Meldungs-/ Tabellenfarbe	Beim Blinken wechseln sich Meldungsfarben und konfigurierte Tabellenfarben ab. Die Meldungsfarben konfigurieren Sie im Alarmlogging-Editor bei der Meldeart. Die Tabellenfarben stellen Sie im AlarmControl auf der Registerkarte "Darstellung" ein.

Das Attribut ist mit dem Namen **MessageBlockFlashMode** dynamisierbar. Der Datentyp ist LONG.

MessageBlockFlashOn-Eigenschaft

Blinken ein - MessageBlockFlashOn

Legt fest, ob der Inhalt des gewählten Meldeblocks beim Erscheinen einer Meldung in Runtime blinkt.

Um die Einstellung ändern zu können, muss die Option "Projekteinstellungen übernehmen" deaktiviert bzw. "ApplyProjectSettings" auf "FALSE" gesetzt sein.

Wert	Erklärung
TRUE	Der Inhalt des Meldeblocks blinkt.
FALSE	Der Inhalt des Meldeblocks blinkt nicht.

Das Attribut ist mit dem Namen **MessageBlockFlashOn** dynamisierbar. Der Datentyp ist BOOLEAN.

MessageBlockHideText-Egienschaft

Inhalt als Text - MessageBlockHideText

Legt fest, ob der Inhalt des ausgewählten Meldeblocks als Text angezeigt wird.

Wert	Erklärung
TRUE	Der Inhalt wird nicht als Text angezeigt. Die Option ist nicht aktiviert
FALSE	Der Inhalt wird als Text angezeigt. Die Option ist aktiviert

Das Attribut ist mit dem Namen **MessageBlockHideText** dynamisierbar. Der Datentyp ist BOOLEAN.

MessageBlockHideTitleText-Eigenschaft

Überschrift als Text - MessageBlockHideTitleText

Legt fest, ob die Überschrift des ausgewählten Meldeblocks als Text angezeigt wird.

Wert	Erklärung
TRUE	Die Überschrift wird nicht als Text angezeigt. Die Option ist nicht aktiviert
FALSE	Die Überschrift wird als Text angezeigt. Die Option ist aktiviert

Das Attribut ist mit dem Namen **MessageBlockHideTitleText** dynamisierbar. Der Datentyp ist BOOLEAN.

MessageBlockId-Eigenschaft

MessageBlockId

Festgelegte Zuordnung von Ident-Nummer und Meldeblock im WinCC AlarmControl.

Das Attribut ist mit dem Namen **MessageBlockID** dynamisierbar. Der Datentyp ist LONG.

MessageBlockInvertUseMessageColor-Eigenschaft

MessageBlockInvertUseMessageColor

Legt für den Meldeblock fest, ob entgegen der zentrale Einstellung für das AlarmControl die Meldungsfarben angezeigt bzw. nicht angezeigt werden. Z. B. haben Sie die Eigenschaft "UseMessageColor" für das AlarmControl auf "FALSE" gesetzt. Die Eigenschaft "MessageBlockInvertUseMessageColor" haben Sie für einen Meldeblock auf "TRUE" gesetzt. Dadurch werden in Runtime für diesen Meldeblock die Meldungsfarben angezeigt.

Wert	Erklärung
TRUE	Für den Meldeblock werden die Meldungsfarben entgegen der zentralen Einstellung in "UseMessageColor" angezeigt bzw. nicht angezeigt
FALSE	Für den Meldeblock werden die Meldungsfarben wie die zentrale Einstellung in "UseMessageColor" angezeigt bzw. nicht angezeigt

Das Attribut ist mit dem Namen **MessageBlockInvertUseMessageColor** dynamisierbar. Der Datentyp ist BOOLEAN.

MessageBlockIndex-Eigenschaft

MessageBlockIndex

Referenziert einen vorhandenen Meldeblock. Unter Verwendung des Attributs können Sie einem bestimmten Meldeblock für andere Attribute Werte zuweisen.

Gültige Werte für "MessageBlockIndex" liegen zwischen 0 und "MessageBlockCount" minus 1. Das Attribut "MessageBlockCount" gibt die Anzahl der vorhandenen Meldeblöcke an.

Das Attribut ist mit dem Namen **MessageBlockIndex** dynamisierbar. Der Datentyp ist LONG.

MessageBlockLeadingZeros-Eigenschaft

Anzahl der Stellen - MessageBlockLeadingZeros

Legt die Anzahl der führenden Nullen für den Inhalt des Meldeblocks fest. Die maximale Anzahl beträgt "11". Der Wert "0" bewirkt, dass die Option "Mit führenden Nullen" deaktiviert wird.

Um die Einstellung ändern zu können, muss die Option "Projekteinstellungen übernehmen" deaktiviert bzw. "ApplyProjectSettings" auf "FALSE" gesetzt sein.

Das Attribut ist mit dem Namen **MessageBlockLeadingZeros** dynamisierbar. Der Datentyp ist LONG.

MessageBlockLength-Eigenschaft

Länge in Zeichen - MessageBlockLength

Legt für den Inhalt des gewählten Meldeblocks die Länge in Zeichen fest.

Um die Länge ändern zu können, muss die Option "Projekteinstellungen übernehmen" deaktiviert bzw. "ApplyProjectSettings" auf "FALSE" gesetzt sein.

Das Attribut ist mit dem Namen **MessageBlockLength** dynamisierbar. Der Datentyp ist LONG.

MessageBlockName-Eigenschaft

Objektname - MessageBlockName

Zeigt den Namen für den ausgewählten Meldeblock an. Den Namen können Sie nicht ändern.

Der Datentyp ist STRING.

MessageBlockPrecisions-Eigenschaft

Nachkommastellen - MessageBlockPrecisions

Legt die Anzahl der Nachkommastellen der Werte des ausgewählten Meldeblocks fest. Sie können den Wert nur eingeben, wenn die Option "Automatisch" nicht aktiviert ist.

Das Attribut ist mit dem Namen **MessageBlockPrecisions** dynamisierbar. Der Datentyp ist SHORT.

MessageBlockSelected-Eigenschaft

Vorhandene Meldeblöcke - MessageBlockSelected

Vorhandene Meldeblöcke sind Blöcke, die in Runtime zur Verwendung für die Meldeliste oder Hitliste zur Verfügung stehen.

Auf der Registerkarte "Meldeblöcke" aktivieren Sie die Blöcke der vorhandenen Meldeblöcke, die Sie im Control benötigen. Auf den Registerkarten "Hitliste" und "Meldeliste" projektieren Sie die Hitliste und Meldeliste aus den vorhandenen Blöcken.

Um die Einstellung ändern zu können, muss die Option "Projekteinstellungen übernehmen" deaktiviert bzw. "ApplyProjectSettings" auf "FALSE" gesetzt sein.

Das Attribut ist mit dem Namen **MessageBlockSelected** dynamisierbar. Der Datentyp ist BOOLEAN.

MessageBlockShowDate-Eigenschaft

Datum anzeigen - MessageBlockShowDate

Legt fest, ob im Meldeblock "Uhrzeit" neben der Uhrzeit auch das Datum angezeigt wird.

Wert	Erklärung
TRUE	Das Datum und die Uhrzeit werden angezeigt.
FALSE	Die Uhrzeit wird angezeigt.

Das Attribut ist mit dem Namen **MessageBlockShowDate** dynamisierbar. Der Datentyp ist BOOLEAN.

MessageBlockShowIcon-Eigenschaft

Inhalt als Symbol - MessageBlockShowIcon

Legt fest, ob der Inhalt des ausgewählten Meldeblocks als Symbol angezeigt wird.

Wert	Erklärung
TRUE	Der Inhalt wird als Symbol angezeigt.
FALSE	Der Inhalt wird nicht als Symbol angezeigt.

Das Attribut ist mit dem Namen **MessageBlockShowIcon** dynamisierbar. Der Datentyp ist BOOLEAN.

MessageBlockShowTitleIcon-Eigenschaft

Überschrift als Symbol - MessageBlockShowTitleIcon

Legt fest, ob die Überschrift des ausgewählten Meldeblocks als Symbol angezeigt wird.

Wert	Erklärung
TRUE	Die Überschrift wird als Symbol angezeigt.
FALSE	Die Überschrift wird nicht als Symbol angezeigt.

Das Attribut ist mit dem Namen **MessageBlockShowTitleIcon** dynamisierbar. Der Datentyp ist BOOLEAN.

MessageBlockTextId-Eigenschaft

Text-ID - MessageBlockTextId

Legt die Bezeichnung für den gewählten Meldeblock mit Hilfe einer Text-Identnummer fest, die der Text-Library entnommen wurde. Beim Wechsel der Runtime-Sprache wird dann die Bezeichnung automatisch angepasst.

Um die Einstellung ändern zu können, muss die Option "Projekteinstellungen übernehmen" deaktiviert bzw. "ApplyProjectSettings" auf "FALSE" gesetzt sein.

Das Attribut ist mit dem Namen **MessageBlockTextId** dynamisierbar. Der Datentyp ist LONG.

MessageBlockTimeFormat-Eigenschaft

MessageBlockTimeFormat

Legt fest, welches Zeitformat bzw. Zeitdauerformat zur Anzeige der Meldungen verwendet wird.

Um die Einstellung ändern zu können, muss die Option "Projekteinstellungen übernehmen" deaktiviert bzw. "ApplyProjectSettings" auf "FALSE" gesetzt sein.

Folgende Zeitformate stehen zur Verfügung:

Wert	Erklärung
Automatisch	Das Zeitformat wird automatisch bestimmt.
HH:mm:ss	Stunden:Minuten:Sekunden, z.B. 15:35:44
HH:mm:ss.ms	Stunden:Minuten:Sekunden.Millisekunden, z.B. 15:35:44.240.
hh:mm:ss tt	Stunden:Minuten:Sekunden AM/PM, z.B. 03:35:44 PM.
hh:mm:ss.ms tt	Stunden:Minuten:Sekunden.Millisekunden AM/PM, z.B. 03:35:44.240 PM.

Folgende Zeitdauerformate stehen zur Verfügung:

Wert	Erklärung
Automatisch	Das Zeitdauerformat wird automatisch bestimmt.
d H:mm:ss	Tag Stunden:Minuten:Sekunden, z.B. 1 2:03:55.
H:mm:ss.	Stunden:Minuten:Sekunden, z.B. 26:03:55.
m:ss	Minuten:Sekunden, Beispiel: 1563:55.
s	Sekunden, z.B. 93835.

Das Attribut ist mit dem Namen **MessageBlockTimeFormat** dynamisierbar. Der Datentyp ist STRING.

MessageBlockType-Eigenschaft

MessageBlockType

Zeigt die Zugehörigkeit des Meldeblocks an.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Systemblock	Der Meldeblock gehört zu den Systemblöcken
1	Textblock	Der Meldeblock gehört zu den Anwendertextblöcken.
2	Prozesswertblock	Der Meldeblock gehört zu den Prozesswertblöcken.
3	Hitlistenblock	Der Meldeblock gehört zu den Meldeblöcken der Hitliste.

Das Attribut ist mit dem Namen **MessageBlockType** dynamisierbar. Der Datentyp ist LONG.

MessageClass-Eigenschaft

Beschreibung

Legt die jeweilige Meldeart (Alarm High, Alarm Low, Warnung High, Warnung Low, ...) fest, für welche die Einstellungen "Anzeigetext", "Gekommen -", "Gekommen Quittiert -" und "Gegangen Unquittiert -" projektiert werden.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

MessageColumnAdd-Eigenschaft

MessageColumnAdd

Übernimmt den markierten Meldeblock aus der Liste der vorhandenen Meldeblöcke in die Liste der gewählten Meldeblöcke.

Das Attribut ist mit dem Namen **MessageColumnAdd** dynamisierbar. Der Datentyp ist STRING.

MessageColumnCount-Eigenschaft

MessageColumnCount

Gibt die Anzahl der gewählten Meldeblöcke an, die in der Meldeliste in Runtime angezeigt werden.

Das Attribut ist mit dem Namen **MessageColumnCount** dynamisierbar. Der Datentyp ist LONG.

MessageColumnIndex-Eigenschaft

MessageColumnIndex

Referenziert einen für die Meldeliste gewählten Meldeblock. Unter Verwendung des Attributs können Sie einem bestimmten Meldeblock der Meldeliste die Werte anderer Attribute zuweisen.

Gültige Werte für "MessageColumnIndex" liegen zwischen 0 und "MessageColumnCount" minus 1. Das Attribut "MessageColumnCount" gibt die Anzahl der für die Meldeliste gewählten Meldeblöcke an.

Das Attribut "MessageColumnIndex" ist über das Attribut **MessageColumnRepos** dynamisierbar. Der Datentyp ist LONG.

MessageColumnName-Eigenschaft

MessageColumnName

Zeigt den Namen des Meldeblocks der Meldeliste an, der mit dem Attribut "MessageColumnIndex" referenziert wird. Den Namen können Sie nicht ändern.

Das Attribut ist mit dem Namen **MessageColumnName** dynamisierbar. Der Datentyp ist STRING.

MessageColumnRemove-Eigenschaft

MessageColumnRemove

Entfernt den markierten Meldeblock aus der Liste der gewählten Meldeblöcke und fügt ihn in die Liste der vorhandenen Meldeblöcke ein.

Das Attribut ist mit dem Namen **MessageColumnRemove** dynamisierbar. Der Datentyp ist STRING.

MessageColumnRepos-Eigenschaft

Auf/Ab - MessageColumnRepos/HitlistColumnRepos

Ändert die Reihenfolge der Meldeblöcke. "Auf" und "Ab" bewegen den ausgewählten Meldeblock in der Liste nach oben oder unten. Dadurch wird in Runtime der Meldeblock im Control weiter vorne oder hinten platziert.

Das Attribut für die Hitliste ist mit dem Namen **HitlistColumnRepos** dynamisierbar.

Das Attribut für die Meldeliste ist mit dem Namen **MessageColumnRepos** dynamisierbar.

Der Datentyp ist LONG.

MessageColumnSort-Eigenschaft

MessageColumnSort

Legt fest, wie der im "MessageColumnIndex" referenzierte Meldeblock sortiert wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	nein	Keine Sortierung
1	aufsteigend	Aufsteigende Sortierung vom kleinsten zum größten Wert.
2	absteigend	Absteigende Sortierung vom größten zum kleinsten Wert.

Das Attribut ist mit dem Namen **MessageColumnSort** dynamisierbar. Der Datentyp ist LONG.

MessageColumnSortIndex-Eigenschaft

MessageColumnSortIndex

Gibt die Sortierreihenfolge des im "MessageColumnIndex" referenzierten Meldeblocks an. Wenn Sie den Wert auf "0" setzen, wird das Sortierkriterium in "MessageColumnSort" entfernt.

Das Attribut ist mit dem Namen **MessageColumnSortIndex** dynamisierbar. Der Datentyp ist LONG.

MessageColumnVisible-Eigenschaft

Gewählte Meldeblöcke - MessageColumnVisible/HitlistColumnVisible

Ausgewählte Meldeblöcke der Meldeliste bzw. Hitliste, die in Runtime angezeigt werden. Legt fest, ob der im "MessageColumnIndex" bzw. "HitlistColumnIndex" referenzierte Meldeblock angezeigt wird.

Das Attribut für die Meldeliste ist mit dem Namen **MessageColumnVisible** dynamisierbar.

Das Attribut für die Hitliste ist mit dem Namen **HitlistColumnVisible** dynamisierbar.

Der Datentyp ist BOOLEAN.

MessageListType-Eigenschaft

Aktive Liste bei Bildaufschlag - MessageListType

Auswahlfeld, das die aktive Liste bei Bildaufschlag festlegt.

Wert	Beschreibung	Erklärung
0	Meldeliste	Bei einem Bildaufschlag werden die aktuell anstehenden Meldungen dargestellt.
1	Kurzzeitarchivliste	Bei einem Bildaufschlag wird eine Kurzzeitarchivliste mit den archivierten Meldungen dargestellt. Bei neu eingehenden Meldungen wird die Anzeige sofort aktualisiert.
2	Langzeitarchivliste	Bei einem Bildaufschlag wird eine Langzeitarchivliste mit den archivierten Meldungen dargestellt.
3	Sperrliste	Bei einem Bildaufschlag werden nur die aktuell gesperrten Meldungen dargestellt.
4	Hitliste	Bei einem Bildaufschlag werden die projektierten statistischen Informationen dargestellt.
5	Liste auszublendender Meldungen	Bei einem Bildaufschlag werden die auszublendenden Meldungen dargestellt.

Das Attribut ist mit dem Namen **MessageListType** dynamisierbar. Der Datentyp ist LONG.

Mi - Ms

Min-Eigenschaft

Beschreibung

Legt den Absolutwert bei kleinster Wertanzeige fest oder gibt ihn zurück. Ist die Skalenanzeige aktiv, so wird dieser Wert angezeigt.

Siehe auch

Slider (Seite 221)

Balken (Seite 186)

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

MinuteNeedleHeight-Eigenschaft

Beschreibung

Legt die Länge des Minutenzeigers der Analoguhr fest oder gibt sie zurück. Die Angabe der Länge erfolgt in Prozent, bezogen auf die halbe Länge der kürzeren Seite des rechteckigen Hintergrunds. Schreib-Lese-Zugriff.

Beispiel:
Die kürzere Seite des rechteckigen Hintergrunds sei 100 Pixel lang.
Die Minutenzeigerlänge sei mit 80 angegeben.
Daraus ergibt sich die Länge des Minutenzeigers zu (100 Pixel / 2) * 0,8 = 40 Pixel.

Siehe auch

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

MinuteNeedleWidth-Eigenschaft

Beschreibung

Legt die Breite des Minutenzeigers der Analoguhr fest oder gibt sie zurück. Die Angabe der Breite erfolgt in Prozent, bezogen auf die doppelte Länge des Minutenzeigers.

Beispiel:
Die Länge des Minutenzeigers sei 40 Pixel.
Die Minutenzeigerbreite sei mit 8 angegeben.
Daraus ergibt sich die Breite des Minutenzeigers zu 40 Pixel * 2 * 0,08 = 6 Pixel (gerundet).

Siehe auch

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

Moveable-Eigenschaft

Beschreibung

TRUE, wenn das Objekt in Runtime verschoben werden kann. Nur Lese-Zugriff.

Siehe auch

Bildfenster (Seite 190)

Applikationsfenster (Seite 185)

ScreenItem-Objekt (Seite 134)

Moveable-Eigenschaft

Verschiebbar - Moveable

Legt fest, ob das Control in Runtime verschiebbar ist.

Wert	Erklärung
TRUE	Das Control ist in Runtime verschiebbar.
FALSE	Das Control ist nicht in Runtime verschiebbar.

Das Attribut ist mit dem Namen **Moveable** dynamisierbar. Der Datentyp ist BOOLEAN.

MsgCtrlFlags-Eigenschaft

Beschreibung

Legt die Sortierreihenfolge im Alarm Control fest. Schreib-Lese-Zugriff.

- 0: Die Sortierung erfolgt aufgrund der Einträge in der Zeitspalte in aufsteigender Reihenfolge, d.h. die ältesten Meldungen stehen im Meldefenster oben.
- 1: Die Sortierung erfolgt aufgrund der Einträge in der Zeitspalte in absteigender Reihenfolge, d.h. die ältesten Meldungen stehen im Meldefenster unten, die aktuellsten oben. Bei diesem Wert wird die Eigenschaft "AutoScroll" automatisch deaktiviert, da sonst die aktuellste Meldung aus dem Anzeigebereich des Meldefensters verschwinden kann.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

MsgFilterSQL-Eigenschaft (vor WinCC V7)

Beschreibung

Legt ein SQL-Statement zur Auswahl der im Meldefenster angezeigten Meldungen fest. Schreib-Lese-Zugriff.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

MsgFilterSQL-Eigenschaft

MsgFilterSQL

Legt für die benutzerdefinierte Selektion der Meldungen ein oder mehrere SQL-Statements fest. Mehrere benutzerdefinierte Selektionen werden mit "OR" verbunden. Wenn Sie eine feste Selektion "DefaultMsgFilterSQL" projektiert haben, werden die SQL-Statements von "DefaultMsgFilterSQL" und "MsgFilterSQL" mit "AND" verbunden.

Das Attribut ist mit dem Namen **MsgFilterSQL** dynamisierbar. Der Datentyp ist STRING.

1.14.4.13 N

Name-Eigenschaft

Beschreibung Layer- und Tag-Objekt

Gibt den Objektnamen zurück. STRING (readonly)

- Bei Variablen den Namen der Variablen ohne Server- und Variablenprefix
- Bei Ebenen den Ebenennamen

Variablen

Mit der Eigenschaft "Name" einer Variablen wird die Variable über die Tags-Auflistung angesprochen. Der Name einer Variablen kann ein Serverprefix enthalten. Variablennamen in WinCC sind nach folgendem Schema aufgebaut:

<Serverprefix>::<Variablenprefix><Name der Variable>

Wird der reine Variablenname angegeben, werden Serverprefix und Variablenprefix aus dem Kontext des Bildfensters genommen.

Wird der Variablen ein Serverprefix im Variablennamen angegeben, werden Variablen und Serverprefix des Kontextes ignoriert und das beigefügte Serverprefix verwendet.

Beschreibung WinCC Function Trend Control

Die Eigenschaft "Index" referenziert eine Kurve. "Name" legt die Bezeichnung dieser Kurve fest.

Beschreibung Project-Objekt

Gibt den Namen des aktuellen Runtime-Projektes zurück. STRING (readonly)

Beispiel

Das folgende Beispiel gibt den Namen des aktuellen Runtime-Projektes als Trace aus:

```
'VBS160
HMIRuntime.Trace "Name: " & HMIRuntime.ActiveProject.Name & vbNewLine
```

Beschreibung DataItem-Objekt

Liefert den Namen des DataItem-Objektes zurück.

Siehe auch

ActiveProject-Eigenschaft (Seite 297)
WinCC Function Trend Control (vor WinCC V7) (Seite 287)
Tag-Objekt (Seite 146)
Ellipsensegment (Seite 158)
Layer-Objekt (Seite 129)
DataItem-Objekt (Seite 122)

NeedleColor-Eigenschaft

Beschreibung

Legt die Farbe des Zeigers fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)
ScreenItem-Objekt (Seite 134)

NormalColor-Eigenschaft

Beschreibung

Legt die Farbe des Normalbereichs auf der Skala fest. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)
ScreenItem-Objekt (Seite 134)

NumberLines-Eigenschaft

Beschreibung

Textliste

Legt für das Objekt Textliste fest, wieviele Zeilen die Auswahlliste enthalten soll, oder gibt den Wert zurück. Ist die Anzahl der projektierten Texte größer als dieser Wert, so erhält die Auswahlliste eine vertikale Bildlaufleiste.

Kombinationsfeld und Listenfeld

Legt für die Objekte Kombinationsfeld und Listenfeld fest, wieviele Einträge das Objekt enthalten soll, oder gibt den Wert zurück. Sie können maximal 100 000 Zeilen einstellen.

Der Wert des Attributs "Anzahl Zeilen" gibt gleichzeitig den oberen Grenzwert für das Attribut "Index" in der Eigenschaftsgruppe "Schrift" an. Die Änderung des Werts kann folgende Auswirkungen haben:

- Erhöhung der Anzahl: Neue Zeilen werden unten angefügt. Die Standardbeschriftung des neuen Felds ändern Sie mit dem Attribut "Text" in der Eigenschaftsgruppe "Schrift".
- Verringerung der Anzahl: Es werden alle Zeilen entfernt, bei denen der Wert des Attributs "Index" höher ist als die neue Anzahl.

Siehe auch

Textliste (Seite 208)

ScreenItem-Objekt (Seite 134)

NumItems-Eigenschaft

Beschreibung

Gibt die Anzahl der projektierten Kurven bzw. Spaltenpaare (sichtbare und unsichtbare) des Fensters zurück. Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

1.14.4.14 O

Ob - On

Object-Eigenschaft

Beschreibung

Wenn Sie ein nicht von WinCC bereitgestelltes Control verwenden, kann es vorkommen, dass die vom Control mitgebrachten Eigenschaften namensgleich sind mit den allgemeinen ScreenItem-Eigenschaften. In diesem Fall haben die ScreenItem-Eigenschaften vorrang. Auf die "verdeckten" Eigenschaften eines Fremdanbieter-Controls kann über die zusätzliche Eigenschaft "object" zugegriffen werden.

Beispiel

Sprechen Sie die Eigenschaften eines Fremdanbieter-Controls z.B. in folgender Form an:

Control.object.type

Wenn Sie nur die Form

Control.type

verwenden, werden bei Namensgleicheit die Eigenschaften des ScreenItem-Objektes verwendet.

Siehe auch

Controls (Seite 226)

ScreenItem-Objekt (Seite 134)

ObjectName-Eigenschaft

Beschreibung

Gibt den Objektnamen zurück.

- Bei Grafikobjekten den Objektnamen
- Bei Bildern den Bildnamen

STRING (readonly)

Beispiel

Das folgende Beispiel gibt die Namen aller im Bild "NewPDL1" enthaltenen Objekte aus:

```
'VBS80
Dim objScreen
Dim lngIndex
Dim lngAnswer
Dim strName
lngIndex = 1
Set objScreen = HMIRuntime.Screens("NewPDL1")
For lngIndex = 1 To objScreen.ScreenItems.Count
strName = objScreen.ScreenItems.Item(lngIndex).ObjectName
lngAnswer = MsgBox("Name of object " & lngIndex & ": " & strName, vbOKCancel)
If vbCancel = lngAnswer Then Exit For
Next
```

Bilder

Ermitteln des Bildnamens direkt über die Eigenschaft "ObjectName":

```
'VBS81
MsgBox "Screenname: " & HMIRuntime.ActiveScreen.ObjectName
```

Siehe auch

Screen-Objekt (Seite 140)

ScreenItem-Objekt (Seite 134)

ObjectSizeDeclutteringEnable-Eigenschaft

Beschreibung

Liefert die ObjectSizeDecluttering-Eigenschaft eines Bildes zurück.

Bei aktiviertem ObjectSizeDecluttering werden nur Objekte innerhalb eines bestimmten Größenbereiches angezeigt.

Die Unter-/Obergrenze für den anzuzeigenden Bereich legen Sie im Graphics Designer unter "Extras > Einstellungen > Ein-/Ausblenden" fest.

BOOLEAN Nur-Lese-Zugriff.

Beispiel:

Das Beispiel gibt die Decluttering-Eigenschaften des Bildes NewPDL1 als Trace aus.

```
'VBS157
Dim objScreen
Set objScreen = HMIRuntime.Screens("NewPDL1")
HMIRuntime.Trace "Min: " & objScreen.ObjectSizeDeclutteringMin & vbNewLine
HMIRuntime.Trace "Max: " & objScreen.ObjectSizeDeclutteringMax & vbNewLine
HMIRuntime.Trace "Enable: " & objScreen.LayerDeclutteringEnable & vbNewLine
```

Siehe auch

Screen-Objekt (Seite 140)

ObjectSizeDeclutteringMax-Eigenschaft

Beschreibung

Über die Eigenschaft ObjectSizeDeclutteringMax kann der obere Größenbereich eines Bildes ausgelesen werden.

Objekte, die größer als die angegebene Größe in Pixeln sind, werden bei aktiviertem ObjectSizeDecluttering nicht mehr angezeigt.

LONG Nur-Lese-Zugriff.

Beispiel:

Das Beispiel gibt die Decluttering-Eigenschaften des Bildes NewPDL1 als Trace aus.

```
'VBS157
Dim objScreen
Set objScreen = HMIRuntime.Screens("NewPDL1")
HMIRuntime.Trace "Min: " & objScreen.ObjectSizeDeclutteringMin & vbNewLine
HMIRuntime.Trace "Max: " & objScreen.ObjectSizeDeclutteringMax & vbNewLine
HMIRuntime.Trace "Enable: " & objScreen.LayerDeclutteringEnable & vbNewLine
```

Siehe auch

Screen-Objekt (Seite 140)

ObjectSizeDeclutteringMin-Eigenschaft

Beschreibung

Über die Eigenschaft ObjectSizeDeclutteringMin kann der untere Größenbereich eines Bildes ausgelesen werden.

Objekte, die kleiner als die angegebene Größe in Pixeln sind, werden bei aktiviertem ObjectSizeDecluttering nicht mehr angezeigt.

LONG Nur-Lese-Zugriff.

Beispiel:

Das Beispiel gibt die Decluttering-Eigenschaften des Bildes NewPDL1 als Trace aus.

```
'VBS157
Dim objScreen
Set objScreen = HMIRuntime.Screens("NewPDL1")
HMIRuntime.Trace "Min: " & objScreen.ObjectSizeDeclutteringMin & vbNewLine
HMIRuntime.Trace "Max: " & objScreen.ObjectSizeDeclutteringMax & vbNewLine
HMIRuntime.Trace "Enable: " & objScreen.LayerDeclutteringEnable & vbNewLine
```

Siehe auch

Screen-Objekt (Seite 140)

OffsetLeft-Eigenschaft

Beschreibung

Legt den Abstand des Bildes vom linken Rand des Bildfensters fest oder gibt ihn zurück.

Das Bild wird aus dem Bildfenster ausgeschnitten dargestellt. Die Bildlaufleisten befinden sich am linken und oberen Rand des Bildes. Wenn Sie das Bild im Bildfenster mit der horizontalen und vertikalen Verschiebung der Bildlaufleisten darstellen wollen, verwenden Sie für die Verschiebung die Eigenschaften "ScrollPositionX" und "ScrollPositionY".

Siehe auch

ScrollPositionY-Eigenschaft (Seite 539)

ScrollPositionX-Eigenschaft (Seite 539)

Bildfenster (Seite 190)

ScreenItem-Objekt (Seite 134)

OffsetTop-Eigenschaft

Beschreibung

Legt den Abstand des Bildes vom oberen Rand des Bildfensters fest oder gibt ihn zurück.

Das Bild wird aus dem Bildfenster ausgeschnitten dargestellt. Die Bildlaufleisten befinden sich am linken und oberen Rand des Bildes. Wenn Sie das Bild im Bildfenster mit der horizontalen und vertikalen Verschiebung der Bildlaufleisten darstellen wollen, verwenden Sie für die Verschiebung die Eigenschaften "ScrollPositionX" und "ScrollPositionY".

Siehe auch

ScrollPositionY-Eigenschaft (Seite 539)

ScrollPositionX-Eigenschaft (Seite 539)

Bildfenster (Seite 190)

ScreenItem-Objekt (Seite 134)

OneY-Eigenschaft

Beschreibung

TRUE, wenn nur die Y-Achse der Kurve im Vordergrund angezeigt wird anstatt aller Y-Achsen der dargestellten Kurven. BOOLEAN Schreib-Lese-Zugriff.

Online-Eigenschaft (vor WinCC V7)

Beschreibung

Dient dem Starten bzw. Stoppen der Aktualisierung.

- 0: Die aktualisierte Darstellung wird angehalten. Die Werte werden zwischengespeichert und bei erneuter Aktivierung der Schaltfläche nachgetragen.
- -1: Die aktualisierte Darstellung wird fortgesetzt.

Siehe auch

ScreenItem-Objekt (Seite 134)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

Online-Eigenschaft

Aktualisierung starten - Online

Legt fest, ob beim Bildaufschlag in Runtime die Darstellung der Werte aktualisiert wird.

Wert	Beschreibung
TRUE	Die Werte werden bei Bildaufschlag aktualisiert.
FALSE	Die Werte werden bei Bildaufschlag nicht aktualisiert.

Das Attribut ist mit dem Namen **Online** dynamisierbar. Der Datentyp ist BOOLEAN.

OnTop-Eigenschaft

Beschreibung

TRUE, wenn das Objekt in Runtime immer im Vordergrund bleibt. Nur Lese-Zugriff.

Siehe auch

Bildfenster (Seite 190)

Applikationsfenster (Seite 185)

ScreenItem-Objekt (Seite 134)

Op

OperationMessage-Eigenschaft

Beschreibung

TRUE, wenn bei einer erfolgten Bedienung eine Meldung ausgegeben werden soll. BOOLEAN Schreib-Lese-Zugriff.
Die Bedienung wird an das Meldesystem gesandt und archiviert. Über das Meldesystem kann eine Meldung z. B. in einer Meldezeile ausgeben werden.

Besonderheit bei EA-Feld, Textliste und Slider

Der Grund für die Bedienung kann nur eingegeben werden, wenn die Eigenschaft "OperationReport" auf TRUE gesetzt ist.

Siehe auch

Slider (Seite 221)

Textliste (Seite 208)

Radio-Box (Seite 217)

Check-Box (Seite 215)

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

OperatorMessageID-Eigenschaft

OperatorMessageID

Festgelegte Zuordnung von Ident-Nummer und Auslöseereignis im WinCC OnlineTableControl:

Wert	Beschreibung	Erklärung
5	EditValue	Auslöseereignis "Archivwert ändern"
6	InsertValue	Auslöseereignis "Archivwert erzeugen"

Das Attribut ist mit dem Namen **OperatorMessageID** dynamisierbar. Der Datentyp ist LONG.

OperatorMessageIndex-Eigenschaft

OperatorMessageIndex

Referenziert das Ereignis einer Archivwertänderung für eine Bedienmeldung. Bei Verwendung des Attributs können Sie einer bestimmten Bedienmeldung die Werte anderer Attribute zuweisen.

Folgende Werte stehen zur Verfügung:

Wert	Erklärung
0	Auslöseereignis "Archivwert ändern"
1	Auslöseereignis "Archivwert erzeugen"

Das Attribut ist mit dem Namen **OperatorMessageIndex** dynamisierbar. Der Datentyp ist LONG.

OperatorMessageName-Eigenschaft

Objektname - OperatorMessageName

Zeigt bei Meldeereignissen für Bedienmeldungen den Namen an, der mit dem Attribut "OperatorMessageIndex" referenziert wird. Den Namen können Sie nicht ändern.

Folgende Namen für Meldeereignisse stehen zur Verfügung:

Wert	Erklärung
Lock	Das Meldeereignis "Sperren".
Unlock	Das Meldeereignis "Freigeben".
Hide	Das Meldeereignis "Ausblenden".
Unhide	Das Meldeereignis "Einblenden".
Quit	Das Meldeereignis "Quittieren".

Das Attribut ist mit dem Namen **OperatorMessageName** dynamisierbar. Der Datentyp ist STRING.

Siehe auch

So konfigurieren Sie Bedienmeldungen

OperatorMessageNumber-Eigenschaft

Meldenummer - OperatorMessageNumber

Legen Sie eine Meldenummer für die Bedienmeldung des ausgewählten Meldeereignisses fest, wenn Sie nicht die Bedienmeldung von WinCC verwenden wollen.

Das Attribut ist mit dem Namen **OperatorMessageNumber** dynamisierbar. Der Datentyp ist LONG.

OperatorMessageSelected-Eigenschaft

Bedienmeldungen beim - OperatorMessageSelected

Aktivieren Sie die Meldeereignisse in der Liste, bei denen Bedienmeldungen ausgelöst werden.

Das Attribut ist mit dem Namen **OperatorMessageSelected** dynamisierbar. Der Datentyp ist BOOLEAN.

OperatorMessageSource1-Eigenschaft

Quelle - OperatorMessageSource1

Bestimmen Sie einen Meldeblock der bedienten Meldung, der zum "Prozesswertblock 1" der hier projektierten Bedienmeldung hinzugefügt wird.

Zum Beispiel wollen Sie beim Sperren einer Meldung eine Bedienmeldung erzeugen. Der Inhalt von "Anwendertextblock 1" der gesperrten Meldung, z. B. "Motor gestört", soll im "Prozesswertblock 1" der Bedienmeldung angezeigt werden. Wählen Sie dazu unter Prozesswert "1" als Meldeblock der bedienten Meldung "Anwendertextblock 1" aus.

Das Attribut ist mit dem Namen **OperatorMessageSource1** dynamisierbar. Der Datentyp ist STRING.

OperatorMessageSource2-Eigenschaft

Quelle - OperatorMessageSource2

Bestimmen Sie einen Meldeblock der bedienten Meldung, der zum "Prozesswertblock 2" der hier projektierten Bedienmeldung hinzugefügt wird.

Zum Beispiel wollen Sie beim Sperren einer Meldung eine Bedienmeldung erzeugen. Der Inhalt von "Anwendertextblock 1" der gesperrten Meldung, z. B. "Motor gestört", soll im "Prozesswertblock 2" der Bedienmeldung angezeigt werden. Wählen Sie dazu unter Prozesswert "2" als Meldeblock der bedienten Meldung "Anwendertextblock 1" aus.

Das Attribut ist mit dem Namen **OperatorMessageSource2** dynamisierbar. Der Datentyp ist STRING.

OperatorMessageSource3-Eigenschaft

Quelle - OperatorMessageSource3

Bestimmen Sie einen Meldeblock der bedienten Meldung, der zum "Prozesswertblock 3" der hier projektierten Bedienmeldung hinzugefügt wird.

Zum Beispiel wollen Sie beim Sperren einer Meldung eine Bedienmeldung erzeugen. Der Inhalt von "Anwendertextblock 1" der gesperrten Meldung, z. B. "Motor gestört", soll im "Prozesswertblock 3" der Bedienmeldung angezeigt werden. Wählen Sie dazu unter Prozesswert "3" als Meldeblock der bedienten Meldung "Anwendertextblock 1" aus.

Das Attribut ist mit dem Namen **OperatorMessageSource3** dynamisierbar. Der Datentyp ist STRING.

OperatorMessageSource4-Eigenschaft

Quelle - OperatorMessageSource4

Bestimmen Sie einen Meldeblock der bedienten Meldung, der zum "Prozesswertblock 4" der hier projektierten Bedienmeldung hinzugefügt wird.

Zum Beispiel wollen Sie beim Sperren einer Meldung eine Bedienmeldung erzeugen. Der Inhalt von "Anwendertextblock 1" der gesperrten Meldung, z. B. "Motor gestört", soll im "Prozesswertblock 4" der Bedienmeldung angezeigt werden. Wählen Sie dazu unter Prozesswert "4" als Meldeblock der bedienten Meldung "Anwendertextblock 1" aus.

Das Attribut ist mit dem Namen **OperatorMessageSource4** dynamisierbar. Der Datentyp ist STRING.

OperatorMessageSource5-Eigenschaft

Quelle - OperatorMessageSource5

Bestimmen Sie einen Meldeblock der bedienten Meldung, der zum "Prozesswertblock 5" der hier projektierten Bedienmeldung hinzugefügt wird.

Zum Beispiel wollen Sie beim Sperren einer Meldung eine Bedienmeldung erzeugen. Der Inhalt von "Anwendertextblock 1" der gesperrten Meldung, z. B. "Motor gestört", soll im "Prozesswertblock 5" der Bedienmeldung angezeigt werden. Wählen Sie dazu unter Prozesswert "5" als Meldeblock der bedienten Meldung "Anwendertextblock 1" aus.

Das Attribut ist mit dem Namen **OperatorMessageSource5** dynamisierbar. Der Datentyp ist STRING.

OperatorMessageSource6-Eigenschaft

Quelle - OperatorMessageSource6

Bestimmen Sie einen Meldeblock der bedienten Meldung, der zum "Prozesswertblock 6" der hier projektierten Bedienmeldung hinzugefügt wird.

Zum Beispiel wollen Sie beim Sperren einer Meldung eine Bedienmeldung erzeugen. Der Inhalt von "Anwendertextblock 1" der gesperrten Meldung, z. B. "Motor gestört", soll im "Prozesswertblock 6" der Bedienmeldung angezeigt werden. Wählen Sie dazu unter Prozesswert "6" als Meldeblock der bedienten Meldung "Anwendertextblock 1" aus.

Das Attribut ist mit dem Namen **OperatorMessageSource6** dynamisierbar. Der Datentyp ist STRING.

OperatorMessageSource7-Eigenschaft

Quelle - OperatorMessageSource7

Bestimmen Sie einen Meldeblock der bedienten Meldung, der zum "Prozesswertblock 7" der hier projektierten Bedienmeldung hinzugefügt wird.

Zum Beispiel wollen Sie beim Sperren einer Meldung eine Bedienmeldung erzeugen. Der Inhalt von "Anwendertextblock 1" der gesperrten Meldung, z. B. "Motor gestört", soll im "Prozesswertblock 7" der Bedienmeldung angezeigt werden. Wählen Sie dazu unter Prozesswert "7" als Meldeblock der bedienten Meldung "Anwendertextblock 1" aus.

Das Attribut ist mit dem Namen **OperatorMessageSource7** dynamisierbar. Der Datentyp ist STRING.

OperatorMessageSource8-Eigenschaft

Quelle - OperatorMessageSource8

Bestimmen Sie einen Meldeblock der bedienten Meldung, der zum "Prozesswertblock 8" der hier projektierten Bedienmeldung hinzugefügt wird.

Zum Beispiel wollen Sie beim Sperren einer Meldung eine Bedienmeldung erzeugen. Der Inhalt von "Anwendertextblock 1" der gesperrten Meldung, z. B. "Motor gestört", soll im "Prozesswertblock 8" der Bedienmeldung angezeigt werden. Wählen Sie dazu unter Prozesswert "8" als Meldeblock der bedienten Meldung "Anwendertextblock 1" aus.

Das Attribut ist mit dem Namen **OperatorMessageSource8** dynamisierbar. Der Datentyp ist STRING.

OperatorMessageSource9-Eigenschaft

Quelle - OperatorMessageSource9

Bestimmen Sie einen Meldeblock der bedienten Meldung, der zum "Prozesswertblock 9" der hier projektierten Bedienmeldung hinzugefügt wird.

Zum Beispiel wollen Sie beim Sperren einer Meldung eine Bedienmeldung erzeugen. Der Inhalt von "Anwendertextblock 1" der gesperrten Meldung, z. B. "Motor gestört", soll im "Prozesswertblock 9" der Bedienmeldung angezeigt werden. Wählen Sie dazu unter Prozesswert "9" als Meldeblock der bedienten Meldung "Anwendertextblock 1" aus.

Das Attribut ist mit dem Namen **OperatorMessageSource9** dynamisierbar. Der Datentyp ist STRING.

OperatorMessageSource10-Eigenschaft

Quelle - OperatorMessageSource10

Bestimmen Sie einen Meldeblock der bedienten Meldung, der zum "Prozesswertblock 10" der hier projektierten Bedienmeldung hinzugefügt wird.

Zum Beispiel wollen Sie beim Sperren einer Meldung eine Bedienmeldung erzeugen. Der Inhalt von "Anwendertextblock 1" der gesperrten Meldung, z. B. "Motor gestört", soll im "Prozesswertblock 10" der Bedienmeldung angezeigt werden. Wählen Sie dazu unter Prozesswert "10" als Meldeblock der bedienten Meldung "Anwendertextblock 1" aus.

Das Attribut ist mit dem Namen **OperatorMessageSource10** dynamisierbar. Der Datentyp ist STRING.

OperatorMessageSourceType1-Eigenschaft

Übergabe als - OperatorMessageSourceType1

Legt fest, in welchem Format der Inhalt der Quelle übergeben wird.

Folgende Formate stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Text	Der Inhalt der Quelle wird als Text übergeben.
1	Wert	Der Inhalt der Quelle wird als Wert übergeben.

Das Attribut ist mit dem Namen **OperatorMessageSourceType1** dynamisierbar. Der Datentyp ist LONG.

OperatorMessageSourceType2-Eigenschaft

Übergabe als - OperatorMessageSourceType2

Legt fest, in welchem Format der Inhalt der Quelle übergeben wird.

Folgende Formate stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Text	Der Inhalt der Quelle wird als Text übergeben.
1	Wert	Der Inhalt der Quelle wird als Wert übergeben.

Das Attribut ist mit dem Namen **OperatorMessageSourceType2** dynamisierbar. Der Datentyp ist LONG.

OperatorMessageSourceType3-Eigenschaft

Übergabe als - OperatorMessageSourceType3

Legt fest, in welchem Format der Inhalt der Quelle übergeben wird.

Folgende Formate stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Text	Der Inhalt der Quelle wird als Text übergeben.
1	Wert	Der Inhalt der Quelle wird als Wert übergeben.

Das Attribut ist mit dem Namen **OperatorMessageSourceType3** dynamisierbar. Der Datentyp ist LONG.

OperatorMessageSourceType4-Eigenschaft

Übergabe als - OperatorMessageSourceType4

Legt fest, in welchem Format der Inhalt der Quelle übergeben wird.

Folgende Formate stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Text	Der Inhalt der Quelle wird als Text übergeben.
1	Wert	Der Inhalt der Quelle wird als Wert übergeben.

Das Attribut ist mit dem Namen **OperatorMessageSourceType4** dynamisierbar. Der Datentyp ist LONG.

OperatorMessageSourceType5-Eigenschaft

Übergabe als - OperatorMessageSourceType5

Legt fest, in welchem Format der Inhalt der Quelle übergeben wird.

Folgende Formate stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Text	Der Inhalt der Quelle wird als Text übergeben.
1	Wert	Der Inhalt der Quelle wird als Wert übergeben.

Das Attribut ist mit dem Namen **OperatorMessageSourceType5** dynamisierbar. Der Datentyp ist LONG.

OperatorMessageSourceType6-Eigenschaft

Übergabe als - OperatorMessageSourceType6

Legt fest, in welchem Format der Inhalt der Quelle übergeben wird.

Folgende Formate stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Text	Der Inhalt der Quelle wird als Text übergeben.
1	Wert	Der Inhalt der Quelle wird als Wert übergeben.

Das Attribut ist mit dem Namen **OperatorMessageSourceType6** dynamisierbar. Der Datentyp ist LONG.

OperatorMessageSourceType7-Eigenschaft

Übergabe als - OperatorMessageSourceType7

Legt fest, in welchem Format der Inhalt der Quelle übergeben wird.

Folgende Formate stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Text	Der Inhalt der Quelle wird als Text übergeben.
1	Wert	Der Inhalt der Quelle wird als Wert übergeben.

Das Attribut ist mit dem Namen **OperatorMessageSourceType7** dynamisierbar. Der Datentyp ist LONG.

OperatorMessageSourceType8-Eigenschaft

Übergabe als - OperatorMessageSourceType8

Legt fest, in welchem Format der Inhalt der Quelle übergeben wird.

Folgende Formate stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Text	Der Inhalt der Quelle wird als Text übergeben.
1	Wert	Der Inhalt der Quelle wird als Wert übergeben.

Das Attribut ist mit dem Namen **OperatorMessageSourceType8** dynamisierbar. Der Datentyp ist LONG.

OperatorMessageSourceType9-Eigenschaft

Übergabe als - OperatorMessageSourceType9

Legt fest, in welchem Format die Übergabe der Quelle erfolgt.

Folgende Formate stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Text	Übergabe der Quelle als Text.
1	Wert	Übergabe der Quelle als Wert.

Das Attribut ist mit dem Namen **OperatorMessageSourceType9** dynamisierbar. Der Datentyp ist LONG.

OperatorMessageSourceType10-Eigenschaft

Übergabe als - OperatorMessageSourceType10

Legt fest, in welchem Format der Inhalt der Quelle übergeben wird.

Folgende Formate stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Text	Der Inhalt der Quelle wird als Text übergeben.
1	Wert	Der Inhalt der Quelle wird als Wert übergeben.

Das Attribut ist mit dem Namen **OperatorMessageSourceType10** dynamisierbar. Der Datentyp ist LONG.

OperationReport-Eigenschaft

Beschreibung

TRUE, wenn der Grund für eine Bedienung mitprotokolliert werden soll. BOOLEAN Schreib-Lese-Zugriff.
Bei der Bedienung des Objekts in Runtime erscheint dann ein Dialog, in dem der Bediener den Grund der Bedienung als Text eingeben kann. Die Bedienung wird an das Meldeesystem gesandt und archiviert.

Siehe auch

Slider (Seite 221)

Textliste (Seite 208)

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

Or - Ou

Orientation-Eigenschaft

Beschreibung

TRUE, wenn der Text im Objekt horizontal dargestellt. BOOLEAN Schreib-Lese-Zugriff.

Beschreibung Objekt-Typ "Verbinder"

Ändert die Orientierung des Verbinders. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Verbinder (Seite 180)

Statischer Text (Seite 178)

Textliste (Seite 208)

Radio-Box (Seite 217)

Check-Box (Seite 215)

Button (Seite 212)

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

OuterBevelStyle-Eigenschaft

Beschreibung

Legt den 3D-Effekt für den Außenrahmen des Objekts fest.

- 0: Kein Rahmen.
- 1: Der Rahmen wird vertieft dargestellt.
- 2: Der Rahmen wird erhaben dargestellt.
- 3: Der Rahmen wird einfarbig, ohne 3D-Effekt dargestellt. Die Rahmenfarbe wird durch die Eigenschaft "BevelColorUp" bestimmt.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

OuterBevelWidth-Eigenschaft

Beschreibung

Legt die Breite des Außenrahmens in Pixel fest.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

Outline-Eigenschaft

Beschreibung

TRUE, wenn die Schaltfläche zusätzlich zum 3D-Rahmen mit einem schwarzen Rahmen versehen wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Push Button Control (Seite 271)

ScreenItem-Objekt (Seite 134)

OutputFormat-Eigenschaft

Beschreibung

Gibt den Wert für die Darstellung des Ausgabewerts zurück und setzt ihn. Die Darstellung ist abhängig vom Datenformat.

Siehe auch

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

OutputValue-Eigenschaft

Beschreibung

Legt die Voreinstellung für den anzuzeigenden Wert fest oder gibt sie zurück.
In Runtime wird dieser Wert verwendet, wenn beim Start des Bildes die zugehörige Variable nicht angeschlossen oder nicht aktualisiert ist.

Siehe auch

Textliste (Seite 208)

EA-Feld (Seite 195)

ScreenItem-Objekt (Seite 134)

1.14.4.15 P

Pa - Pe

PageMode-Eigenschaft

Blättern aktivieren - PageMode

Legt fest, ob das Blättern in der Langzeitarchivliste möglich ist. Damit können Sie alle Meldungen des Umlaufarchivs seitenweise in der Langzeitarchivliste anzeigen. Über die Eigenschaft "Meldungen pro Seite" bzw. "PageModeMessageNumber" bestimmen Sie die Anzahl der Meldungen, die pro Seite angezeigt werden.

Wenn das Blättern aktiviert ist, können Sie die Schaltflächen zum Blättern in der Symbolleiste verwenden.

Wert	Erklärung
TRUE	Das Blättern in der Langzeitarchivliste ist möglich.
FALSE	Das Blättern in der Langzeitarchivliste ist nicht möglich.

Das Attribut ist mit dem Namen **PageMode** dynamisierbar. Der Datentyp ist BOOLEAN.

PageModeMessageNumber-Eigenschaft

Meldungen pro Seite - PageModeMessageNumber

Legt die Anzahl der Meldungen fest, die beim Blättern in der Langzeitarchivliste pro Seite angezeigt werden.

Das Attribut ist mit dem Namen **PageModeMessageNumber** dynamisierbar. Der Datentyp ist LONG.

Parent-Eigenschaft

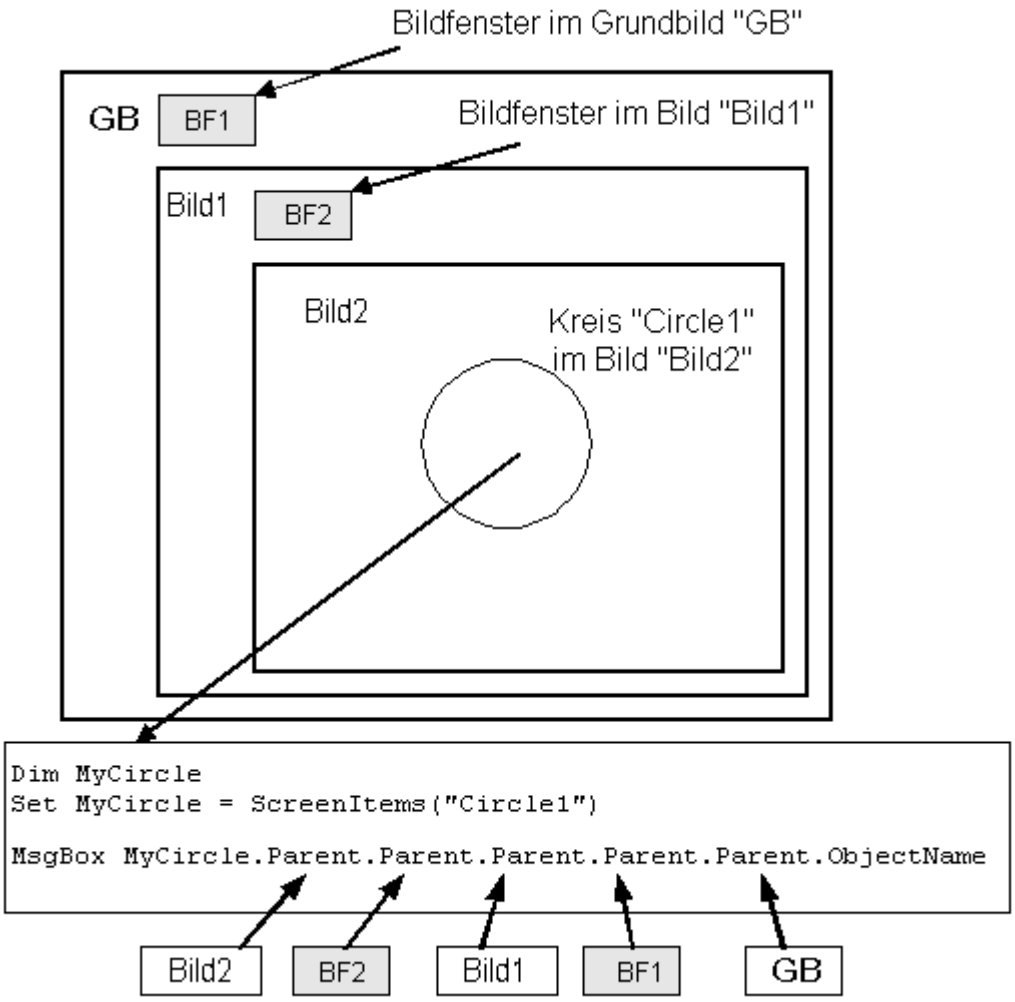
Beschreibung

Liefert eine Referenz auf das übergeordnete Objekt.

Auf Objekte innerhalb des VBS-Objektmodells wird hierarchisch zugegriffen. Über Screen und Screenitems kann man sich in der Bildhierarchie abwärts bewegen. Mit der Parent-Eigenschaft kann man sich in der Hierarchie aufwärts bewegen.

Verwendung

Die Parent-Eigenschaft können Sie innerhalb einer Objekthierarchie beliebig oft verwenden. Im Folgenden sehen Sie schematisch, wie Sie auf alle Elemente einer Hierarchie zugreifen:



Der Befehl

```
MsgBox MyCircle.Parent.Objectname
```

liefert den Namen von "Bild2" zurück, das in der Objekthierarchie eine Ebene höher liegt als das ursprüngliche ScreenItem-Objekt "Circle1".

Wenn Sie z.B. "Parent" dreimal verwenden, gehen Sie in der Objekthierarchie drei Ebenen höher:

```
MsgBox MyCircle.Parent.Parent.Parent.Objectname
```

liefert den Namen von Bild 1.

Begründung:

- Ursprünglich referenziert ist ScreenItem "Circle1"
- "Circle1" liegt in "Bild2" (1. Ebene)
- "Bild2" liegt in Bildfenster2 "BF2" (2. Ebene)
- "BF2" liegt in "Bild 1" (3. Ebene)

Beispiel

Im folgenden Beispiel wird der Objektname des Parent-Objekts ausgegeben:

```
'VBS120
Dim objCircle
Set objCircle = HMIRuntime.Screens("ScreenWindow1").ScreenItems("Circle1")
MsgBox objCircle.Parent.ObjectName
```

Siehe auch

Bildfenster (Seite 190)

Screen-Objekt (Seite 140)

Objekte und Auflistungen (Seite 117)

PasswordLevel-Eigenschaft

Beschreibung

Legt die Berechtigung für die Bedienung (z.B. keine Eingabe oder keine Auslösung von Aktionen) für das Objekt fest.

Siehe auch

ScreenItem-Objekt (Seite 134)

Path-Eigenschaft

Beschreibung

Gibt den Pfad des aktuellen Projektes zurück (ohne den Dateinamen). Auf einen WinCC-Client ohne eigenes Projekt wird der Pfad im UNC-Format zurückgeliefert, sonst der lokale Pfad.

STRING (Nur-Lesen-Zugriff)

Beispiel

Das folgende Beispiel gibt den Projektpfad als Trace aus:

```
'VBS161
HMIRuntime.Trace "Path: " & HMIRuntime.ActiveProject.Path & vbNewLine
```

Siehe auch

Project-Objekt (Seite 133)

PercentageAxis-Eigenschaft

PercentageAxis

Legt fest, ob im Kurvenfenster zu den Wertachsen auch eine Achse mit prozentualer Skalierung angezeigt wird.

Wert	Erklärung
TRUE	Eine Achse mit prozentualer Skalierung wird angezeigt.
FALSE	Keine Achse mit prozentualer Skalierung wird angezeigt.

Das Attribut ist mit dem Namen **PercentageAxis** dynamisierbar. Der Datentyp ist BOOLEAN.

PercentageAxisAlign-Eigenschaft

PercentageAxisAlign

Legt die Ausrichtung der Achse mit prozentualer Skalierung im Kurvenfenster fest.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	links	Die Achse mit prozentualer Skalierung ist links ausgerichtet.
1	rechts	Die Achse mit prozentualer Skalierung ist rechts ausgerichtet.

Das Attribut ist mit dem Namen **PercentageAxisAlign** dynamisierbar. Der Datentyp ist LONG.

PercentageAxisColor-Eigenschaft

PercentageAxisColor

Gibt die Farbe der Achse mit prozentualer Skalierung an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **PercentageAxisColor** dynamisierbar. Der Datentyp ist LONG.

PersistentRT-Eigenschaft

Beschreibung

TRUE, wenn veränderte Einstellungen des Fensters auch nach einem Bildwechsel erhalten bleiben sollen. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "AllowPersistance".

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

PersistentRTCS-Eigenschaft

Beschreibung

TRUE, wenn veränderte Einstellungen nach einem Bildwechsel erhalten bleiben und in das Konfigurationssystem übernommen werden. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "AllowPersistance". BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

PersistentRTCSPermission-Eigenschaft

Beschreibung

Legt die Bedienberechtigung fest, die notwendig ist, um die Einstellung bezüglich der Persistenz verändern zu können. Der einzugebende Wert entspricht der Nummer, die die gewünschte Berechtigung im User Administrator besitzt. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "AllowPersistance" (gilt nicht für WinCC Alarm Control).

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

PersistentRTPermission-Eigenschaft

Beschreibung

Legt die Bedienberechtigung fest, die notwendig ist, um die Einstellung bezüglich der Persistenz im Runtime verändern zu können. Der einzugebende Wert entspricht der Nummer, die die gewünschte Berechtigung im User Administrator besitzt. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "AllowPersistance" (gilt nicht für WinCC Alarm Control).

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

Pi

PicDeactReferenced-Eigenschaft

Beschreibung

TRUE, wenn das zugeordnete Bild für den Zustand "Deaktiviert" im Objekt RoundButton gespeichert wird. Sonst wird nur der zugehörige Objektverweis gespeichert. Nur Lese-Zugriff.

Siehe auch

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

PicDeactTransparent-Eigenschaft

Beschreibung

Legt fest, welche Farbe des zugeordneten Bitmap-Objekts (.bmp, .dib) für den Zustand "Deaktiviert" auf "transparent" gesetzt werden soll oder gibt die Farbe zurück. LONG Schreib-Lese-Zugriff.
Die Farbe wird nur dann auf "transparent" gesetzt, wenn die Eigenschaft "PicDeactUseTransColor" den Wert TRUE hat.

Siehe auch

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

PicDeactUseTransColor-Eigenschaft

Beschreibung

TRUE, wenn die mit der Eigenschaft "PicDeactTransparent" festgelegte Transparentfarbe für den Zustand "Deaktiviert" verwendet werden soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

PicDownReferenced-Eigenschaft

Beschreibung

TRUE, wenn das zugeordnete Bild für den Zustand "Ein" gespeichert wird. Sonst wird nur der zugehörige Objektverweis gespeichert. Nur Lese-Zugriff.

Siehe auch

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

PicDownTransparent-Eigenschaft

Beschreibung

Legt fest, welche Farbe des zugeordneten Bitmap-Objekts (.bmp, .dib) für den Zustand "Ein" auf "transparent" gesetzt werden soll oder gibt die Farbe zurück. LONG Schreib-Lese-Zugriff. Die Farbe wird nur dann auf "transparent" gesetzt, wenn die Eigenschaft "PicDownUseTransColor" den Wert TRUE hat.

Siehe auch

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

PicDownUseTransColor-Eigenschaft

Beschreibung

TRUE, wenn die mit der Eigenschaft "PicDownTransparent" festgelegte Transparentfarbe für den Zustand "Ein" verwendet werden soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

PicReferenced-Eigenschaft

Beschreibung

TRUE, wenn das zugeordnete Bild referenziert und nicht im Objekt gespeichert wird. Nur Lese-Zugriff.

Siehe auch

Grafik-Objekt (Seite 198)

ScreenItem-Objekt (Seite 134)

PictAlignment-Eigenschaft

Beschreibung

Legt die Bildausrichtung des Bildes auf dem Button bzw. Rundbutton fest oder gibt sie zurück.

LONG Schreib-Lese-Zugriff.

PicTransColor-Eigenschaft

Beschreibung

Legt fest, welche Farbe des zugeordneten Bitmap-Objekts (.bmp, .dib) auf "transparent" gesetzt werden soll oder gibt die Farbe zurück. LONG Schreib-Lese-Zugriff.
Die Farbe wird nur dann auf "transparent" gesetzt, wenn die Eigenschaft "PicUseTransColor" den Wert TRUE hat.

Siehe auch

Grafik-Objekt (Seite 198)

ScreenItem-Objekt (Seite 134)

Picture-Eigenschaft

Beschreibung

Gibt den Bildnamen des Hintergrundbilds für den rechteckigen Hintergrund sowohl der Analog- wie auch der Digitaluhr zurück. Nur Lese-Zugriff

Siehe auch

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

PictureBack-Eigenschaft

Beschreibung

Gibt den Bildnamen des Bildes für den Hintergrund des Objekts zurück. Nur Lese-Zugriff.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

PictureDeactivated-Eigenschaft

Beschreibung

Legt das Bild fest, das im Zustand "Deaktiviert" angezeigt wird oder gibt den Bildnamen zurück. Das Bild (*.BMP oder *.DIB) muss sich im Verzeichnis "GraCS" des aktuellen Projekts befinden, damit es eingebunden werden kann.

Siehe auch

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

PictureDown-Eigenschaft

Beschreibung

Legt das Bild fest, das im Zustand "Ein" angezeigt wird oder gibt den Bildnamen zurück. Das Bild (*.BMP oder *.DIB) muss sich im Verzeichnis "GraCS" des aktuellen Projekts befinden, damit es eingebunden werden kann.

Siehe auch

Button (Seite 212)

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

PictureName-Eigenschaft

Beschreibung

Legt das Bild fest, das im Grafik-Objekt in Runtime angezeigt wird oder gibt den Bildnamen zurück.

Das Bild (*.BMP oder *.DIB) muss sich im Verzeichnis "GraCS" des aktuellen Projekts befinden, damit es eingebunden werden kann.

Siehe auch

Grafik-Objekt (Seite 198)

ScreenItem-Objekt (Seite 134)

PictureSelected-Eigenschaft

Beschreibung

Gibt den Bildnames des Bildes zurück, das im Zustand "Ein" angezeigt wird. "AutoSize" steuert die Anpassung der Größe von Bild und Schaltfläche. Nur Lese-Zugriff.

Siehe auch

WinCC Push Button Control (Seite 271)

ScreenItem-Objekt (Seite 134)

PictureSizeMode-Eigenschaft

PictureSizeMode

Legt die Größenanpassung zwischen Bild und Control fest.

Wert	Bezeichnung	Erläuterung
0	Fit size to content	Das Control wird an die Bildgröße angepasst.
1	Fit content to size	Das Bild wird an das Control angepasst bzw. skaliert.

Das Attribut ist mit dem Namen **PictureSizeMode** dynamisierbar. Der Datentyp ist LONG.

PictureThumb-Eigenschaft

Beschreibung

Gibt den Bildnamen des Hintergrundbilds für den Schieber zurück. Nur Lese-Zugriff.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

PictureUnselected-Eigenschaft

Beschreibung

Gibt den Bildnamen des Bildes zurück, das im Zustand "Aus" angezeigt wird. "AutoSize" steuert die Anpassung der Größe von Bild und Schaltfläche. Nur Lese-Zugriff.

Siehe auch

WinCC Push Button Control (Seite 271)

ScreenItem-Objekt (Seite 134)

PictureUp-Eigenschaft

Beschreibung

Legt das Bild fest, das im Zustand "Aus" angezeigt wird oder gibt den Bildnamen zurück. Das Bild (*.BMP oder *.DIB) muss sich im Verzeichnis "GraCS" des aktuellen Projekts befinden, damit es eingebunden werden kann.

Siehe auch

Rundbutton (Seite 219)

Button (Seite 212)

ScreenItem-Objekt (Seite 134)

PicUpReferenced-Eigenschaft

Beschreibung

TRUE, wenn das zugeordnete Bild für den Zustand "Aus" im Objekt gespeichert wird. Sonst wird nur der zugehörige Objektverweis gespeichert. Nur Lese-Zugriff.

Siehe auch

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

PicUpTransparent-Eigenschaft

Beschreibung

Legt fest, welche Farbe des zugeordneten Bitmap-Objekts (.bmp, .dib) für den Zustand "Aus" auf "transparent" gesetzt werden soll oder gibt die Farbe zurück. LONG Schreib-Lese-Zugriff. Die Farbe wird nur dann auf "transparent" gesetzt, wenn die Eigenschaft "PicUpUseTransColor" den Wert TRUE hat.

Siehe auch

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

PicUpUseTransColor-Eigenschaft

Beschreibung

TRUE, wenn die mit der Eigenschaft "PicUpTransparent" festgelegte Transparentfarbe für den Zustand "Aus" verwendet werden soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

PicUseTransColor-Eigenschaft

Beschreibung

TRUE, wenn die mit der Eigenschaft "PicDeactTransparent" festgelegte Transparentfarbe für den Zustand "Deaktiviert" verwendet werden soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Grafik-Objekt (Seite 198)

ScreenItem-Objekt (Seite 134)

Pl - Pr

PlayEndless-Eigenschaft

PlayEndless

Legt fest, ob im Control die Filme endlos wiedergegeben werden.

Das Attribut ist mit dem Namen **PlayEndless** dynamisierbar. Der Datentyp ist BOOLEAN.

PointCount-Eigenschaft

Beschreibung

Legt die Anzahl der Eckpunkte fest oder gibt sie zurück. Jeder Eckpunkt hat Positionskoordinaten und wird über einen Index identifiziert.

Siehe auch

Polygonzug (Seite 170)

Polygon (Seite 168)

ScreenItem-Objekt (Seite 134)

Position-Eigenschaft

Beschreibung

Legt die Voreinstellung für die Position des Schiebers fest.

In Runtime wird dieser Wert als Anfangswert verwendet.

Für die Bedienung des mit diesem Attribut verbundenen Prozesswerts ist es notwendig, den Prozesswert auch mit dem Ereignis "Position" zu verbinden. Sie finden das Ereignis "Position"

in der Registerkarte "Ereignis", im Themenbaum unter SliderCtrl\Propertythemen\Control Eigenschaften\Wert.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

Precisions-Eigenschaft

Beschreibung

WinCC Online Table Control

Die Eigenschaft "Index" referenziert ein Spaltenpaar. "Precision" legt die Anzahl der Nachkommastellen fest, die in dieser Wertespalte angezeigt werden sollen. Es können maximal 16 Nachkommastellen angezeigt werden.

WinCC Online Trend Control

Legt die Anzahl der Nachkommastellen fest, mit denen die Angabe der Skalierungswerte erfolgt.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

PrecisionX-Eigenschaft

Beschreibung

Legt die Anzahl der Nachkommastellen fest, mit denen die Angabe der Skalierungswerte für die X-Achse erfolgt, oder gibt sie zurück. Schreib-Lese-Zugriff.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

PrecisionY-Eigenschaft

Beschreibung

Legt die Anzahl der Nachkommastellen fest, mit denen die Angabe der Skalierungswerte für die Y-Achse erfolgt, oder gibt sie zurück. Schreib-Lese-Zugriff.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

PredefinedAngles-Eigenschaft

Beschreibung

Legt die Tiefendarstellung beim Objekt 3DBarGraph fest oder gibt sie zurück. Wertebereich von 0 bis 3.

0 = Kavalier

1 = Isometrisch

2 = Axonometrisch

3 = frei definiert

Siehe auch

ScreenItem-Objekt (Seite 134)

3D-Balken (Seite 181)

Pressed-Eigenschaft

Beschreibung

TRUE, wenn das Objekt Button oder RoundButton gedrückt ist. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

PrintBackgroundColor-Eigenschaft

Beschreibung

TRUE, wenn beim Ausdruck des Controls die festgelegte Hintergrundfarbe mit ausgedruckt wird. BOOLEAN Schreib-Lese-Zugriff.

PrintJob-Eigenschaft

Beschreibung

Legt fest oder liest aus, welches Printlayout für die Druckausgabe verwendet wird.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

PrintJobName-Eigenschaft

Druckauftrag aktuelle Ansicht - PrintJobName

Legt den Druckauftrag fest, den die Druckfunktion der Schaltfläche "Drucken" in der Symbolleiste auslöst. Der empfohlene Druckauftrag für das Control ist voreingestellt.

Über die Auswahlschaltfläche öffnen Sie den Dialog "Druckauftrag auswählen" zur Auswahl eines Druckauftrags.

Das Attribut ist mit dem Namen **PrintJobName** dynamisierbar. Der Datentyp ist STRING.

Process-Eigenschaft

Beschreibung

Legt die Voreinstellung für den anzuzeigenden Wert fest oder gibt sie zurück.
In Runtime wird dieser Wert verwendet, wenn beim Start des Bildes die zugehörige Variable nicht angeschlossen oder nicht aktualisiert ist.

Siehe auch

Slider (Seite 221)

Radio-Box (Seite 217)

Check-Box (Seite 215)

Balken (Seite 186)

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

ProcessValue-Eigenschaft

Beschreibung

Gibt ein Objekt vom Typ "ProcessValue" zurück.

Siehe auch

Alarms-Objekt (Auflistung) (Seite 120)

ProjectPath-Eigenschaft

Beschreibung

Enthält Pfad und Namen des zugehörigen Projektes.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

ProviderClsid-Eigenschaft

Beschreibung

"ProviderClsid" legt fest, ob die Kurven des Kurven-Control entweder mit Archivvariablen oder mit Online-Variablen versorgt wird.

- {416A09D2-8B5A-11D2-8B81-006097A45D48}: Die Kurven sind mit Archivvariablen verbunden.
- {A3F69593-8AB0-11D2-A440-00A0C9DBB64E}: Die Kurven sind mit Online-Variablen verbunden.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

ProviderType-Eigenschaft

Beschreibung

Legt fest, welche Art von Werten in einer über "Index" referenzierten Kurve dargestellt werden sollen. Bei der Änderung von "ProviderType" können unzulässige Kombinationen mit anderen Attributen zur Datenanbindung entstehen. Deshalb muss vor einer Änderung von "ProviderType" die sofortige Übernahme der Änderung mit "FreezeProviderConnections" verhindert werden.

0: Werteversorgung erfolgt über die API-Schnittstelle.

-1: Darstellung von Online-Variablen oder Archiv-Variablen

-2: Darstellung von Werten aus einem Anwenderarchiv

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

1.14.4.16 Q

QualityCode-Eigenschaft

Beschreibung

Stellt ein Maß für die Qualität des Variablenwertes nach dem Lesen dar. Der QualityCode wird als 16-Bit-Wert zum selbstständigen Auswerten angeboten. Nach dem Schreiben einer Variable ist der Wert ungültig.

SHORT (readonly)

Hinweis

Eine Zusammenfassung der möglichen QualityCodes finden Sie im WinCC Information-System unter dem Stichwort "Kommunikation" > "Diagnose" oder "Kommunikation" > "Quality Codes".

Beispiel

Das folgende Beispiel gibt die Qualität des gelesenen Wertes aus, wenn beim Lesen kein Fehler aufgetreten ist:

```
'VBS83
Dim objTag
Dim lngLastErr
Set objTag = HMIRuntime.Tags("Tag1")
objTag.Read
lngLastErr = objTag.LastError
If 0 = lngLastErr Then
MsgBox objTag.QualityCode
End If
```

Siehe auch

LastError-Eigenschaft (Seite 435)

ErrorDescription-Eigenschaft (Seite 389)

Tag-Objekt (Seite 146)

1.14.4.17 R

Ra - Ri

Radius-Eigenschaft

Beschreibung

Legt den Radius in Pixel fest oder gibt ihn zurück.

Siehe auch

Kreissegment (Seite 164)

Kreisbogen (Seite 162)

Kreis (Seite 160)

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

RadiusHeight-Eigenschaft

Beschreibung

Legt den vertikalen Radius in Pixel (0 bis 5000) fest oder gibt ihn zurück.

Siehe auch

Ellipsensegment (Seite 158)
Ellipsenbogen (Seite 156)
Ellipse (Seite 153)
ScreenItem-Objekt (Seite 134)

RadiusWidth-Eigenschaft

Beschreibung

Legt den horizontalen Radius in Pixel (0 bis 5000) fest oder gibt ihn zurück.

Siehe auch

Ellipsensegment (Seite 158)
Ellipsenbogen (Seite 156)
Ellipse (Seite 153)
ScreenItem-Objekt (Seite 134)

RangeMax-Eigenschaft

Beschreibung

Legt den maximalen Absolutwert der Wertanzeige fest.
Hat die Eigenschaft "WithLabels" den Wert -1 (ja), so wird dieser Wert an der Skala angezeigt.

Siehe auch

WinCC Slider Control (Seite 278)
ScreenItem-Objekt (Seite 134)

RangeMin-Eigenschaft

Beschreibung

Legt den minimalen Absolutwert der Wertanzeige fest.

Hat die Eigenschaft "WithLabels" den Wert -1 (ja), so wird dieser Wert an der Skala angezeigt.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

Rectangular-Eigenschaft

Beschreibung

Legt das Seitenverhältnis des rechteckigen Hintergrunds der Gauge fest oder gibt es zurück. BOOLEAN Schreib-Lese-Zugriff.

FALSE: Die Größe der Gauge lässt sich mit der Maus durch Ziehen an den Markierungspunkten in jedem gewünschten Seitenverhältnis einstellen.

TRUE: Die Größe der Gauge lässt sich mit der Maus nur an den Markierungseckpunkten durch Ziehen einstellen. Das Seitenverhältnis des Hintergrunds bleibt dabei immer 1:1.

Siehe auch

ScreenItem-Objekt (Seite 134)

WinCC Gauge Control (Seite 261)

ReferenceRotationLeft-Eigenschaft

Beschreibung

Legt die x-Koordinate des Referenzpunktes fest, um den das Objekt in Runtime gedreht werden soll oder gibt sie zurück.
Der Wert der x-Koordinate ist relativ zur Objektbreite. Geben Sie den Wert ausgehend von der linken Kante des objektumfassenden Rechteckes in Prozent an.

Siehe auch

Linie (Seite 166)

Polygonzug (Seite 170)

Polygon (Seite 168)

ScreenItem-Objekt (Seite 134)

ReferenceRotationTop-Eigenschaft

Beschreibung

Legt die y-Koordinate des Referenzpunktes fest, um den das Objekt in Runtime gedreht werden soll oder gibt sie zurück.

Der Wert der y-Koordinate ist relativ zur Objekthöhe. Geben Sie den Wert ausgehend von der oberen Kante des objektumfassenden Rechteckes in Prozent an.

Siehe auch

ScreenItem-Objekt (Seite 134)

Linie (Seite 166)

Polygonzug (Seite 170)

Polygon (Seite 168)

RelayCurves-Eigenschaft

Beschreibung

TRUE, wenn die Kurven gestaffelt dargestellt werden. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

Relevant-Eigenschaft

Beschreibung

TRUE, wenn das Objekt für die Bildung der Sammelanzeige berücksichtigt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

Replacement-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. Werte, deren Anfangswert nach dem Aktivieren des Runtime nicht bekannt ist, oder für die ein Ersatzwert verwendet wird, besitzen einen unsicheren Status. "Replacement" legt fest, ob derartige Werte mit der in "ReplacementColor" festgelegten Farbe gekennzeichnet werden sollen. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

ReplacementColor-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. Werte, deren Anfangswert nach dem Aktivieren des Runtime nicht bekannt ist, oder für die ein Ersatzwert verwendet wird, besitzen einen unsicheren Status. "ReplacementColor" legt die Farbe fest, die für die Kennzeichnung dieser Werte verwendet wird. Die Angabe der Farbe erfolgt als RGB-Wert. Ob die Angabe ausgewertet wird ist abhängig von der Eigenschaft "Replacement".

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

RightComma-Eigenschaft

Beschreibung

Legt die Anzahl der Nachkommastellen (0 bis 20) fest oder gibt sie zurück.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

Ro - Ru

Rotation-Eigenschaft

Drehung (Rotation)

Legt die Drehung entgegen dem Uhrzeigersinn um den Mittelpunkt des Symbols fest.

Folgende Einstellungen sind möglich:

Wert	Erläuterung
0	Das Symbol wird nicht gedreht.
90	Das Symbol wird um 90 Grad gedreht.
180	Das Symbol wird um 180 Grad gedreht.
270	Das Symbol wird um 270 Grad gedreht.

Das Attribut ist mit dem Namen **Rotation** dynamisierbar. Der Datentyp ist LONG.

RotationAngle-Eigenschaft

Beschreibung

Standard-Objekte

Legt den Rotationswinkel in Grad fest oder gibt ihn zurück.

Das Objekt wird in Runtime (ausgehend von der projektierten Ausgangsstellung) im Uhrzeigersinn um den eingegebenen Wert um den Referenzpunkt gedreht dargestellt. Die geänderte Orientierung des Objekts ist nur in Runtime sichtbar.

Die Koordinaten des Referenzpunkts sind mit den Attributen "Rotationsreferenz X" und "Rotationsreferenz Y" definiert.

T-Stück

Definiert die Orientierung eines T-Stücks in Grad oder gibt sie zurück.

Das Attribut kann vier Werte annehmen. Wenn Sie einen anderen Wert eingeben, wird dieser automatisch zum Modul 360 umgerechnet und auf den nächstliegenden zulässigen Wert auf- oder abgerundet.

Die Orientierung ergibt sich aus einer Drehung des T-Stücks um den Mittelpunkt um die angegebene Gradzahl in Uhrzeigerrichtung.

0 Die Standardlage des T-Stücks in der Gestalt des Buchstabens "T"
90 Das "Bein" des "T"s weist nach links
180 Das "Bein" des "T"s weist nach oben
270 Das "Bein" des "T"s weist nach rechts

Siehe auch

Linie (Seite 166)

Polygonzug (Seite 170)

Polygon (Seite 168)

ScreenItem-Objekt (Seite 134)

RoundCornerHeight-Eigenschaft

Beschreibung

Legt den Eckradius fest oder gibt ihn zurück.
Geben Sie den Wert prozentual zur halben Höhe des Objektes ein.

Siehe auch

Rundrechteck (Seite 175)

ScreenItem-Objekt (Seite 134)

RoundCornerWidth-Eigenschaft

Beschreibung

Legt den Eckradius fest oder gibt ihn zurück.
Geben Sie den Wert prozentual zur halben Breite des Objektes ein.

Siehe auch

ScreenItem-Objekt (Seite 134)

RowCellCount-Eigenschaft

RowCellCount

Gibt die Anzahl der Zellen des Row-Objekts eines Tabellen-Controls an. Die Anzahl der Zellen entspricht der Anzahl der Spalten.

RowCellText-Eigenschaft

RowCellText

Gibt den Inhalt einer Zelle als String zurück. Die Zelle wird ermittelt aus der Spaltennummer des Row-Objekts. Die Nummerierung läuft von "1" bis "CellCount".

RowCount-Eigenschaft

RowCount

Gibt die Anzahl der Zeilen des Row-Objekts eines Tabellen-Controls an.

RowNumber-Eigenschaft

RowNumber

Gibt die Zeilennummer des Row-Objekts eines Tabellen-Controls an.

RowScrollbar-Eigenschaft

Zeilen-Rollbalken - RowScrollbar

Legt fest, ob Zeilen-Rollbalken angezeigt werden.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	nein	Die Zeilen-Rollbalken werden nicht angezeigt.
1	bei Bedarf	Die Zeilen-Rollbalken werden angezeigt, wenn der Platzbedarf des Controls in horizontaler Richtung größer ist als der zur Verfügung stehende Anzeigebereich.
2	immer	Die Zeilen-Rollbalken werden immer angezeigt.

Das Attribut ist mit dem Namen **RowScrollbar** dynamisierbar. Der Datentyp ist LONG.

RowTitleAlign-Eigenschaft

Ausrichtung Zeilenbeschriftung - RowTitleAlign

Legt fest, wie die Zeilenüberschriften ausgerichtet werden.

Folgende Einstellungen stehen zur Auswahl:

Wert	Beschreibung	Erklärung
0	links	Die Zeilenüberschriften werden linksbündig angezeigt.
1	zentriert	Die Zeilenüberschriften werden zentriert angezeigt.
2	rechts	Die Zeilenüberschriften werden rechtsbündig angezeigt.

Das Attribut ist mit dem Namen **RowTitleAlign**dynamisierbar. Der Datentyp ist LONG.

RowTitles-Eigenschaft

Anzeigen Zeilenbeschriftung - RowTitles

Legt fest, ob die Zeilenbeschriftungen angezeigt werden.

Wert	Erklärung
TRUE	Die Zeilenbeschriftungen werden angezeigt.
FALSE	Die Zeilenbeschriftungen werden nicht angezeigt.

Das Attribut ist mit dem Namen **RowTitles** dynamisierbar. Der Datentyp ist BOOLEAN.

RTPersistence-Eigenschaft

Online-Projektierung beim nächsten Bildwechsel - RTPersistence

Legt fest, ob Online-Projektierungen des Controls auch nach einem Bildwechsel beibehalten werden.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	verwerfen	Die aktuellen Online-Projektierungen werden beim nächsten Bildwechsel verworfen.
1	beibehalten	Die aktuellen Online-Projektierungen werden beim nächsten Bildwechsel beibehalten.
2	zurücksetzen	Alle jemals vorgenommenen Online-Projektierungen gehen verloren. Das Bild wird auf den im Konfigurationssystem vorhandenen Inhalt gesetzt.

Das Attribut ist mit dem Namen **RTPersistence** dynamisierbar. Der Datentyp ist LONG.

RTPersistencePasswordLevel-Eigenschaft

Bedienberechtigung zur Online-Projektierung - RTPersistencePasswordLevel

Zeigt die Berechtigung für die Online-Projektierung. Über die Auswahlschaltfläche können Sie die Berechtigung ändern. Berechtigungen werden im Editor "User Administrator" projektiert.

Das Attribut ist mit dem Namen **RTPersistencePasswordLevel** dynamisierbar. Der Datentyp ist LONG.

RTPersistenceType-Eigenschaft

Online-Projektierung - RTPersistenceType

Legt fest, wie Online-Projektierungen von WinCC beibehalten werden.

Folgende Einstellungen sind möglich:

Wert	Beschreibung	Erklärung
0	nicht beibehalten	Online-Projektierungen werden nicht beibehalten. Sie gehen beim nächsten Bildwechsel verloren.
1	während Runtime beibehalten	Online-Projektierungen werden während Runtime beibehalten. Sie gehen beim Beenden verloren.
2	permanent beibehalten	Online-Projektierungen werden permanent beibehalten. Sie sind auch nach einem Neustart verfügbar.

Das Attribut ist nicht dynamisierbar.

RulerFont-Eigenschaft

Beschreibung

Dieses Attribut legt die Schriftart der Tabelle der Variablenwerte fest, die durch die Tastenfunktion "Anzeige Wert an dieser Stelle" / "Lineal" angezeigt werden. Schreib-Lese-Zugriff.

RulerPrecisions-Eigenschaft

Beschreibung

Legt die Anzahl der Nachkommastellen fest, mit denen ein Messwert angezeigt wird, wenn er über die Funktion "Anzeige Wert an dieser Stelle" ermittelt wird.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

RulerPrecisionX-Eigenschaft

Beschreibung

Legt die Anzahl der Nachkommastellen fest, die von der Funktion "Anzeige Wert an dieser Stelle" zur Anzeige der X-Koordinate eines Messwertes verwendet werden. Ob die Angabe ausgewertet wird, ist abhängig vom Wert des Attributes "TimeAxisX".

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

RulerPrecisionY-Eigenschaft

Beschreibung

Legt die Anzahl der Nachkommastellen fest, die von der Funktion "Anzeige Wert an dieser Stelle" zur Anzeige der Y-Koordinate eines Messwertes verwendet werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

RulerType-Eigenschaft

Fenster - RulerType

Legt fest, welches Fenster in Runtime angezeigt wird. Je nach Fenstertyp sind nur bestimmte Blöcke als Spalten des WinCC RulerControls nutzbar.

Folgende Fenstertypen stehen zur Auswahl:

Wert	Beschreibung	Erklärung
0	Fenster "Lineal"	Das Linealfenster zeigt die Koordinatenwerte der Kurven am Lineal oder Werte einer selektierten Zeile in der Tabelle.
1	Fenster "Statistikbereich"	Das Statistikbereich-Fenster zeigt die Werte der Untergrenze und Obergrenze der Kurven zwischen zwei Linealen oder des selektierten Bereichs in der Tabelle.
2	Fenster "Statistik"	Das Statistikfenster zeigt die statistische Auswertung der Kurven zwischen zwei Linealen oder der selektierten Werte in der Tabelle.

Das Attribut ist mit dem Namen **RulerType** dynamisierbar. Der Datentyp ist LONG.

1.14.4.18 S

Sa - Sc

SameSize-Eigenschaft

Beschreibung

TRUE, wenn alle vier Buttons beim Objekt GroupDisplay die gleiche Größe haben. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

SavedTrend-Eigenschaft

Beschreibung

Zeigt den Namen der zuletzt gespeicherten Kurven an, die im WinCC Online Trend Control über die Schaltfläche "Report speichern" exportiert wurden. Nur Lese-Zugriff.

ScaleColor-Eigenschaft

Beschreibung

Legt die Farbe der Skala fest oder gibt sie zurück. LONG Schreib Lese-Zugriff.
Damit die Farbe angezeigt wird, muss die Eigenschaft "Scaling" auf "True" gesetzt sein.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ScaleTicks-Eigenschaft

Beschreibung

Legt die Anzahl der Segmente fest, in die der Balken durch die großen Teilstriche der Skala unterteilt wird:
0-100: Objekt kann maximal in 100 Segmente unterteilt werden
= 0: Die optimale Anzahl der Segmente wird automatisch festgelegt

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

Scaling-Eigenschaft

Beschreibung

TRUE, wenn zusätzlich eine Skala zur Darstellung der Werte verwendet wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ScalingType-Eigenschaft

Beschreibung Balkenskalierung

Legt die Art der Balkenskalierung fest oder gibt sie zurück. Wertebereich von 0 bis 6.

0 = linear

1 = logarithmisch

2 = negativ logarithmisch

3 = automatisch (linear)

4 = tangens

5 = quadratisch

6 = kubisch

Damit die Farbe angezeigt wird, muss die Eigenschaft "Scaling" auf "True" gesetzt sein.

Beschreibung Online Trend Control

Legt die Art der Skalierung der über "Index" referenzierten Kurve fest oder gibt sie zurück. Wertebereich von 0 bis 2.

0 = linear

1 = logarithmisch

2 = negativ logarithmisch

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ScalingTypeX-Eigenschaft

Beschreibung

Legt die Art der Skalierung der X-Achse einer über "Index" referenzierten Kurve fest. Ob die Angabe ausgewertet wird, ist abhängig vom Wert des Attributes "TimeAxisX".

0: Linear

-1: Logarithmisch. Bei dieser Einstellung können keine negativen Werte angezeigt werden.

-2: Logarithmisch negiert. Bei dieser Einstellung können keine positiven Werte angezeigt werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

ScalingTypeY-Eigenschaft

Beschreibung

Legt die Art der Skalierung der Y-Achse einer über "Index" referenzierten Kurve fest.

0: Linear

-1: Logarithmisch. Bei dieser Einstellung können keine negativen Werte angezeigt werden.

-2: Logarithmisch negiert. Bei dieser Einstellung können keine positiven Werte angezeigt werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

ScreenName-Eigenschaft

Beschreibung

Legt das Bild fest, das im Bildfenster in Runtime angezeigt wird oder gibt den Bildnamen zurück.

Hinweis

Bildnamen sind aus Kompatibilitätsgründen zu zukünftigen Versionen immer ohne die Dateierweiterung ".PDL" zu schreiben.

Siehe auch

Bildfenster (Seite 190)

ScreenItem-Objekt (Seite 134)

Screens-Eigenschaft

Beschreibung

Liefert ein Objekt vom Typ "Screens" zurück.

Screens (read-only)

Beispiel

Das folgende Beispiel greift auf das Bild "NewPDL1" zu:

```
'VBS84
Dim objScreen
Set objScreen = HMIRuntime.Screens("NewPDL1")
```

Siehe auch

Screens-Objekt (Auflistung) (Seite 143)

Screen-Objekt (Seite 140)

HMIRuntime-Objekt (Seite 127)

ScreenItems-Eigenschaft

Beschreibung

Liefert ein Objekt vom Typ "ScreenItems" zurück.

ScreenItems (readonly)

Beispiel

Das folgende Beispiel gibt die Anzahl der im Bild "NewPDL1" enthaltenen Objekte aus:

```
'VBS85
Dim objScreen
Set objScreen = HMIRuntime.Screens("NewPDL1")
Msgbox objScreen.ScreenItems.Count
```

Siehe auch

ScreenItems-Objekt (Auflistung) (Seite 138)

HMIRuntime-Objekt (Seite 127)

ScrollBars-Eigenschaft

Beschreibung

TRUE, wenn das Objekt in Runtime Bildlaufleisten hat. Nur Lese-Zugriff.

Siehe auch

Bildfenster (Seite 190)

ScreenItem-Objekt (Seite 134)

ScrollPositionX-Eigenschaft

Beschreibung

Legt die horizontale Verschiebung der Bildlaufleiste in einem Bildfenster mit Rollbalken fest oder gibt den Wert zurück.

Das Bild wird im Bildfenster mit der horizontalen bzw. vertikalen Verschiebung der Bildlaufleisten dargestellt. Wenn Sie das Bild ausgeschnitten darstellen wollen, in der die Bildlaufleisten sich am linken und oberen Rand des Bildes befinden, verwenden Sie für den Ursprung dieses Ausschnitts die Eigenschaften "OffsetLeft" und "OffsetTop".

Siehe auch

ScreenItem-Objekt (Seite 134)

OffsetTop-Eigenschaft (Seite 493)

OffsetLeft-Eigenschaft (Seite 493)

Bildfenster (Seite 190)

ScrollPositionY-Eigenschaft

Beschreibung

Legt die vertikale Verschiebung der Bildlaufleiste in einem Bildfenster mit Rollbalken fest oder gibt den Wert zurück.

Das Bild wird im Bildfenster mit der horizontalen bzw. vertikalen Verschiebung der Bildlaufleisten dargestellt. Wenn Sie das Bild ausgeschnitten darstellen wollen, in der die Bildlaufleisten sich am linken und oberen Rand des Bildes befinden, verwenden Sie für den Ursprung dieses Ausschnitts die Eigenschaften "OffsetLeft" und "OffsetTop".

Siehe auch

OffsetTop-Eigenschaft (Seite 493)

OffsetLeft-Eigenschaft (Seite 493)

Bildfenster (Seite 190)

ScreenItem-Objekt (Seite 134)

Se

SecondNeedleHeight-Eigenschaft

Beschreibung

Legt die Länge des Sekundenzeigers der Analoguhr fest oder gibt sie zurück. Die Angabe der Länge erfolgt in Prozent, bezogen auf die halbe Länge der kürzeren Seite des rechteckigen Hintergrunds. Schreib-Lese-Zugriff.

Beispiel:
Die kürzere Seite des rechteckigen Hintergrunds sei 100 Pixel lang.
Die Sekundenzeigerlänge sei mit 80 angegeben.
Daraus ergibt sich die Länge des Sekundenzeigers zu (100 Pixel / 2) * 0,8 = 40 Pixel.

Siehe auch

ScreenItem-Objekt (Seite 134)

WinCC Digital Analog Clock (Seite 255)

SecondNeedleWidth-Eigenschaft

Beschreibung

Legt die Breite des Sekundenzeigers der Analoguhr fest oder gibt sie zurück. Die Angabe der Breite erfolgt in Prozent, bezogen auf die doppelte Länge des Sekundenzeigers. Schreib-Lese-Zugriff.

Beispiel:
Die Länge des Sekundenzeigers sei 40 Pixel.
Die Sekundenzeigerbreite sei mit 2 angegeben.
Daraus ergibt sich die Breite des Sekundenzeigers zu 40 Pixel * 2 * 0,02 = 2 Pixel (gerundet).

Siehe auch

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

SelBGColor-Eigenschaft

Beschreibung

Legt die Hintergrundfarbe für den ausgewählten Eintrag beim Objekt Textliste fest oder gibt ihn zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Textliste (Seite 208)

ScreenItem-Objekt (Seite 134)

SelectArchiveName-Eigenschaft

SelectArchiveName

Öffnet den Dialog zur Auswahl des Anwenderarchivs.

Das Attribut können Programmierer nutzen, um z. B. über eine Schaltfläche den Benutzer ein Anwenderarchiv auswählen zu lassen.

Das Attribut ist mit dem Namen **SelectArchiveName** dynamisierbar. Der Datentyp ist BOOLEAN.

SelectedCellColor-Eigenschaft

Hintergrundfarbe markierte Zelle - SelectedCellColor

Gibt die Hintergrundfarbe der markierten Zelle an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **SelectedCellColor** dynamisierbar. Der Datentyp ist LONG.

SelectedCellForeColor-Eigenschaft

Schriftfarbe markierte Zelle - SelectedCellForeColor

Gibt die Schriftfarbe der markierten Zelle an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **SelectedCellForeColor** dynamisierbar. Der Datentyp ist LONG.

SelectedRowColor-Eigenschaft

Hintergrundfarbe markierte Zeile - SelectedRowColor

Gibt die Hintergrundfarbe der markierten Zeile an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **SelectedRowColor** dynamisierbar. Der Datentyp ist LONG.

SelectedRowForeColor-Eigenschaft

Schriftfarbe markierte Zeile - SelectedRowForeColor

Gibt die Schriftfarbe der markierten Zeile an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **SelectedRowForeColor** dynamisierbar. Der Datentyp ist LONG.

SelectedTitleColor-Eigenschaft

Hintergrund Markierungsfarbe - SelectedTitleColor

Gibt die Hintergrundfarbe der markierten Tabellenüberschrift an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Einstellung ist in Runtime nur wirksam, wenn die Option "Markierungsfarbe" bzw. "UseSelectedTitleColor" aktiviert ist.

Das Attribut ist mit dem Namen **SelectedTitleColor** dynamisierbar. Der Datentyp ist LONG.

SelectedTitleForeColor-Eigenschaft

Schrift Markierungsfarbe - SelectedTitleForeColor

Gibt die Schriftfarbe der markierten Tabellenüberschrift an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Einstellung ist in Runtime nur wirksam, wenn die Option "Markierungsfarbe" bzw. "UseSelectedTitleColor" aktiviert ist.

Das Attribut ist mit dem Namen **SelectedTitleForeColor** dynamisierbar. Der Datentyp ist LONG.

SelectedTrend-Eigenschaft

Beschreibung

Mit dieser Eigenschaft wird eine Kurve über ihren Namen in den Vordergrund gebracht. Schreib-Lese-Zugriff.

SelectionColoring-Eigenschaft

Markierungsfarben für - SelectionColoring

Legt fest, ob Markierungsfarben für Zelle oder Zeile verwendet werden.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Keine	Keine Markierungsfarben für Zelle und Zeile.
1	Zelle	Markierungsfarbe für Zelle.
2	Zeile	Markierungsfarbe für Zeile.
3	Zelle und Zeile	Markierungsfarben für Zelle und Zeile.

Das Attribut ist mit dem Namen **SelectionColoring** dynamisierbar. Der Datentyp ist LONG.

SelectionMode-Eigenschaft

Beschreibung

Legt fest, ob und wie eine Meldezeile selektiert werden kann.

- 0 - NoSelection: Verhindert die Selektion einer Meldung. Eine Quittierung wirkt sich immer auf die älteste anstehende Meldung aus.
- 1 - Cell: Ermöglicht die Selektion von Feldern in der Meldezeile. Eine Quittierung wirkt sich immer auf die selektierte Meldung aus.
- 2 - Line: Ermöglicht die Selektion einer Meldezeile. Eine Quittierung wirkt sich immer auf die selektierte Meldung aus.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

SelectionRect-Eigenschaft

Auswahlrahmen - SelectionRect

Legt fest, ob ein Auswahlrahmen für die markierten Zellen bzw. Zeilen verwendet wird.

Folgende Einstellungen sind möglich:

Wert	Beschreibung	Erklärung
0	Kein	Für markierte Zellen und Zeilen wird kein Auswahlrahmen verwendet.
1	Zelle	Für die markierte Zelle wird ein Auswahlrahmen verwendet.
2	Zeile	Für die markierte Zeile wird ein Auswahlrahmen verwendet.

Das Attribut ist mit dem Namen **SelectionRect** dynamisierbar. Der Datentyp ist LONG.

SelectionRectColor-Eigenschaft (vor WinCC V7)

Beschreibung

Legt die Farbe des Auswahlrechtecks im Meldefenster fest, wenn der SelectionType "1" ist.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

SelectionRectColor-Eigenschaft

Farbe Auswahlrechteck - SelectionRectColor

Gibt die Linienfarbe des Auswahlrahmens an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **SelectionRectColor** dynamisierbar. Der Datentyp ist LONG.

SelectionRectWidth-Eigenschaft (vor WinCC V7)

Beschreibung

Legt die Linienstärke des Auswahlrechtecks im Meldefenster fest, wenn der SelectionType "1" ist.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

SelectionRectWidth-Eigenschaft

Stärke Auswahlrechteck - SelectionRectWidth

Legt die Linienstärke des Auswahlrahmens in Pixel fest.

Das Attribut ist mit dem Namen **SelectionRectWidth** dynamisierbar. Der Datentyp ist LONG.

SelectionType-Eigenschaft (vor WinCC V7)

Beschreibung

Legt fest, ob die selektierte Meldung im Meldefenster durch Farbwechsel oder durch ein Auswahlrechteck optisch hervorgehoben wird.

- 0 - Farbwechsel: die selektierte Meldung wird durch Farbwechsel optisch hervorgehoben
- 1 - Auswahlrechteck: die selektierte Meldung wird durch ein Auswahlrechteck optisch hervorgehoben

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

SelectionType-Eigenschaft

Markierbare Zeilen - SelectionType

Legt fest, wie viele Zeilen Sie markieren können. Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Keine	Keine Zeile wird markiert.
1	Einzelselektion	Eine Zeile kann markiert werden.
2	Mehrfachselektion	Mehrere Zeilen können markiert werden.

Das Attribut ist mit dem Namen **SelectionType** dynamisierbar. Der Datentyp ist LONG.

SelIndex-Eigenschaft

Beschreibung

Legt den Index fest und gibt ihn zurück, dessen zugehöriger Text im Kombinationsfeld oder Listenfeld hervorgehoben angezeigt wird.

Der Maximalwert ist die Anzahl der Zeilen (NumberLines) des Objekts.

SelText-Eigenschaft

Beschreibung

Zeigt den mit dem Attribut "Selektiertes Feld" (SelIndex) festgelegten Text, der im Kombinationsfeld oder Listenfeld hervorgehoben dargestellt wird.

SelTextColor-Eigenschaft

Beschreibung

Legt die Textfarbe des ausgewählten Eintrages des Objektes Textliste fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Textliste (Seite 208)

ScreenItem-Objekt (Seite 134)

ServerData-Eigenschaft

Beschreibung

Das Attribut kann nur über den Dialog "Eigenschaften von WinCC Online Trend Control" verändert werden. Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

ServerDataX

ServerDataX

Greift beim WinCC FunctionTrendControl auf die projektierte Datenanbindung für die X-Achse zu.

Das Attribut ist mit dem Namen **ServerDataX** dynamisierbar. Der Datentyp ist LONG.

Beispiel: Start-ID verändern

Es besteht die Möglichkeit, mit dem Attribut **ServerDataX** die Start-ID der X-Achse zu verändern.

Voraussetzung ist, dass eine Kurve existiert, ein Kurvenfenster angelegt ist, die X- und Y-Achsen projektiert sind und eine Datenanbindung zum Benutzerarchiv besteht.

Im folgenden Beispiel referenzieren Sie mit der GetTrend-Methode im ersten Schritt auf das Objekt, im zweiten Schritt auf die verwendete Kurve. Im dritten Schritt ermitteln Sie die Datenanbindungeinstellungen. Im vierten Schritt wird die Start-ID auf 4 gesetzt. Die (3) steht für den Auflistungstyp "Anwenderarchiv" bei der Datenversorgung. Im fünften Schritt wird die geänderte Datenanbindungeinstellungen geändert:

```
Sub OnCklick(ByVal Item)
```

1. Schritt:
```
Dim fx_ctrl
Set fx_ctrl  ScreenItems.Item("Control1")
```
2. Schritt:
```
Dim fx_trend
Set fx_trend = fx_ctrl.Getrend("myTrend1")
```
3. Schritt:
```
Dim vServerDataX, vServerDataY
vServerDataX = fx_trend.ServerDataX
vServerDataY = fx_trend.ServerDataY
```
4. Schritt
```
Dim startId
startId = CLng(4)
vServerDataX(3) = startId
vServerDataY(3) = startId
```
5. Schritt:
```
fx_trend.ServerDataX = ServerDataX
fx_trend.ServerDataY = ServerDataY
```

```
End Sub
```

ServerDataY

ServerDataY

Greift beim WinCC FunctionTrendControl auf die projektierte Datenanbindung für die Y-Achse zu.

Das Attribut ist mit dem Namen **ServerDataY** dynamisierbar. Der Datentyp ist LONG.

Beispiel: Start-ID verändern

Es besteht die Möglichkeit, mit dem Attribut **ServerDataY** die Start-ID der Y-Achse zu verändern.

Voraussetzung ist, dass eine Kurve existiert, ein Kurvenfenster angelegt ist, die X- und Y-Achsen projektiert sind und eine Datenanbindung zum Benutzerarchiv besteht.

Im folgenden Beispiel referenzieren Sie mit der GetTrend-Methode im ersten Schritt auf das Objekt, im zweiten Schritt auf die verwendete Kurve. Im dritten Schritt ermitteln Sie die Datenanbindungeinstellungen. Im vierten Schritt wird die Start-ID auf 4 gesetzt. Die (3) steht für den Auflistungstyp "Anwenderarchiv" bei der Datenversorgung. Im fünften Schritt wird die geänderte Datenanbindungeinstellungen geändert:

```
Sub OnCklick(ByVal Item)
```

1. Schritt:
```
Dim fx_ctrlSet fx_ctrl  ScreenItems.Item("Control1")
```
2. Schritt:
```
Dim fx_trendSet fx_trend = fx_ctrl.Getrend("myTrend1")
```

3. Schritt:
```
Dim vServerDataX, vServerDataYvServerDataX =
fx_trend.ServerDataXvServerDataY = fx_trend.ServerDataY
```
4. Schritt:
```
Dim startIdstartId = CLng(4)vServerDataX(3) =
startIdvServerDataY(3) = startId
```
5. Schritt:
```
fx_trend.ServerDataX = ServerDataXfx_trend.ServerDataY =
ServerDataY
```

```
End Sub
```

ServerNames-Eigenschaft

Serverauswahl - ServerNames

Legt fest, von welchen Servern in einem verteilten System das Meldefenster die darzustellenden Daten bezieht.

Das Attribut ist mit dem Namen **ServerNames** dynamisierbar. Der Datentyp ist STRING.

ServerNames-Eigenschaft (vor WinCC V7)

Beschreibung

Legt die Server eines Verteilten Systems fest, von denen das Meldefenster Daten beziehen soll. Die Angabe erfolgt in der Form: NameServer1;NameServer2;NameServer3. Schreib-Lese-Zugriff.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

ServerPrefix-Eigenschaft

Beschreibung

Legt fest, auf welchem Server das Bild liegt, das in Runtime im Bildfenster angezeigt wird, oder gibt den Servernamen zurück.
Geben Sie den Servernamen gefolgt von zwei Doppelpunkten an: "<Servername>::". Es wird nicht überpüft, ob der Server tatsächlich vorhanden ist.

Siehe auch

Bildfenster (Seite 190)

ScreenItem-Objekt (Seite 134)

Sh - Sk

ShareSpaceWithSourceControl-Eigenschaft

ShareSpaceWithSourceControl

Legt fest, ob die Größe des Quell-Controls im Bildfenster angepasst wird, damit das WinCC RulerControl auch in einem kleinen Bildfenster angezeigt wird.

Wert	Erklärung
TRUE	Das Quell-Control im Bildfenster wird angepasst.
FALSE	Das Quell-Control im Bildfenster wird nicht angepasst.

Das Attribut ist mit dem Namen **ShareSpaceWithSourceControl** dynamisierbar. Der Datentyp ist BOOLEAN.

ShowBar-Eigenschaft

Beschreibung

TRUE, wenn der Balken angezeigt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

ShowDanger-Eigenschaft

Beschreibung

Steuert die Anzeige des "Gefahrenbereichs" auf der Instrumentenskala. BOOLEAN Schreib-Lese-Zugriff.

TRUE: Der Bereich wird mit der durch "DangerColor" festgelegten Farbe gekennzeichnet.

FALSE: Die farbliche Kennzeichnung des Bereichs ist ausgeschaltet.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

ShowDecimalPoint-Eigenschaft

Beschreibung

TRUE, wenn die Beschriftung der Skalenteilung mit Dezimalzahlen (Dezimalkomma und eine Dezimalstelle) erfolgt.

FALSE, wenn die Beschriftung der Skalenteilung mit ganzen Zahlen erfolgt.

BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

ShowNormal-Eigenschaft

Beschreibung

Steuert die Anzeige des "Normalbereichs" auf der Instrumentenskala. BOOLEAN Schreib-Lese-Zugriff.

TRUE: Der Bereich wird mit der durch Normalfarbe festgelegten Farbe gekennzeichnet.

FALSE: Die farbliche Kennzeichnung des Bereichs ist ausgeschaltet.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

ShowPeak-Eigenschaft

Beschreibung

Bestimmt die Anzeige eines Schleppzeigers zur Anzeige des Maximalwerts. BOOLEAN Schreib-Lese-Zugriff.

TRUE: Der Schleppzeiger wird angezeigt.

FALSE: Der Schleppzeiger ist ausgeblendet.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

ShowPosition-Eigenschaft

Beschreibung

TRUE, wenn die Schieberposition angezeigt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

ShowRuler-Eigenschaft

Lineal anzeigen - ShowRuler

Legt fest, ob bei einem Bildaufschlag das Lineal zur Abfrage der Koordinatenpunkte angezeigt wird.

Wert	Erklärung
TRUE	Das Lineal zur Abfrage der Koordinatenpunkte wird angezeigt.
FALSE	Das Lineal zur Abfrage der Koordinatenpunkte wird nicht angezeigt.

Das Attribut ist mit dem Namen **ShowRuler** dynamisierbar. Der Datentyp ist BOOLEAN.

ShowRulerImmediately-Eigenschaft

Beschreibung

TRUE, wenn bei einem Bildaufschlag das Lineal zur Ermittlung der Koordinatenwerte angezeigt werden soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

ShowRulerInAxis-Eigenschaft

ShowRulerInAxis

Legt fest, ob die Lineale auch in den Zeitachsen angezeigt werden.

Wert	Erklärung
TRUE	Die Lineale werden auch in den Zeitachsen angezeigt.
FALSE	Die Lineale werden nicht in den Zeitachsen angezeigt.

Das Attribut ist mit dem Namen **ShowRulerInAxis** dynamisierbar. Der Datentyp ist BOOLEAN.

ShowScrollbars-Eigenschaft

Rollbalken - ShowScrollbars

Legt fest, ob die Rollbalken angezeigt werden.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	nein	Die Rollbalken werden nicht angezeigt.
1	bei Bedarf	Die Rollbalken werden angezeigt, wenn der Platzbedarf des Controls in größer ist als der aktuelle Anzeigebereich.
2	immer	Die Rollbalken werden immer angezeigt.

Das Attribut ist mit dem Namen **ShowScrollbars** dynamisierbar. Der Datentyp ist LONG.

ShowSlider-Eigenschaft

ShowSlider

Legt fest, ob im Control eine Zeitleiste angezeigt wird.

Das Attribut ist mit dem Namen **ShowSlider** dynamisierbar. Der Datentyp ist BOOLEAN.

ShowSortButton-Eigenschaft

Sortiertaste verwenden - ShowSortButton

Legt fest, ob die Sortiertaste über dem vertikalen Rollbalken angezeigt wird. Über die Sortiertaste sortieren Sie mit einem Mausklick die ausgewählte Spalte in der projektierten

Sortierreihenfolge. Wenn in der Tabelle kein vertikaler Rollbalken vorhanden ist, wird die Sortiertaste nicht angezeigt.

Wert	Erklärung
TRUE	Sie können über die Sortiertaste die ausgewählte Spalte sortieren.
FALSE	Die Sortiertaste wird nicht angezeigt.

Das Attribut ist mit dem Namen **ShowSortButton** dynamisierbar. Der Datentyp ist BOOLEAN.

ShowSortIcon-Eigenschaft

Sortiersymbol anzeigen - ShowSortIcon

Legt fest, ob das Sortiersymbol angezeigt wird.

Wert	Erklärung
TRUE	Das Sortiersymbol wird angezeigt.
FALSE	Das Sortiersymbol wird nicht angezeigt.

Das Attribut ist mit dem Namen **ShowSortIcon** dynamisierbar. Der Datentyp ist BOOLEAN.

ShowSortIndex-Eigenschaft

Sortierindex anzeigen - ShowSortIndex

Legt fest, ob ein Sortierindex angezeigt wird.

Wert	Erklärung
TRUE	Ein Sortierindex wird angezeigt.
FALSE	Ein Sortierindex wird nicht angezeigt.

Das Attribut ist mit dem Namen **ShowSortIndex** dynamisierbar. Der Datentyp ist BOOLEAN.

ShowSpanNames-Eigenschaft

Beschreibung

TRUE, wenn in der Spalte "Wert" des Linealfensters im Kurven-Control neben dem Messwert und der Statusanzeige "i" und "u" zusätzlich ein Bereichsnamen angezeigt wird. BOOLEAN Schreib-Lese-Zugriff.

ShowStatisticRuler-Eigenschaft

ShowStatisticRuler

Legt fest, ob bei einem Bildaufschlag die Lineale für den Statistikbereich angezeigt werden.

Wert	Erklärung
TRUE	Die Lineale für den Statistikbereich werden angezeigt.
FALSE	Die Lineale für den Statistikbereich werden nicht angezeigt.

Das Attribut ist mit dem Namen **ShowStatisticRuler** dynamisierbar. Der Datentyp ist BOOLEAN.

ShowThumb-Eigenschaft

Beschreibung

TRUE, wenn der Schieber angezeigt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

ShowTitle-Eigenschaft

Fensterüberschrift - ShowTitle

Legt die Darstellung der Fensterüberschrift des Controls fest.

Wert	Bezeichnung	Erläuterung
0	nein	Keine Fensterüberschrift.
1	normal	Die Fensterüberschrift besteht aus einem WinCC Symbol und Text. Der Text wird im Feld "Text" eingegeben.
2	schmal	Die Fensterüberschrift besteht nur aus Text. Der Text wird im Feld "Text" eingegeben.

Das Attribut ist mit dem Namen **ShowTitle** dynamisierbar. Der Datentyp ist LONG.

ShowToolbar-Eigenschaft

ShowToolbar

Legt fest, ob im Control eine Symbolleiste angezeigt wird.

Das Attribut ist mit dem Namen **ShowToolbar** dynamisierbar. Der Datentyp ist BOOLEAN.

ShowTrendIcon-Eigenschaft

ShowTrendIcon

Legt fest, ob unterhalb der Wertachsen ein Symbol eingeblendet wird. Das Symbol zeigt, welche Kurve im Vordergrund angezeigt wird.

Das Attribut ist mit den Namen **ShowTrendIcon** dynamisierbar. Der Datentyp ist BOOLEAN.

ShowValuesExponentialX-Eigenschaft

Beschreibung

TRUE, wenn die von der Funktion "Anzeige Wert an dieser Stelle" ermittelte X-Koordinate eines Messwertes einer über "Index" referenzierten Kurve in Exponentialschreibweise dargestellt wird. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "TimeAxisX". BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

ShowValuesExponentialY-Eigenschaft

Beschreibung

TRUE, wenn die von der Funktion "Anzeige Wert an dieser Stelle" ermittelte Y-Koordinate eines Messwertes einer über "Index" referenzierten Kurve in Exponentialschreibweise dargestellt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

ShowWarning-Eigenschaft

Beschreibung

Steuert die Anzeige des "Warnbereichs" auf der Instrumentenskala. BOOLEAN Schreib-Lese-Zugriff.

TRUE: Der Bereich wird mit der durch Warnungsfarbe festgelegten Farbe gekennzeichnet.

FALSE: Die farbliche Kennzeichnung des Bereichs ist ausgeschaltet.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

SignificantMask-Eigenschaft

Beschreibung

Wird in Runtime zur Darstellung der aktiven Meldeklasse mit der höchsten Priorität benötigt. Der Wert der SignificantMask-Eigenschaft stellt einen systeminternen Ausgabewert dar und erfordert keine spezifische Projektierung durch den Anwender. Die Aktualisierung erfolgt in Runtime durch Anklicken des Objektes.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

Sizeable-Egienschaft

Größe änderbar - Sizeable

Legt fest, ob die Größe des Controls in Runtime veränderbar ist.

Wert	Erklärung
TRUE	Die Größe des Controls ist in Runtime veränderbar.
FALSE	Die Größe des Controls ist nicht in Runtime veränderbar.

Das Attribut ist mit dem Namen **Sizeable** dynamisierbar. Der Datentyp ist BOOLEAN.

SkinName-Eigenschaft

Stil - SkinName

In diesem Auswahlfeld wird der Stil des Controls festgelegt.

Folgende Einstellungen stehen zur Verfügung:

Wert	Bezeichnung	Erläuterung
	Projekteinstellung	Der Stil entspricht den Projekteinstellungen im WinCC Explorer.
0	Einfach	"Klassischer" WinCC Stil
1	Standard	Neuer WinCC V7 Stil
	Basic Process Control	Der Stil ist für die interne Verwendung bei Basic Process Control reserviert.

Das Attribut ist mit dem Namen **SkinName** dynamisierbar. Der Datentyp ist STRING.

Sm - Sq

SmallChange-Eigenschaft

Beschreibung

Legt fest, um wieviel Schritte der Regler mit einem Mausklick verschoben werden kann oder gibt diesen Wert zurück.

Siehe auch

Slider (Seite 221)

ScreenItem-Objekt (Seite 134)

SmartTag-Eigenschaft

Beschreibung

Gibt ein Objekt vom Typ "SmartTag" zurück.

Siehe auch

SmartTags-Objekt (Seite 145)

SortOrder-Eigenschaft

Beschreibung

Legt die Sortierreihenfolge der Meldeblöcke im Meldefenster fest.

SortSequence-Eigenschaft

Sortierabfolge bei Mausklick - SortSequence

Legt fest, wie die Sortierreihenfolge durch Mausklick verändert werden kann.

Folgende Sortierreihenfolgen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	auf/ab/keine	Per Mausklick können Sie zwischen aufsteigender, absteigender und keiner Sortierung umschalten.
1	auf/ab	Per Mausklick können Sie zwischen aufsteigender und absteigender Sortierung umschalten.

Das Attribut ist mit dem Namen **SortSequence** dynamisierbar. Der Datentyp ist LONG.

SourceBeginTime-Eigenschaft

Beschreibung

Legt für Online-Variablen und Archiv-Variablen ("ProviderType" = -1) den Startzeitpunkt des im Kurvenfenster darzustellenden Zeitbereichs einer über "Index" referenzierten Kurve fest. Bei der Änderung von "SourceBeginTime" können unzulässige Kombinationen mit anderen Attributen zur Datenanbindung entstehen. Deshalb muss vor einer Änderung von "SourceBeginTime" die sofortige Übernahme der Änderung mit "FreezeProviderConnections" verhindert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

SourceControl-Eigenschaft

Quelle - SourceControl

Legt fest, mit welchem Control das WinCC RulerControl verbunden wird.

Das Attribut ist mit dem Namen **SourceControl** dynamisierbar. Der Datentyp ist STRING.

SouceControlType-Eigenschaft

Typ - SouceControlType

Gibt den Typ des Controls an, mit dem das WinCC RulerControl im Feld "Quelle" verbunden ist.

Wert	Bezeichnung	Erläuterung
0	kein	Das RulerControl ist mit keiner Quelle verbunden.
1	OnlineTrend Control	Das RulerControl ist mit einem OnlineTrendControl verbunden.
2	OnlineTable Control	Das RulerControl ist mit einem OnlineTableControl verbunden.
3	FunctionTrend Control	Das RulerControl ist mit einem FunctionTrendControl verbunden.

Das Attribut ist mit dem Namen **SourceControlType** dynamisierbar. Der Datentyp ist LONG.

SourceEndTime-Eigenschaft

Beschreibung

Legt für Online-Variablen und Archiv-Variablen ("ProviderType" = -1) den Endzeitpunkt des im Kurvenfenster darzustellenden Zeitbereichs einer über "Index" referenzierten Kurve fest. Bei der Änderung von "SourceEndTime" können unzulässige Kombinationen mit anderen Attributen zur Datenanbindung entstehen. Deshalb muss vor einer Änderung von "SourceEndTime" die sofortige Übernahme der Änderung mit "FreezeProviderConnections" verhindert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

SourceNumberOfUAValues-Eigenschaft

Beschreibung

Legt für Werte aus Anwenderarchiven ("ProviderType" = -2) einer über "Index" referenzierten Kurve die Anzahl der Werte fest, die aus dem Anwenderarchiv geladen werden sollen. Bei der Änderung von "SourceNumberOfUAValues" können unzulässige Kombinationen mit anderen Attributen zur Datenanbindung entstehen. Deshalb muss vor einer Änderung von "SourceNumberOfUAValues" die sofortige Übernahme der Änderung mit "FreezeProviderConnections" verhindert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

SourceNumberOfValues-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. Für Online-Variablen und Archiv-Variablen ("ProviderType" = -1) legt "SourceNumberOfValues" die Anzahl der Werte fest, die im Kurvenfenster dargestellt werden sollen. Ob die Angabe ausgewertet wird, ist abhängig vom Wert der Eigenschaft "SourceTimeRange".

Bei der Änderung von "SourceNumberOfValues" können unzulässige Kombinationen mit anderen Attributen zur Datenanbindung entstehen. Deshalb muss vor einer Änderung von "SourceNumberOfValues" die sofortige Übernahme der Änderung mit "FreezeProviderConnections" verhindert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

SourceTagNameX-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. Für Online-Variablen und Archiv-Variablen ("ProviderType" = -1) legt "SourceTagNameX" die Variable fest, die entlang der X-Achse dargestellt werden sollen. Bei der Änderung von "SourceTagNameX" können unzulässige Kombinationen mit anderen Attributen zur Datenanbindung entstehen. Deshalb muss vor einer Änderung von "SourceTagNameX" die sofortige Übernahme der Änderung mit "FreezeProviderConnections" verhindert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

SourceTagNameY-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. Für Online-Variablen und Archiv-Variablen ("ProviderType" = -1) legt "SourceTagNameY" die Variable fest, die entlang der Y-Achse dargestellt werden sollen. Bei der Änderung von "SourceTagNameY" können unzulässige Kombinationen mit anderen Attributen zur Datenanbindung entstehen. Deshalb muss vor einer Änderung von "SourceTagNameY" die sofortige Übernahme der Änderung mit "FreezeProviderConnections" verhindert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

SourceTagProviderDataX-Eigenschaft

Beschreibung

Kann nur über den Dialog "Eigenschaften von WinCC Function Trend Control" verändert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

SourceTagProviderDataY-Eigenschaft

Beschreibung

Kann nur über den Dialog "Eigenschaften von WinCC Function Trend Control" verändert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

SourceTimeRange-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. Für Online-Variablen und Archiv-Variablen ("ProviderType" = -1) legt "SourceTimeRange" fest, wie der im Kurvenfenster darzustellende Zeitbereich definiert ist. Bei der Änderung von "SourceTimeRange" können unzulässige Kombinationen mit anderen Attributen zur Datenanbindung entstehen. Deshalb muss vor einer Änderung von "SourceTimeRange" die sofortige Übernahme der Änderung mit "FreezeProviderConnections" verhindert werden.

0: Der darzustellende Zeitbereich ist durch Startzeitpunkt (SourceBeginTime) und die Anzahl der Wertepaare (SourceNumberOfValues) festgelegt.

-1: Der darzustellende Zeitbereich ist durch Startzeitpunkt (SourceBeginTime) und Endzeitpunkt (SourceEndTime) festgelegt

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

SourceUAArchive-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. Für Werte aus Anwenderarchiven ("ProviderType" = -2) legt "SourceUAArchive" das Anwenderarchiv fest, aus dem die Werte geladen werden sollen. Bei der Änderung von "SourceUAArchive" können unzulässige Kombinationen mit anderen Attributen zur Datenanbindung entstehen. Deshalb muss vor einer

Änderung von "SourceUAArchive" die sofortige Übernahme der Änderung mit "FreezeProviderConnections" verhindert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

SourceUAArchiveStartID-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. Für Werte aus Anwenderarchiven ("ProviderType" = -2) legt "SourceUAArchiveStartID" den Datensatz fest, ab dem die Werte aus dem Anwenderarchiv geladen werden sollen. Bei der Änderung von "SourceUAArchiveStartID" können unzulässige Kombinationen mit anderen Attributen zur Datenanbindung entstehen. Deshalb muss vor einer Änderung von "SourceUAArchiveStartID" die sofortige Übernahme der Änderung mit "FreezeProviderConnections" verhindert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

SourceUAColumnX-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. Für Werte aus Anwenderarchiven ("ProviderType" = -2) legt "SourceUAColumnX" die Spalte im Anwenderarchiv fest, aus der die Werte für die X-Achse geladen werden sollen. Bei der Änderung von "SourceUAColumnX" können unzulässige Kombinationen mit anderen Attributen zur Datenanbindung entstehen. Deshalb muss vor einer Änderung von "SourceUAColumnX" die sofortige Übernahme der Änderung mit "FreezeProviderConnections" verhindert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

SourceUAColumnY-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. Für Werte aus Anwenderarchiven ("ProviderType" = -2) legt "SourceUAColumnY" die Spalte im Anwenderarchiv fest, aus der die Werte für die Y-Achse geladen werden sollen. Bei der Änderung von "SourceUAColumnY" können unzulässige Kombinationen mit anderen Attributen zur Datenanbindung entstehen. Deshalb muss vor einer Änderung von "SourceUAColumnY" die sofortige Übernahme der Änderung mit "FreezeProviderConnections" verhindert werden.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

SquareExtent-Eigenschaft

Beschreibung

TRUE, wenn sich die Größe der Uhr mit der Maus durch Ziehen an den Markierungspunkten in jedem gewünschten Seitenverhältnis einstellen lässt. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

St - Sy

StartAngle-Eigenschaft

Beschreibung

Legt den Beginn des Objektes fest oder gibt ihn zurück. Die Angabe erfolgt im Uhrzeigersinn in Grad, beginnend bei 12:00 Uhr.

Siehe auch

Kreissegment (Seite 164)
Kreisbogen (Seite 162)
Ellipsensegment (Seite 158)
Ellipsenbogen (Seite 156)
ScreenItem-Objekt (Seite 134)

State-Eigenschaft

Beschreibung

Gibt den Status einer Meldung aus.

Die folgende Tabelle zeigt die möglichen Status einer Meldung:

State	Status der Meldung
1	Gekommen
2	Gegangen
5	Gekommen und Kommentar
6	Gegangen und Kommentar

Siehe auch

Alarms-Objekt (Auflistung) (Seite 120)

Statusbar-Eigenschaft

Beschreibung

TRUE, wenn eine Statuszeile angezeigt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)
WinCC Online Trend Control (vor WinCC V7) (Seite 291)
WinCC Function Trend Control (vor WinCC V7) (Seite 287)
WinCC Alarm Control (vor WinCC V7) (Seite 285)
ScreenItem-Objekt (Seite 134)

StatusbarBackColor-Eigenschaft

Hintergrundfarbe - StatusbarBackColor

Gibt die Hintergrundfarbe für die Statuszeile an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Damit die Einstellung wirksam wird, muss die Option "anzeigen" bzw. "StatusbarUseBackColor" aktiviert sein.

Das Attribut ist mit dem Namen **StatusbarBackColor** dynamisierbar. Der Datentyp ist LONG.

StatusbarElementAdd-Eigenschaft

Neu - StatusbarElementAdd

Legt ein neues, benutzerdefiniertes Element der Statuszeile an. Den von WinCC vergebenen Namen können Sie im Feld "Objektname" ändern.

Das Attribut ist mit dem Namen **StatusbarElementAdd** dynamisierbar. Der Datentyp ist STRING.

StatusbarElementAutoSize-Eigenschaft

Automatisch - StatusbarElementAutoSize

Legt fest, ob die Breite des ausgewählten Elements der Statuszeile automatisch eingestellt wird.

Wert	Erklärung
TRUE	Die Breite des ausgewählten Elements wird automatisch eingestellt.
FALSE	Die Breite des ausgewählten Elements wird nicht automatisch eingestellt.

Das Attribut ist mit dem Namen **StatusbarElementAutoSize** dynamisierbar. Der Datentyp ist BOOLEAN.

StatusbarElementCount-Eigenschaft

StatusbarElementCount

Gibt die Anzahl der projektierbaren Elemente der Statuszeile an.

Das Attribut ist mit dem Namen **StatusbarElementCount** dynamisierbar. Der Datentyp ist LONG.

StatusbarElementIconId-Eigenschaft

StatusbarElementIconId

Festgelegte Zuordnung von Ident-Nummer und Symbol eines Elementes der Statuszeile.

Das Attribut ist für benutzerdefinierte Elemente der Statuszeile mit dem Namen **StatusbarElementIconId** dynamisierbar. Der Datentyp ist LONG.

StatusbarElementID-Eigenschaft

Objekt-ID - StatusbarElementID

Eindeutige Ident-Nummer für das gewählte Element der Statuszeile. Die Ident-Nummer wird von WinCC vergeben und kann nicht geändert werden.

Das Attribut ist mit dem Namen **StatusbarElementID** dynamisierbar. Der Datentyp ist LONG.

StatusbarElementIndex-Eigenschaft

StatusbarElementIndex

Referenziert ein Element der Statuszeile. Unter Verwendung des Attributs können Sie einem bestimmten Element der Statuszeile die Werte anderer Attribute zuweisen.

Gültige Werte für "StatusbarElementIndex" liegen zwischen 0 und "StatusbarElementCount" minus 1. Das Attribut "StatusbarElementCount" gibt die Anzahl der projektierbaren Elemente der Statuszeile an.

Das Attribut "StatusbarElementIndex" ist über das Attribut **StatusbarElementIndex** dynamisierbar. Der Datentyp ist LONG.

StatusbarElementName-Eigenschaaft

Objektname - StatusbarElementName

Zeigt den Objektnamen für das gewählte Element der Statuszeile an. Den Objektnamen für benutzerdefinierte Elemente der Statuszeile können Sie ändern.

Das Attribut "StatusbarElementName" für benutzerdefinierte Elemente ist über das Attribut **StatusbarElementRename** dynamisierbar. Der Datentyp ist STRING.

StatusbarElementRemove-eigenschaft

Entfernen - StatusbarElementRemove

Entfernt das ausgewählte Element der Statuszeile. Nur benutzerdefinierte Elemente der Statuszeile können Sie aus der Liste entfernen.

Das Attribut ist mit dem Namen **StatusbarElementRemove** dynamisierbar. Der Datentyp ist STRING.

StatusbarElementRename-Eigenschaft

StatusbarElementRename

Ändert den Namen des benutzerdefinierten Elements der Statuszeile, das über das Attribut "StatusbarElementIndex" referenziert wird.

Für benutzerdefinierte Elemente ist das Attribut mit dem Namen **StatusbarElementRename** dynamisierbar. Mit "StatusbarElementRename" dynamisieren Sie auch das Attribut "StatusbarElementName". Der Datentyp ist STRING.

StatusbarElementRepos-Eigenschaft

Auf/Ab - StatusbarElementRepos

Ändert die Reihenfolge der Tastenfunktionen. "Auf" und "Ab" verschieben das ausgewählte Element der Statuszeile in der Liste nach oben oder unten. Dadurch wird in Runtime das Element in der Statuszeile des Controls weiter vorne oder hinten platziert.

Das Attribut ist mit dem Namen **StatusbarElementRepos** dynamisierbar. Der Datentyp ist LONG.

StatusbarElementText-Eigenschaft

StatusbarElementText

Legt den Text fest, der für das Element in der Statuszeile angezeigt wird. Sie können das Attribut "StatusbarElementText" für benutzerdefinierte Elemente ändern.

Für benutzerdefinierte Elemente ist das Attribut mit dem Namen **StatusbarElementText** dynamisierbar. Der Datentyp ist STRING.

StatusbarElementTooltipText-Eigenschaft

StatusbarElementTooltipText

Legt den Text für den Tooltip des benutzerdefinierten Elements der Statuszeile fest.

Das Attribut ist mit dem Namen **StatusbarElementTooltipText** dynamisierbar. Der Datentyp ist STRING.

StatusbarElementVisible-Eigenschaft

Elemente der Statuszeile - StatusbarElementVisible

Aktivieren Sie in der Liste die Elemente der Statuszeile, die sie in Runtime anzeigen wollen.

Klicken Sie auf einen Eintrag in der Liste, um die Eigenschaften anzupassen oder um die Position in der Statuszeile des Controls mit den Tasten "Auf" und "Ab" zu ändern.

Wert	Erklärung
TRUE	Das Element der Statuszeile wird angezeigt.
FALSE	Das Element der Statuszeile wird nicht angezeigt.

Das Attribut ist mit dem Namen **StatusbarElementVisible** dynamisierbar. Der Datentyp ist BOOLEAN.

StatusbarElementUserDefined-Eigenschaft

StatusbarElementUserDefined

Zeigt an, ob das Element der Statuszeile vom Projekteur als neues, benutzerdefiniertes Element hinzugefügt wurde.

Wert	Erklärung
TRUE	Das Element der Statuszeile ist benutzerdefiniert.
FALSE	Das Element der Statuszeile ist vom System vorgegeben.

Das Attribut ist mit dem Namen **StatusbarElementUserDefined** dynamisierbar. Der Datentyp ist BOOLEAN.

StatusbarElementWidth-Eigenschaft

Breite in Pixel - StatusbarElementWidth

Gibt die Breite des ausgewählten Elements der Statuszeile in Pixel an. Wenn die Option "Automatisch" nicht aktiviert ist, können Sie die Breite festlegen.

Das Attribut ist mit dem Namen **StatusbarElementWidth** dynamisierbar. Der Datentyp ist LONG.

StatusbarFontColor-Eigenschaft

StatusbarFontColor

Gibt die Schriftfarbe der Texte in der Statuszeile an.

Das Attribut ist mit dem Namen **StatusbarFontColor** dynamisierbar. Der Datentyp ist LONG.

StatusbarPanes-Eigenschaft

Beschreibung

Legt die in der Statuszeile darzustellenden Elemente fest. Schreib-Lese-Zugriff.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

StatusbarShowTooltips-Eigenschaft

Tooltips - StatusbarShowTooltips

Legt fest, ob in Runtime die Tooltips zu den Elementen der Statuszeile angezeigt werden.

Wert	Erklärung
TRUE	Die Tooltips werden angezeigt.
FALSE	Die Tooltips werden nicht angezeigt.

Das Attribut ist mit dem Namen **StatusbarShowTooltips** dynamisierbar. Der Datentyp ist BOOLEAN.

Das Attribut zum Festlegen des Tooltip-Texts ist "StatusbarElementTooltipText".

StatusbarText-Eigenschaft

StatusbarText

Festgelegter Text in der Statuszeile.

Das Attribut ist mit dem Namen **StatusbarText** dynamisierbar. Der Datentyp ist STRING.

StatusbarUseBackColor-Egienschaft

anzeigen Hintergrundfarbe - StatusbarUseBackColor

Legt fest, ob die Hintergrundfarbe der Statuszeile angezeigt wird.

Wert	Erklärung
TRUE	Die Hintergrundfarbe der Statuszeile wird angezeigt.
FALSE	Die Hintergrundfarbe der Statuszeile wird nicht angezeigt.

Das Attribut ist mit dem Namen **StatusbarUseBackColor** dynamisierbar. Der Datentyp ist BOOLEAN.

StatusbarVisible-Eigenschaft

Statuszeile anzeigen - StatusbarVisible

Legt fest, ob die Statuszeile des Controls angezeigt wird.

Wert	Erklärung
TRUE	Die Statuszeile wird angezeigt.
FALSE	Die Statuszeile wird nicht angezeigt.

Das Attribut ist mit dem Namen **StatusbarVisible** dynamisierbar. Der Datentyp ist BOOLEAN.

StepSeconds-Eigenschaft

StepSeconds

Legt das Intervall für den Vorwärts- oder Rückwärtssprung bei Filmen fest.

Das Attribut ist mit dem Namen **StepSeconds** dynamisierbar. Der Datentyp ist LONG.

Stretch-Eigenschaft

Beschreibung

Legt fest, ob bei Änderung der Symbolgröße das Seitenverhältnis erhalten bleibt oder veränderlich ist. BOOLEAN Schreib-Lese-Zugriff.

- FALSE: Bei Änderung der Symbolgröße bleibt das Seitenverhältnis des Symbols erhalten.
- TRUE: Bei Änderung der Symbolgröße kann gleichzeitig das Seitenverhältnis des Symbols geändert werden.

Siehe auch

HMI Symbol Library (Seite 248)

ScreenItem-Objekt (Seite 134)

SymbolAppearance-Eigenschaft

Vordergrundmodus (SymbolAppearance)

Legt die Erscheinungsform des Symbols fest.

Folgende Einstellungen sind möglich:

Wert	Beschreibung	Erläuterung
0	Original	Die Erscheinungsform des Symbols entspricht der mehrfarbigen Darstellung in der Auswahl der Registerkarte "Symbole".
1	Schattiert	Linien der Farbe "Schwarz" bleiben als Umrisslinien erhalten. Andersfarbige Elemente des Symbols werden als Helligkeitsstufen der aktuellen Vordergrundfarbe dargestellt.
2	Massiv	Linien der Farbe "Schwarz" bleiben als Umrisslinien erhalten. Allen andersfarbigen Elementen des Symbols wird der Farbwert der aktuellen Vordergrundfarbe zugewiesen.
3	Umriss	Linien der Farbe "Schwarz" bleiben als Umrisslinien erhalten. Allen andersfarbigen Elementen des Symbols wird der Farbwert des Hintergrundes zugewiesen.

Das Attribut ist mit dem Namen **SymbolAppearance** dynamisierbar. Der Datentyp ist LONG.

1.14.4.19 T

Ta -Tic

TableColor-Eigenschaft

Hintergrund Zeilenfarbe 1 - TableColor

Gibt die Hintergrundfarbe für die Zeilen an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **TableColor** dynamisierbar. Der Datentyp ist LONG.

TableColor2-Eigenschaft

Hintergrund Zeilenfarbe 2 - TableColor2

Gibt die Hintergrundfarbe der "Zeilenfarbe 2" an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Einstellung ist in Runtime nur wirksam, wenn die Option "Zeilenfarbe 2" bzw. "UseTableColor2" aktiviert ist. Dann werden die Hintergrundfarben der "Zeilenfarbe 1" und der "Zeilenfarbe 2" abwechselnd verwendet.

Das Attribut ist mit dem Namen **TableColor2** dynamisierbar. Der Datentyp ist LONG.

TableFocusOnButtonCommand-Eigenschaft

Beschreibung

Legt fest, ob in Runtime beim Klick auf ein Button in einem Skript der Fokus auf die Tabelle des Control gesetzt wird.

TableForeColor-Eigenschaft

Schrift Zeilenfarbe 1 - TableForeColor

Gibt die Schriftfarbe der Zeilen an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **TableForeColor** dynamisierbar. Der Datentyp ist LONG.

TableForeColor2-Eigenschaft

Schrift Zeilenfarbe 2 - TableForeColor2

Gibt die Schriftfarbe der "Zeilenfarbe 2" an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Einstellung ist in Runtime nur wirksam, wenn die Option "Zeilenfarbe 2" bzw. "UseTableColor2" aktiviert ist. Dann werden die Schriftfarben der "Zeilenfarbe 1" und der "Zeilenfarbe 2" abwechselnd verwendet.

Das Attribut ist mit dem Namen **TableForeColor2** dynamisierbar. Der Datentyp ist LONG.

TagName-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve. "TagName" legt fest, welche Variable mit dieser Kurve verknüpft ist. Die Angabe erfolgt in der Form "Archivname\Variablenname" zur Darstellung von Variablen eines Prozesswertarchives bzw. "Variablenname" zur Darstellung einer internen oder externen Variablen, die nicht in einem Archiv gespeichert wird.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

TagPrefix-Eigenschaft

Beschreibung

Legt das Variablen-Präfix fest, das allen Variablen vorangestellt wird, die im Bildfenster-Objekt enthalten sind oder gibt es zurück. So behält ein Bild, das in einem Bildfenster eingebunden ist, den Zugriff auf eigene Variablen, während ein anderes auf andere Variablen zugreift.

Eine Änderung des TagPrefix wird beim erneuten Laden eines Bildes wirksam. Bei einem Bildwechsel geschieht dies automatisch, ansonsten muss der Bildname neu zugewiesen werden.

Das TagPrefix ist frei definierbar, muss aber mit dem Namen der Strukturvariablen übereinstimmen.

Hinweis

Die TagPrefix-Eigenschaft steht für die Controls nicht zur Verfügung.

Siehe auch

Bildfenster (Seite 190)

ScreenItem-Objekt (Seite 134)

Tags-Eigenschaft

Beschreibung

Liefert ein Objekt vom Typ "Tags" zurück.

Tags (read-only)

Beispiel

Das folgende Beispiel greift auf die Variable "Tag1" zu:

```
'VBS86
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
```

Siehe auch

Tags-Objekt (Auflistung) (Seite 149)

HMIRuntime-Objekt (Seite 127)

TagProviderClsid-Eigenschaft

Beschreibung

Die Eigenschaft Index referenziert eine Kurve. "TagProviderClsid" legt fest, ob in dieser Kurve eine Online-Variable oder archivierte Werte dargestellt werden sollen. Die Angabe wird nur für Online-Variablen und Archiv-Variablen ("ProviderType" = -1) ausgewertet.

{A3F69593-8AB0-11D2-A440-00A0C9DBB64E}: Online-Variable.

{416A09D2-8B5A-11D2-8B81-006097A45D48}: Werte werden aus einem Prozesswertarchiv oder einem Anwenderarchiv gelesen.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

Template-Eigenschaft

Beschreibung

Gibt die Vorlage für die Anzeige des Fensterinhalts des Objektes "Applikationsfenster" zurück. Nur Lese-Zugriff.

Abhängig vom Wert der Eigenschaft sind die folgenden Vorlagen möglich:

Fensterinhalt = Global Script

"GSC-Diagnose"
Das Applikationsfenster wird von Applikationen des Global Script versorgt. Dargestellt werden die Ergebnisse des Diagnosesystems.

"GSC-Runtime"
Das Applikationsfenster wird von Applikationen des Global Script versorgt. Dargestellt werden Analyseergebnisse zum Verhalten in Runtime.

Fensterinhalt = Print Jobs

"All Jobs":
Das Applikationsfenster wird vom Protokolliersystem versorgt. Die verfügbaren Protokolle werden als Liste dargestellt.

"All Jobs - Context Menu":
Das Applikationsfenster wird vom Protokolliersystem versorgt. Die verfügbaren Protokolle werden als Liste dargestellt. Ein Kontextmenü ermöglicht die Auswahl der Druckoptionen, die Darstellung einer Druckvorschau sowie den Ausdruck eines Protokolls.

"Job Detail View":
Das Applikationsfenster wird vom Protokolliersystem versorgt. Die verfügbaren Protokolle werden in einem Auswahlmenü dargestellt. Für das gewählte Protokoll werden Detailinformationen angezeigt.

"Selected Jobs - Context Menu":
Das Applikationsfenster wird vom Protokolliersystem versorgt. Die verfügbaren Protokolle werden als Liste dargestellt. Diese Liste beinhaltet nur die Protokolle, für die Sie im Dialog "Druckauftrageigenschaften" die Option "Markierung für Druckauftragsliste" aktiviert haben. Ein Kontextmenü ermöglicht die Auswahl der Druckoptionen, die Darstellung einer Druckvorschau sowie den Ausdruck eines Protokolls.

Siehe auch

ScreenItem-Objekt (Seite 134)

Applikationsfenster (Seite 185)

Text-Eigenschaft

Beschreibung

Legt die Beschriftung für ein Objekt fest oder gibt sie zurück.

Siehe auch

Radio-Box (Seite 217)
Check-Box (Seite 215)
Button (Seite 212)
Statischer Text (Seite 178)
ScreenItem-Objekt (Seite 134)

ThumbBackColor-Eigenschaft

Beschreibung

Legt die Farbe des Schiebers fest.

Siehe auch

WinCC Slider Control (Seite 278)
ScreenItem-Objekt (Seite 134)

TicColor-Eigenschaft

Beschreibung

Legt die Farbe der Skalenteilung fest. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)
ScreenItem-Objekt (Seite 134)

TicFont-Eigenschaft

Beschreibung

Steuert die Darstellung der Beschriftung der Skalenteilung. Nur Lese-Zugriff.

Folgende Eigenschaften sind einstellbar:

- Schriftart
- Schriftstil
- Schriftgröße
- Effekt "durchgestrichen"
- Effekt "unterstrichen"

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

TicOffset-Eigenschaft

Beschreibung

Legt den Durchmesser des gedachten Kreises fest, auf dem sich die Skalenteilung befindet. Der Wert ist bezogen auf den kleineren Wert der Geometrieeigenschaften Breite und Höhe.

Die Hauptstriche der Skalenteilung liegen dabei mit ihrem nach außen gerichteten Ende auf diesem Kreis.

Wertebereich 0 bis 1.

0: Die Skalenteilung befindet sich in der Mitte der Skalenscheibe.

1: Der Durchmesser des gedachten Kreises für die Skalenteilung ist der kleinere Wert der Geometrieeigenschaften Breite und Höhe.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

TicTextColor-Eigenschaft

Beschreibung

Legt die Farbe der Beschriftung der Skalenteilung fest.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

TicTextOffset-Eigenschaft

Beschreibung

Legt den Durchmesser des gedachten Kreises fest, auf dem sich die Beschriftung der Skalenteilung befindet. Der Wert ist bezogen auf den kleineren Wert der Geometrieeigenschaften Breite und Höhe.

Wertebereich 0 bis 1.

0: Die Beschriftung befindet sich in der Mitte der Skalenscheibe.

1: Der Durchmesser des gedachten Kreises für die Beschriftung ist der kleinere Wert der Geometrieeigenschaften Breite und Höhe. Ein Teil der Beschriftung kann dadurch außerhalb der Objektbegrenzung liegen und ist damit unsichtbar.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

TicWidth-Eigenschaft

Beschreibung

Legt die Länge der Hauptstriche der Skalenteilung fest. Der Wert ist bezogen auf die Hälfte des kleineren Werts der Geometrieeigenschaften Breite und Höhe.

Die Länge der Teilstriche für die Feinteilung ist 0,5*Skalenbreite.

Wertebereich 0 bis Skalenabstand.

0: Es ist keine Skalenteilung vorhanden. Auch die Aufteilung der Skala in Bereiche ist nicht sichtbar.

Skalenabstand: Die Skalenteilung reicht vom Mittelpunkt der Skalenscheibe bis zum Wert, der durch Skalenabstand festgelegt ist.

Siehe auch

ScreenItem-Objekt (Seite 134)

WinCC Gauge Control (Seite 261)

Ticks-Eigenschaft

Beschreibung

TRUE, wenn das Zifferblatt angezeigt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

TicksColor-Eigenschaft

Beschreibung

Legt die Farbe der Stundenmarkierungen auf dem Zifferblatt der Analoguhr fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Digital Analog Clock (Seite 255)

ScreenItem-Objekt (Seite 134)

TickStyle-Eigenschaft

Beschreibung

Mit dem Attribut legen Sie das Erscheinungsbild der Skala fest. Wertebereich: 0 bis 3.

Auf Grund der automatischen Skalierung ist es möglich, dass teilweise zwei Teilstriche der Skala direkt nebeneinander liegen (scheinbar breiter Teilstrich). Durch geringfügiges Verlängern oder Verkürzen des Sliderobjekts kann dieser Effekt korrigiert werden.

Darüber hinaus können Sie auch die Darstellung der Skalierung völlig unterdrücken ("WithAxes").

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

TimeAxis - TimeBase

TimeAxis-Eigenschaft

Beschreibung

Legt fest, ob im Kurvenfenster für alle Kurven eine gemeinsame Zeitachse verwendet werden soll.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

TimeAxisActualize-Eigenschaft

Aktualisieren - TimeAxisActualize

Legt fest, ob die Werte der ausgewählten Zeitachse aktualisiert werden.

Wert	Erklärung
TRUE	Das der Zeitachse zugeordnete Kurvenfenster wird aktualisiert.
FALSE	Das der Zeitachse zugeordnete Kurvenfenster wird nicht aktualisiert. Diese Einstellung ist sinnvoll, wenn eine archivierte Kurve mit einer aktuellen Kurve verglichen wird.

Das Attribut ist mit dem Namen **TimeAxisActualize** dynamisierbar. Der Datentyp ist BOOLEAN.

TimeAxisAdd-Eigenschaft

Neu - TimeAxisAdd

Legt eine neue Zeitachse an.

Das Attribut ist mit dem Namen **TimeAxisAdd** dynamisierbar. Der Datentyp ist STRING.

TimeAxisAlign-Eigenschaft

Ausrichtung - TimeAxisAlign

Legt fest, wie die ausgewählte Zeitachse ausgerichtet wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	unten	Die ausgewählte Zeitachse wird unter der Kurve angezeigt.
1	oben	Die ausgewählte Zeitachse wird über der Kurve angezeigt.

Das Attribut ist mit dem Namen **TimeAxisAlign** dynamisierbar. Der Datentyp ist LONG.

TimeAxisBeginTime-Eigenschaft

Anfangszeitpunkt - TimeAxisBeginTime

Legt den Anfangszeitpunkt des Zeitbereichs für die ausgewählte Zeitachse fest.

Das Attribut ist mit dem Namen **TimeAxisBeginTime** dynamisierbar. Der Datentyp ist Date.

Wenn Sie den Zeitbereich dynamisieren, verwenden Sie das Format "jjjj-mm-dd hh:mm:ss".

TimeAxisColor-Eigenschaft

Farbe Zeitachse - TimeAxisColor

Gibt die Farbe der Zeitachse an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Einstellung ist wirksam, wenn die Option "in Kurvenfarbe" nicht aktiviert bzw. "TimeAxisInTrendColor" "FALSE" ist.

Das Attribut ist mit dem Namen **TimeAxisColor** dynamisierbar. Der Datentyp ist LONG.

TimeAxisCount-Eigenschaft

TimeAxisCount

Gibt die Anzahl der projektierten Zeitachsen an.

Das Attribut ist mit dem Namen **TimeAxisCount** dynamisierbar. Der Datentyp ist LONG.

TimeAxisDateFormat-Eigenschaft

Datumsformat - TimeAxisDateFormat

Legt fest, welches Datumsformat zur Darstellung der ausgewählten Zeitachse verwendet wird.

Folgende Datumsformate stehen zur Verfügung:

Wert	Erklärung
Automatisch	Das Datumsformat wird automatisch bestimmt.
dd.MM.yy	Tag.Monat.Jahr, z.B. 24.12.07.
dd.MM.yyyy	Tag.Monat.Jahr, z.B. 24.12.2007.
dd/MM/yy	Tag/Monat/Jahr, z.B. 24/12/07.
dd/MM/yyyy	Tag/Monat/Jahr, z.B. 24/12/2007.

Das Attribut ist mit dem Namen **TimeAxisDateFormat** dynamisierbar. Der Datentyp ist STRING.

TimeAxisEndTime-Eigenschaft

Endzeitpunkt - TimeAxisEndTime

Legt den Endzeitpunkt des Zeitbereichs für die ausgewählte Zeitachse fest.

Das Attribut ist mit dem Namen **TimeAxisEndTime** dynamisierbar. Der Datentyp ist Date.

Wenn Sie den Zeitbereich dynamisieren, verwenden Sie das Format "jjjj-mm-dd hh:mm:ss".

TimeAxisFormat-Eigenschaft

Beschreibung

Legt das Format der Angaben entlang der Zeitachse fest.

- 0: Die Angaben erfolgen in der Form hh:mm
- -1: Die Angaben erfolgen in der Form hh:mm:ss
- -2: Die Angaben erfolgen in der Form hh:mm:ss.ms
- -3: Die Angaben erfolgen in der Form hh:mm (Volle Stunden)
- -4: Die Angaben erfolgen in der Form hh:mm:ss (Volle Minuten)
- -5: Die Angaben erfolgen in der Form hh:mm:ss.ms (Volle Sekunden)

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

TimeAxisIndex-Eigenschaft

TimeAxisIndex

Referenziert eine projektierte Zeitachse. Unter Verwendung des Attributs können Sie einer bestimmten Zeitachse die Werte anderer Attribute zuweisen.

Gültige Werte für "TimeAxisIndex" liegen zwischen 0 und "TimeAxisCount" minus 1. Das Attribut "TimeAxisCount" gibt die Anzahl der projektierten Zeitachsen an.

Das Attribut "TimeAxisIndex" ist über das Attribut **TimeAxisRepos** dynamisierbar. Der Datentyp ist LONG.

TimeAxisInTrendColor-Eigenschaft

in Kurvenfarbe - TimeAxisInTrendColor

Legt fest, ob die ausgewählte Zeitachse in der Kurvenfarbe angezeigt wird. Wenn mehrere Kurven im Kurvenfenster angezeigt werden, wird die Farbe der ersten Kurve verwendet. Die Reihenfolge der Kurven legen Sie auf der Registerkarte "Kurven" fest.

Wert	Erklärung
TRUE	Die ausgewählte Zeitachse wird in der Kurvenfarbe angezeigt. Die Einstellung im Feld "Farbe" bzw. "TimeAxisColor" ist unwirksam.
FALSE	Die ausgewählte Zeitachse wird in der Farbe angezeigt, die im Feld "Farbe" bzw. "TimeAxisColor" eingestellt ist.

Das Attribut ist mit dem Namen **TimeAxisInTrendColor** dynamisierbar. Der Datentyp ist BOOLEAN.

TimeAxisLabel-Eigenschaft

Beschriftung - TimeAxisLabel

Legt den Text fest, mit dem die Zeitachse beschriftet wird.

Das Attribut ist mit dem Namen **TimeAxisLabel** dynamisierbar. Der Datentyp ist STRING.

TimeAxisMeasurePoints-Eigenschaft

Anzahl der Messpunkte - TimeAxisMeasurePoints

Legt die Anzahl der Messpunkte fest, die für die ausgewählte Zeitachse angezeigt werden.

Das Attribut ist mit dem Namen **TimeAxisMeasurePoints** dynamisierbar. Der Datentyp ist LONG.

TimeAxisName-Eigenschaft

Objektname - TimeAxisName

Legt den Namen der ausgewählten Zeitachse fest.

Das Attribut "TimeAxisName" ist über das Attribut **TimeAxisRename** dynamisierbar. Der Datentyp ist STRING.

TimeAxisRangeType-Eigenschaft

Einstellung Zeitbereich - TimeAxisRangeType

Legt die Einstellung für den Zeitbereich fest, der für die ausgewählte Zeitachse verwendet wird.

Wert	Beschreibung	Erklärung
0	Zeitbereich	Für die Zeitachse werden Anfangszeitpunkt und Zeitbereich festgelegt.
1	Anfangs- bis Endzeitpunkt	Für die Zeitachse werden Anfangs- und Endzeitpunkt festgelegt.
2	Anzahl der Messpunkte	Für die Zeitachse werden Anfangszeitpunkt und Anzahl der Messpunkte festgelegt.

Das Attribut ist mit dem Namen **TimeAxisRangeType** dynamisierbar. Der Datentyp ist LONG.

TimeAxisRemove-Eigenschaft

Entfernen - TimeAxisRemove

Entfernt die ausgewählte Zeitachse aus der Liste.

Das Attribut ist mit dem Namen **TimeAxisRemove** dynamisierbar. Der Datentyp ist STRING.

TimeAxisRename-Eigenschaft

TimeAxisRename

Ändert den Namen der Zeitachse, die über das Attribut "TimeAxisIndex" referenziert wird.

Das Attribut ist mit dem Namen **TimeAxisRename** dynamisierbar. Mit "TimeAxisRename" dynamisieren Sie auch das Attribut "TimeAxisName". Der Datentyp ist STRING.

TimeAxisRepos-Eigenschaft

Auf/Ab - TimeAxisRepos

Ändert die Reihenfolge der Zeitachsen. "Auf" und "Ab" bewegen die ausgewählte Zeitachse in der Liste nach oben oder unten.

Die Reihenfolge in der Liste bestimmt in Runtime die Position der Zeitachse im Kurvenfenster. Wenn die Ausrichtung gleich ist und die Zeitachse weiter oben steht, wird die Zeitachse an einer kurvenferneren Position dargestellt.

Das Attribut ist mit dem Namen **TimeAxisRepos** dynamisierbar. Der Datentyp ist LONG.

TimeAxisShowDate-Eigenschaft

Datum anzeigen - TimeAxisShowDate

Legt fest, ob die ausgewählte Zeitachse mit Datum und Uhrzeit angezeigt wird.

Wert	Erklärung
TRUE	Datum und Uhrzeit werden angezeigt. Das Datumsformat wird im Feld "Datumsformat" festgelegt.
FALSE	Das Datum wird nicht angezeigt. Nur die Uhrzeit wird angezeigt.

Das Attribut ist mit dem Namen **TimeAxisShowDate** dynamisierbar. Der Datentyp ist BOOLEAN.

TimeAxisTimeFormat-Eigenschaft

Zeitformat - TimeAxisTimeFormat

Legt fest, welches Zeitformat zur Darstellung der ausgewählten Zeitachse verwendet wird.

Folgende Zeitformate stehen zur Verfügung:

Wert	Erklärung
Automatisch	Das Zeitformat wird automatisch bestimmt.
hh:mm:ss.ms	Stunden:Minuten:Sekunden, z.B. 15:35:44.240.
hh:mm:ss tt	Stunden:Minuten:Sekunden AM/PM, z.B. 03:35:44 PM.
hh:mm:ss.ms tt	Stunden:Minuten:Sekunden.Millisekunden AM/PM, z.B. 03:35:44.240 PM.

Das Attribut ist mit dem Namen **TimeAxisTimeFormat** dynamisierbar. Der Datentyp ist STRING.

TimeAxisTimeRangeBase-Eigenschaft

Zeitbereich - TimeAxisTimeRangeBase

Legt die Zeiteinheit zur Bestimmung des Zeitbereichs fest.

Folgende Zeiteinheiten stehen zur Verfügung:

Wert	Beschreibung
500	500 ms
1000	1 Sekunde
60000	1 Minute
3600000	1 Stunde
86400000	1 Tag

Das Attribut ist mit dem Namen **TimeAxisTimeRangeBase** dynamisierbar. Der Datentyp ist LONG.

TimeAxisTimeRangeFactor-Eigenschaft

Zeitbereich - TimeAxisTimeRangeFactor

Legt den Faktor zur Bestimmung des Zeitbereichs fest. Nur ganzzahlige Faktoren sind zulässig.

Das Attribut ist mit dem Namen **TimeAxisTimeRangeFactor** dynamisierbar. Der Datentyp ist SHORT.

TimeAxisTrendWindow-Eigenschaft

Kurvenfenster - TimeAxisTrendWindow

Legt fest, in welchem Kurvenfenster die ausgewählte Zeitachse verwendet wird. Die zur Verfügung stehenden Kurvenfenster legen Sie auf der Registerkarte "Kurvenfenster" bzw. über "TrendWindowAdd" fest.

Das Attribut ist mit dem Namen **TimeAxisTrendWindow** dynamisierbar. Der Datentyp ist STRING.

TImeAxisVisible-Eigenschaft

Zeitachse - TimeAxisVisible

In der Liste werden die Zeitachsen aufgelistet, die Sie angelegt haben. Klicken Sie auf eine Zeitachse in der Liste, um die Eigenschaften anzupassen und um ein Kurvenfenster der Zeitachse zuzuordnen.

Aktivieren Sie in der Liste die Zeitachsen, die Sie in den Kurvenfenstern anzeigen wollen.

Legt fest, ob die ausgewählte Zeitachse angezeigt wird.

Wert	Erklärung
TRUE	Die Zeitachse wird angezeigt.
FALSE	Die Zeitachse wird nicht angezeigt.

Das Attribut ist mit dem Namen **TimeAxisVisible** dynamisierbar. Der Datentyp ist BOOLEAN.

TimeAxisX-Eigenschaft

Beschreibung

TRUE, wenn im Kurvenfenster für alle Kurven eine gemeinsame X-Achse verwendet wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

TimeBase-Eigenschaft

Zeitbasis - TimeBase

In diesem Auswahlfeld wird die Zeitbasis festgelegt, die für die Zeitangaben im Control gilt.

Wert	Bezeichnung
0	Lokale Zeitzone
1	Koordinierte Weltzeit (UTC)
2	Projekteinstellung

Das Attribut ist mit dem Namen **TimeBase** dynamisierbar. Der Datentyp ist LONG.

TimeColumn

TimeColumnActualize-Eigenschaft

TimeColumnActualize

Legt fest, ob die Werte der ausgewählten Spalte aktualisiert werden.

Wert	Erklärung
TRUE	Die Zeitspalte wird aktualisiert.
FALSE	Die Zeitspalte wird nicht aktualisiert. Diese Einstellung ist sinnvoll, wenn eine Tabelle mit einer andere Tabelle verglichen wird.

Das Attribut ist mit dem Namen **TimeColumnActualize** dynamisierbar. Der Datentyp ist BOOLEAN.

TimeColumnAdd-Eigenschaft

Neu - TimeColumnAdd

Legt eine neue Zeitspalte an.

Das Attribut ist mit dem Namen **TimeColumnAdd** dynamisierbar. Der Datentyp ist STRING.

TimeColumnAlign-Eigenschaft

Ausrichtung - TimeColumnAlign

Legt fest, wie die ausgewählte Zeitspalte ausgerichtet wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	links	Die ausgewählte Zeitspalte wird links angezeigt.
1	zentriert	Die ausgewählte Zeitspalte wird zentriert angezeigt.
2	rechts	Die ausgewählte Zeitspalte wird rechts angezeigt.

Das Attribut ist mit dem Namen **TimeColumnAlign** dynamisierbar. Der Datentyp ist LONG.

TimeColumnAlignment-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert ein Spaltenpaar. "TimeColumnAlignment" legt die Ausrichtung der Zeitspalte dieses Spaltenpaares fest.

- 0: Zeitwerte werden linksbündig eingetragen.
- 1: Zeitwerte werden zentriert eingetragen
- 2: Zeitwerte werden rechtsbündig eingetragen.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

TimeColumnBackColor-Eigenschaft

Hintergrundfarbe - TimeColumnBackColor

Gibt die Hintergrundfarbe der ausgewählten Zeitspalte an. Mit der Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Einstellung ist wirksam:

- Wenn die Option "in den Farben der Wertspalte" nicht aktiviert bzw. "TimeColumnUseValueColumnColors" "FALSE" ist.
- Wenn auf der Registerkarte "Allgemeines" im Bereich "Spaltenfarbe verwenden" die Option "Hintergrundfarbe" aktiviert bzw. "UseColumnBackColor" "TRUE" ist.

Das Attribut ist mit dem Namen **TimeColumnBackColor** dynamisierbar. Der Datentyp ist LONG.

TimeColumnBeginTime-Eigenschaft

Anfangszeitpunkt - TimeColumnBeginTime

Legt den Anfangszeitpunkt des Zeitbereichs für die ausgewählte Zeitspalte fest.

Das Attribut ist mit dem Namen **TimeColumnBeginTime** dynamisierbar. Der Datentyp ist Date.

Wenn Sie den Zeitbereich dynamisieren, verwenden Sie das Format "jjjj-mm-dd hh:mm:ss".

TimeColumnCaption-Eigenschaft

Bezeichnung - TimeColumnCaption

Legt die Bezeichnung der Zeitspalte fest.

Das Attribut ist mit dem Namen **TimeColumnCaption** dynamisierbar. Der Datentyp ist STRING.

TimeColumnCount-Eigenschaft

TimeColumnCount

Gibt die Anzahl der projektierten Zeitspalten an.

Das Attribut ist mit dem Namen **TimeColumnCount** dynamisierbar. Der Datentyp ist LONG.

TimeColumnDateFormat-Eigenschaft

Datumsformat - TimeColumnDateFormat

Legt fest, welches Datumsformat zur Darstellung der ausgewählten Zeitspalte verwendet wird.

Folgende Datumsformate stehen zur Verfügung:

Wert	Erklärung
Automatisch	Das Datumsformat wird automatisch bestimmt.
dd.MM.yy	Tag.Monat.Jahr, z.B. 24.12.07.
dd.MM.yyyy	Tag.Monat.Jahr, z.B. 24.12.2007.
dd/MM/yy	Tag/Monat/Jahr, z.B. 24/12/07.
dd/MM/yyyy	Tag/Monat/Jahr, z.B. 24/12/2007.

Das Attribut ist mit dem Namen **TimeColumnDateFormat** dynamisierbar. Der Datentyp ist STRING.

TimeColumnEndTime-Eigenschaft

Endzeitpunkt - TimeColumnEndTime

Legt den Endzeitpunkt des Zeitbereichs für die ausgewählte Zeitspalte fest.

Das Attribut ist mit dem Namen **TimeColumnEndTime** dynamisierbar. Der Datentyp ist Date.

Wenn Sie den Zeitbereich dynamisieren, verwenden Sie das Format "jjjj-mm-dd hh:mm:ss".

TimeColumnForeColor-Eigenschaft

Schriftfarbe - TimeColumnForeColor

Gibt die Schriftfarbe der ausgewählten Zeitspalte an. Mit der Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Einstellung ist wirksam:

- Wenn die Option "in den Farben der Wertspalte" nicht aktiviert bzw. "TimeColumnUseValueColumnColors" "FALSE" ist.
- Wenn auf der Registerkarte "Allgemeines" im Bereich "Spaltenfarbe verwenden" die Option "Schriftfarbe" aktiviert bzw. "UseColumnForeColor" "TRUE" ist.

Das Attribut ist mit dem Namen **TimeColumnForeColor** dynamisierbar. Der Datentyp ist LONG.

TimeColumnHideText-Eigenschaft

TimeColumnHideText

Legt fest, ob der Inhalt der Zeitspalte als Text angezeigt wird.

Wert	Erklärung
TRUE	Der Inhalt wird nicht als Text angezeigt.
FALSE	Der Inhalt wird als Text angezeigt.

Das Attribut ist mit dem Namen **TimeColumnHideText** dynamisierbar. Der Datentyp ist BOOLEAN.

TimeColumnHideTitleText-Eigenschaft

TimeColumnHideTitleText

Legt fest, ob die Überschrift der Zeitspalte als Text angezeigt wird.

Wert	Erklärung
TRUE	Die Überschrift wird nicht als Text angezeigt.
FALSE	Die Überschrift wird als Text angezeigt.

Das Attribut ist mit dem Namen **TimeColumnHideTitleText** dynamisierbar. Der Datentyp ist BOOLEAN.

TimeColumnIndex-Eigenschaft

TimeColumnIndex

Referenziert eine projektierte Zeitspalte. Unter Verwendung des Attributs können Sie einer bestimmten Zeitspalte die Werte anderer Attribute zuweisen.

Gültige Werte für "TimeColumnIndex" liegen zwischen 0 und "TimeColumnCount" minus 1. Das Attribut "TimeColumnCount" gibt die Anzahl der projektierten Zeitspalten an.

Das Attribut "TimeColumnIndex" ist über das Attribut **TimeColumnRepos** dynamisierbar. Der Datentyp ist LONG.

TimeColumnLength-Eigenschaft

Länge in Zeichen - TimeColumnLength

Legt die Breite für die ausgewählte Zeitspalte fest.

Das Attribut ist mit dem Namen **TimeColumnLength** dynamisierbar. Der Datentyp ist LONG.

TimeColumnMeasurePoints-Eigneschaft

Anzahl der Messpunkte - TimeColumnMeasurePoints

Legt die Anzahl der Messpunkte fest, die in der ausgewählten Zeitspalte angezeigt werden.

Das Attribut ist mit dem Namen **TimeColumnMeasurePoints** dynamisierbar. Der Datentyp ist LONG.

TimeColumnName-Eigenschaft

Objektname - TimeColumnName

Legt den Namen der ausgewählten Zeitspalte fest.

Das Attribut "TimeColumnName" ist über das Attribut **TimeColumnRename** dynamisierbar. Der Datentyp ist STRING.

TimeColumnRangeType-Eigenschaft

Einstellung Zeitbereich - TimeColumnRangeType

Legt den Zeitbereich fest, der für die ausgewählte Zeitspalte verwendet wird.

Wert	Beschreibung	Erklärung
0	Zeitbereich	Für die Zeitspalte werden Anfangszeitpunkt und Zeitbereich festgelegt.
1	Anfangs- bis Endzeitpunkt	Für die Zeitspalte werden Anfangs- und Endzeitpunkt festgelegt.
2	Anzahl der Messpunkte	Für die Zeitspalte werden Anfangszeitpunkt und Anzahl der Messpunkte festgelegt.

Das Attribut ist mit dem Namen **TimeColumnRangeType** dynamisierbar. Der Datentyp ist LONG.

TimeColumnRemove-Eigenschaft

Entfernen - TimeColumnRemove

Entfernt die ausgewählte Zeitspalte aus der Liste.

Das Attribut ist mit dem Namen **TimeColumnRemove** dynamisierbar. Der Datentyp ist STRING.

TimeColumnRename-Eigenschaft

TimeColumnRename

Ändert den Namen der Zeitspalte, die über das Attribut "TimeColumnIndex" referenziert wird.

Das Attribut ist mit dem Namen **TimeColumnRename** dynamisierbar. Mit "TimeColumnRename" dynamisieren Sie auch das Attribut "TimeColumnName". Der Datentyp ist STRING.

TimeColumnRepos-Eigenschaft

Auf/Ab - TimeColumnRepos

Ändert die Reihenfolge der Zeitspalten mit den zugehörigen Wertspalten. "Auf" und "Ab" bewegen die ausgewählte Zeitspalte in der Liste nach oben oder unten. Dadurch wird die Zeitspalte mit den zugehörigen Wertspalten in der Tabelle weiter vorne oder hinten platziert.

Das Attribut ist mit dem Namen **TimeColumnRepos** dynamisierbar. Der Datentyp ist LONG.

TimeColumnShowDate-Eigenschaft

Datum anzeigen - TimeColumnShowDate

Legt fest, ob die ausgewählte Zeitspalte mit Datum und Uhrzeit angezeigt wird.

Wert	Erklärung
TRUE	Datum und Uhrzeit werden angezeigt. Das Datumsformat wird im Feld "Datumsformat" bzw. über "TimeColumnDateFormat" festgelegt.
FALSE	Das Datum wird nicht angezeigt. Nur die Uhrzeit wird angezeigt.

Das Attribut ist mit dem Namen **TimeColumnShowDate** dynamisierbar. Der Datentyp ist BOOLEAN.

TimeColumnShowIcon-Eigenschaft

TimeColumnShowIcon

Legt fest, ob der Inhalt der Zeitspalte als Symbol angezeigt wird.

Wert	Erklärung
TRUE	Der Inhalt wird als Symbol angezeigt.
FALSE	Der Inhalt wird nicht als Symbol angezeigt.

Das Attribut ist mit dem Namen **TimeColumnShowIcon** dynamisierbar. Der Datentyp ist BOOLEAN.

TimeColumnShowTitleIcon-Eigenschaft

TimeColumnShowTitleIcon

Legt fest, ob die Überschrift der Zeitspalte als Symbol angezeigt wird.

Wert	Erklärung
TRUE	Die Überschrift wird als Symbol angezeigt.
FALSE	Die Überschrift wird nicht als Symbol angezeigt.

Das Attribut ist mit dem Namen **TimeColumnShowTitleIcon** dynamisierbar. Der Datentyp ist BOOLEAN.

TimeColumnSort-Eigenschaft

TimeColumnSort

Legt fest, wie die im "TimeColumnIndex" referenzierte Zeitspalte sortiert wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	nein	Keine Sortierung
1	aufsteigend	Aufsteigende Sortierung vom kleinsten zum größten Wert.
2	absteigend	Absteigende Sortierung vom größten zum kleinsten Wert.

Das Attribut ist mit dem Namen **TimeColumnSort** dynamisierbar. Der Datentyp ist LONG.

TimeColumnSortIndex-Eigenschaft

TimeColumnSortIndex

Gibt die Sortierreihenfolge der im "TimeColumnIndex" referenzierten Zeitspalte an. Wenn Sie den Wert auf "0" setzen, wird das Sortierkriterium in "TimeColumnSort" entfernt.

Das Attribut ist mit dem Namen **TimeColumnSortIndex** dynamisierbar. Der Datentyp ist LONG.

TimeColumnTimeFormat-Eigenschaft

Zeitformat - TimeColumnTimeFormat

Legt fest, welches Zeitformat zur Darstellung der ausgewählten Zeitspalte verwendet wird.

Folgende Zeitformate stehen zur Verfügung:

Wert	Erklärung
Automatisch	Das Zeitformat wird automatisch bestimmt.
HH:mm:ss.ms	Stunden:Minuten:Sekunden, z.B. 15:35:44.240.
hh:mm:ss tt	Stunden:Minuten:Sekunden AM/PM, z.B. 03:35:44 PM.
hh:mm:ss.ms tt	Stunden:Minuten:Sekunden.Millisekunden AM/PM, z.B. 03:35:44.240 PM.

Das Attribut ist mit dem Namen **TimeColumnTimeFormat** dynamisierbar. Der Datentyp ist STRING.

TimeColumnTimeRangeBase-Eigenschaft

Zeitbereich - TimeColumnTimeRangeBase

Legt die Zeiteinheit zur Bestimmung des Zeitbereichs fest.

Folgende Zeiteinheiten stehen zur Verfügung:

Wert	Beschreibung
500	500 ms
1000	1 Sekunde
60000	1 Minute
3600000	1 Stunde
86400000	1 Tag

Das Attribut ist mit dem Namen **TimeColumnTimeRangeBase** dynamisierbar. Der Datentyp ist LONG.

TimeColumnTimeRangeFactor-Eigenschaft

Zeitbereich - TimeColumnTimeRangeFactor

Legt den Faktor zur Bestimmung des Zeitbereichs fest. Nur ganzzahlige Faktoren sind zulässig.

Das Attribut ist mit dem Namen **TimeColumnTimeRangeFactor** dynamisierbar. Der Datentyp ist SHORT.

TimeColumnUseValueColumnColors-Eigenschaft

in den Farben der Wertspalte - TimeColumnUseValueColumnColors

Legt fest, ob die ausgewählte Zeitspalte in den Farben der Wertspalte angezeigt wird.

Wert	Erklärung
TRUE	Die ausgewählte Zeitspalte wird in den Farben der Wertspalte angezeigt. Die Einstellungen in den Feldern "Schriftfarbe" und "Hintergrundfarbe" sind unwirksam.
FALSE	Die ausgewählte Zeitspalte wird in den Farben angezeigt, die in den Feldern "Schriftfarbe" und "Hintergrundfarbe" festgelegt sind.

Das Attribut ist mit dem Namen **TimeColumnUseValueColumnColors** dynamisierbar. Der Datentyp ist BOOLEAN.

TimeColumnVisible-Eigenschaft

Zeitspalten - TimeColumnVisible

In der Liste werden die Zeitspalten aufgelistet, die Sie angelegt haben. Klicken Sie auf eine Zeitspalte in der Liste, um die Eigenschaften anzupassen und um den Zeitbereich der Zeitspalte festzulegen.

Aktivieren Sie in der Liste die Zeitspalten, die Sie in der Tabelle anzeigen wollen.

Legt fest, ob die ausgewählte Zeitspalte angezeigt wird.

Das Attribut ist mit dem Namen **TimeColumnVisible** dynamisierbar. Der Datentyp ist BOOLEAN.

TimeFormat - Tolerance

TimeFormat-Eigenschaft

Beschreibung

Legt das Format der Zeitangaben fest.

- 0: Die Angaben erfolgen in der Form hh:mm
- -1: Die Angaben erfolgen in der Form hh:mm:ss
- -2: Die Angaben erfolgen in der Form hh:mm:ss.ms
- -3: Die Angaben erfolgen in der Form hh:mm (Volle Stunden)
- -4: Die Anzgaben erfolgen in der Form hh:mm:ss (Volle Minuten)
- -5: Die Angaben erfolgen in der Form hh:mm:ss.ms (Volle Sekunden)

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

TimeJump-Eigenschaft

Beschreibung

WinCC Online Trend Control

Die Eigenschaft "Index" referenziert eine Kurve. "TimeJump" legt fest, ob die im Archiv vorhandenen Zeitsprünge mit der in "TimeJumpColor" festgelegten Farbe gekennzeichnet werden sollen.

WinCC Online Table Control

Der Wert dieses Attributs kann nicht verändert werden. Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

TimeJumpColor-Eigenschaft

Beschreibung

WinCC Online Trend Control

Die Eigenschaft "Index" referenziert eine Kurve. "TimeJumpColor" legt die Farbe fest, die zur Kennzeichnung der im Archiv vorhandenen Zeitsprünge verwendet wird. Ob die Angabe ausgewertet wird ist abhängig von der Eigenschaft "TimeJump". Die Angabe der Farbe erfolgt als RGB-Wert. LONG Schreib-Lese-Zugriff.

WinCC Online Table Control

Der Wert dieser Eigenschaft kann nicht verändert werden. Lese-Zugriff.

Siehe auch

ScreenItem-Objekt (Seite 134)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

TimeOverlap-Eigenschaft

Beschreibung

WinCC Online Trend Control

Die Eigenschaft "Index" referenziert eine Kurve. "TimeOverlap" legt fest, ob die im Archiv vorhandenen Zeitüberlappungen mit der in "TimeOverlapColor" festgelegten Farbe gekennzeichnet werden sollen.

WinCC Online Table Control

Der Wert dieser Eigenschaft kann nicht verändert werden. Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

TimeOverlapColor-Eigenschaft

Beschreibung

WinCC Online Trend Control

Die Eigenschaft "Index" referenziert eine Kurve. "TimeOverlapColor" legt die Farbe fest, die zur Kennzeichnung der im Archiv vorhandenen Zeitüberlappungen verwendet wird. Ob die Angabe ausgewertet wird ist abhängig vom Attribut "TimeOverlap". Die Angabe der Farbe erfolgt als RGB-Wert.

WinCC Online Table Control

Der Wert dieser Eigenschaft kann nicht verändert werden. Lese-Zugriff.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

ScreenItem-Objekt (Seite 134)

TimeRange-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert ein Spaltenpaar bzw. eine Kurve. "TimeRange" legt fest, wie der darzustellende Zeitbereich definiert ist.

- 0: Der darzustellende Zeitbereich ist durch einen Startzeitpunkt ("BeginTime") und einen Endzeitpunkt ("EndTime") festgelegt.
- -1: Der darzustellende Zeitbereich ist durch einen Startzeitpunkt ("BeginTime") und einen Zeitbereich ("TimeRangeBase" und "TimeRangeFactor") festgelegt.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

TimeRangeBase-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert ein Spaltenpaar bzw. eine Kurve. Der für dieses Spaltenpaar/diese Kurve darzustellende Zeitbereich ergibt sich aus der Multiplikation der Werte "TimeRangeBase" und "TimeRangeFactor", wobei der Wert von "TimeRangeBase" in Millisekunden interpretiert wird.

Die Eigenschaften "TimeRangeBase" und "TimeRangeFactor" werden nur ausgewertet, wenn die Eigenschaft "TimeRange" gesetzt ist, also den Wert "-1" besitzt.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

TimeRangeFactor-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert eine Kurve bzw. ein Spaltenpaar. Der für diese Kurve/ dieses Spaltenpaar darzustellende Zeitbereich ergibt sich aus der Multiplikation der Werte "TimeRangeBase" und "TimeRangeFactor", wobei der Wert von "TimeRangeBase" in Millisekunden interpretiert wird.

Die Eigenschaften "TimeRangeBase" und "TimeRangeFactor" werden nur ausgewertet, wenn die Eigenschaft "TimeRange" den Wert "-1" besitzt.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

TimeStamp-Eigenschaft

Beschreibung

Liest den Zeitstempel des letzten Lesezugriffs einer Variablen. Der Zeitstempel wird in lokaler Zeit zurückgeliefert. DATE (readonly)

Mit der VBS-Standardfunktion "FormatDateTime(Date[, NamedFormat])" kann die TimeStamp-Eigenschaft in Klartext ausgegeben werden. Die Ausgabe ist abhängig von der aktuellen Spracheinstellung. Die Spracheinstellung kann über die VBS-Standardfunktion SetLocale() gesetzt werden.

Durch den zweiten Parameter der Funktion FormatDate() und durch weitere VBS-Standardfunktionen wie Year, WeekDay, Day, Hour, Minute, Second können die Angaben, wie vom Anwender gewünscht, aufgesplittet werden. Möchte man zum WeekDay den Namen des Wochentages erhalten, so muss man die Funktion WeekdayName verwenden.

Beispiel:

```
'VBS87
Dim objTag
Dim lngCount
lngCount = 0
Set objTag = HMIRuntime.Tags("Tag11")
objTag.Read
SetLocale("en-gb")
MsgBox FormatDateTime(objTag.TimeStamp)    'Output: e.g. 06/08/2002 9:07:50
MsgBox Year(objTag.TimeStamp)    'Output: e.g. 2002
MsgBox Month(objTag.TimeStamp)    'Output: e.g. 8
MsgBox Weekday(objTag.TimeStamp)    'Output: e.g. 3
MsgBox WeekdayName(Weekday(objTag.TimeStamp))    'Output: e.g. Tuesday
MsgBox Day(objTag.TimeStamp)    'Output: e.g. 6
MsgBox Hour(objTag.TimeStamp)    'Output: e.g. 9
MsgBox Minute(objTag.TimeStamp)    'Output: e.g. 7
MsgBox Second(objTag.TimeStamp)    'Output: e.g. 50
For lngCount = 0 To 4
MsgBox FormatDateTime(objTag.TimeStamp, lngCount)
Next
'lngCount = 0: Output: e.g. 06/08/2002 9:07:50
'lngCount = 1: Output: e.g. 06 August 2002
'lngCount = 2: Output: e.g. 06/08/2002
'lngCount = 3: Output: e.g. 9:07:50
'lngCount = 4: Output: e.g. 9:07
```

Beispiel

Das folgende Beispiel gibt den Zeitstempel der Variablen "Tag1" aus:

```
'VBS88
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
objTag.Read
MsgBox objTag.TimeStamp
```

Siehe auch

Tag-Objekt (Seite 146)

Alarms-Objekt (Auflistung) (Seite 120)

TimeStepBase-Eigenschaft

Genauigkeit - TimeStepBase

Legt die Genauigkeit des Zeitstempels fest, der in der Tabelle dargestellt wird.

Sie bestimmen die Genauigkeit durch Multiplizieren von Faktor und Zeiteinheit. Wenn Sie z. B. alle Werte, die innerhalb von 3 Sekunden aufgetreten sind, in der gleichen Zeile darstellen wollen, geben Sie für den Faktor "3" und für die Zeiteinheit "1 s" ein.

Wert	Beschreibung	Erklärung
0	exakt	In der Tabelle werden nur Werte mit exakt gleichem Zeitstempel innerhalb einer Zeile dargestellt.
100	100 ms	In der Tabelle werden alle Werte, die innerhalb von 100 Millisekunden liegen, zu einer Zeile zusammengefasst.
250	250 ms	In der Tabelle werden alle Werte, die innerhalb von 250 Millisekunden liegen, zu einer Zeile zusammengefasst.
500	500 ms	In der Tabelle werden alle Werte, die innerhalb von 500 Millisekunden liegen, zu einer Zeile zusammengefasst.
1000	1 s	In der Tabelle werden alle Werte, die innerhalb von einer Sekunde liegen, zu einer Zeile zusammengefasst.

Das Attribut ist mit dem Namen **TimeStepBase** dynamisierbar. Der Datentyp ist LONG.

TimeStepFactor-Eigenschaft

Genauigkeit - TimeStepFactor

Legt die Genauigkeit des Zeitstempels fest, der in der Tabelle dargestellt wird.

Sie bestimmen die Genauigkeit durch Multiplizieren von Faktor und Zeiteinheit. Wenn Sie z.B. alle Werte, die innerhalb von 3 Sekunden aufgetreten sind, in der gleichen Zeile darstellen wollen, geben Sie für den Faktor "3" und für die Zeiteinheit "1 s" ein.

Der eingegebene Faktor ist unwirksam, wenn für die Zeiteinheit "exakt" bzw. für "TimeStepBase" der Wert "0" gewählt ist.

Das Attribut ist mit dem Namen **TimeStepFactor** dynamisierbar. Der Datentyp ist LONG.

TimeZone-Eigenschaft

Beschreibung

Legt die Zeitzone fest, auf deren Basis die Zeitwerte angezeigt werden. Es sind 4 Einstellungen möglich:

- Lokale Zeitzone
- Zeitzone des Servers
- UTC (Universal Time Coordinated)
- Projekteinstellungen übernehmen (=> Über den WinCC-Explorer bei der Eigenschaftsseite des Rechners kann der Uhrzeitmodus rechnerspezifisch festgelegt werden. Als Auswahl wird angeboten: WinCC V50 (Kompatibilitätsmodus => Darstellung wie es in den einzelnen Anzeigeteilen bis zur V5 üblich war), Lokale Zeit und UTC.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

TitleColor-Eigenschaft

Hintergrund Tabellenüberschrift - TitleColor

Gibt die Hintergrundfarbe der Tabellenüberschrift an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **TitleColor** dynamisierbar. Der Datentyp ist LONG.

TitleCut-Eigenschaft

Inhalte abkürzen - TitleCut

Legt fest, ob die Spaltenüberschriften abgekürzt werden, wenn die Spalten nicht breit genug sind.

Wert	Erklärung
TRUE	Die Spaltenüberschriften werden abgekürzt.
FALSE	Die Spaltenüberschriften werden nicht abgekürzt.

Das Attribut ist mit dem Namen **TitleCut** dynamisierbar. Der Datentyp ist BOOLEAN.

TitleCut-Eigenschaft (vor WinCC V7)

Beschreibung

Legt fest, ob die Inhalte der Felder einer Titelleiste bei zu kleiner Spaltenbreite abgekürzt werden sollen. Schreib-Lese-Zugriff.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

TitleDarkShadowColor-Eigenschaft

Schattierungsfarbe dunkel - TitleDarkShadowColor

Gibt die Farbe für die dunkle Seite der Schattierung an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Einstellung ist nur wirksam, wenn die Option "Schattierungsfarbe" bzw. "TitleStyle" aktiviert ist.

Das Attribut ist mit dem Namen **TitleDarkShadowColor** dynamisierbar. Der Datentyp ist LONG.

TitleForeColor-Eigenschaft

Schriftfarbe Tabellenüberschrift - TitleForeColor

Gibt die Schriftfarbe der Tabellenüberschrift an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **TitleForeColor** dynamisierbar. Der Datentyp ist LONG.

TitleGridLineColor-Eigenschaft

Farbe der Trennlinie / Überschrift - TitleGridLineColor

Gibt die Farbe der Trennlinien in der Tabellenüberschrift an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **TitleGridLineColor** dynamisierbar. Der Datentyp ist LONG.

TitleLightShadowColor-Eigenschaft

Schattierungsfarbe hell - TitleLightShadowColor

Gibt die Farbe für die helle Seite der Schattierung an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Einstellung ist nur wirksam, wenn die Option "Schattierungsfarbe" bzw. "TitleStyle" aktiviert ist.

Das Attribut ist mit dem Namen **TitleLightShadowColor** dynamisierbar. Der Datentyp ist LONG.

Titleline-Eigenschaft

Beschreibung

TRUE, wenn das Control eine Titelleiste besitzt und im Runtime verschoben werden kann. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

TitleSort-Eigenschaft

Sortieren über Spaltenüberschrift - TitleSort

Legt fest, wie das Sortieren über die Spaltenüberschrift ausgelöst wird. Um über die Spaltenüberschrift sortieren zu können, muss die Option "Auto Scrolling" deaktiviert sein.

Wert	Beschreibung	Erklärung
0	nein	Das Sortieren über die Spaltenüberschrift ist nicht möglich.
1	mit Klick	Das Sortieren wird durch einen Klick auf die Spaltenüberschrift ausgelöst.
2	mit Doppelklick	Das Sortieren wird durch einen Doppelklick auf die Spaltenüberschrift ausgelöst.

Das Attribut ist mit dem Namen **TitleSort** dynamisierbar. Der Datentyp ist LONG.

TitleStyle-Eigenschaft

Schattierungsfarbe - TitleStyle

Legt fest, ob eine Schattierungsfarbe für die Tabellenüberschrift verwendet wird.

Wert	Beschreibung	Erklärung
0	Flach	Eine Schattierungsfarbe wird nicht verwendet. Flach wirkende Darstellung der Überschrift.
1	Button	Eine Schattierungsfarbe wird verwendet. Räumlich wirkende Darstellung der Überschrift.

Das Attribut ist mit dem Namen **TitleStyle** dynamisierbar. Der Datentyp ist LONG.

Toggle-Eigenschaft

Beschreibung

TRUE, wenn der Button oder Rundbutton in Runtime nach dem Betätigen einrasten soll.
BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Rundbutton (Seite 219)

ScreenItem-Objekt (Seite 134)

ToleranceHigh-Eigenschaft

Beschreibung

Legt den Grenzwert für "Toleranz high" fest oder gibt ihn zurück.
Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft TypeToleranceHigh fest.

Die Überwachung des Grenzwerts ist nur wirksam, wenn die Eigenschaft CheckToleranceHigh auf "True" gesetzt ist.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

ToleranceLow-Eigenschaft

Beschreibung

Legt den Grenzwert für "Toleranz low" fest oder gibt ihn zurück.
Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft TypeToleranceLow fest.
Die Überwachung des Grenzwerts ist nur wirksam, wenn die Eigenschaft CheckToleranceLow auf "True" gesetzt ist.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

Toolbar

Toolbar-Eigenschaft

Beschreibung

TRUE, wenn eine Symbolleiste angezeigt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

ToolbarAlignment-Eigenschaft (vor WinCC V7)

Beschreibung

Legt die Position der Symbolleiste fest oder gibt sie zurück. Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

ToolbarAlignment-Eigenschaft

Ausrichtung - ToolbarAlignment

Legt die Ausrichtung der Symbolleiste im Control fest.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	oben	Die Symbolleiste wird am oberen Rand angezeigt.
1	unten	Die Symbolleiste wird am unteren Rand angezeigt.
2	links	Die Symbolleiste wird am linken Rand angezeigt.
3	rechts	Die Symbolleiste wird am rechten Rand angezeigt.

Das Attribut ist mit dem Namen **ToolbarAlignment** dynamisierbar. Der Datentyp ist LONG.

ToolbarBackColor-Eigenschaft

Hintergrundfarbe - ToolbarBackColor

Gibt die Hintergrundfarbe der Symbolleiste an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die projektierte Hintergrundfarbe wird nur angezeigt, wenn die Option "anzeigen" aktiviert bzw. "ToolbarUseBackColor" "TRUE" ist.

Das Attribut ist mit dem Namen **ToolbarBackColor** dynamisierbar. Der Datentyp ist LONG.

ToolbarButtonActive-Eigenschaft

Aktiv - ToolbarButtonActive

Legt fest, ob die mit der Taste verbundene Funktion in Runtime aktiv geschaltet ist. Ein Klick auf die Taste in Runtime löst die zugehörige Funktion aus.

Wert	Erklärung
TRUE	Die mit der Taste verbundene Funktion ist aktiv.
FALSE	Die mit der Taste verbundene Funktion ist nicht aktiv. Sie können eine eigene Funktion über Skript mit der Taste verbinden.

Das Attribut ist mit dem Namen **ToolbarButtonActive** dynamisierbar. Der Datentyp ist BOOLEAN.

ToolbarButtonAdd-Eigneschaft

Neu - ToolbarButtonAdd

Legt eine neue, benutzerdefinierte Tastenfunktion an. Den von WinCC vergebenen Name können Sie im Feld "Objektname" ändern.

Das Attribut ist mit dem Namen **ToolbarButtonAdd** dynamisierbar. Der Datentyp ist STRING.

ToolbarButtonBeginGroup-Eigenschaft

Separator - ToolbarButtonBeginGroup

Legt fest, ob vor der ausgewählten Tastenfunktion ein Trennzeichen eingefügt wird. Mit den Trennzeichen können Sie die Schaltflächen der Tastenfunktionen gruppieren.

Wert	Erklärung
TRUE	Vor der ausgewählten Tastenfunktion ist das Trennzeichen eingefügt.
FALSE	Vor der ausgewählten Tastenfunktion ist kein Trennzeichen eingefügt.

Das Attribut ist mit dem Namen **ToolbarButtonBeginGroup** dynamisierbar. Der Datentyp ist BOOLEAN.

ToolbarButtonClick AlarmControl-Eigenschaft

ToolbarButtonClick

Löst die mit der Taste der Symbolleiste verbundene Funktion aus. Die "ID" kann der Programmierer nutzen, um die entsprechende Tastenfunktion aufzurufen.

ID	Tastenfunktion	ID	Tastenfunktion
1	"Hilfe"	21	"Nächste Meldung"
2	"Konfigurationsdialog".	22	"Letzte Meldung"
3	"Meldeliste".	23	"Infotext-Dialog"
4	"Kurzzeitarchivliste".	24	"Kommentar-Dialog"
5	"Langzeitarchivliste"	25	"Loop in Alarm"
6	"Sperrliste".	26	"Meldung sperren"
7	"Hitliste"	27	"Meldung freigeben"
8	"Liste auszublendender Meldungen"	28	"Meldung ausblenden"
9	"Quittierung zentraler Melder"	29	"Meldung einblenden"
10	"Einzelquittierung"	30	"Sortier-Dialog"
11	"Sammelquittierung"	31	"Zeitbasis-Dialog"
18	"Not-Quittierung"	32	"Zeilen kopieren"
13	"Selektions-Dialog"	33	"Backup verbinden"
14	"Anzeigeoptions-Dialog"	34	"Backup trennen"
15	"Sperr-Dialog"	36	"Erste Seite"
17	"Drucken"	37	"Vorgehende Seite"
35	"Daten exportieren"	38	"Nächste Seite"
12	"Autoscroll"	39	"Letzte Seite"
19	"Erste Meldung"	1001	"Benutzerdefiniert 1"
20	"Vorhergehende Meldung"		

Das Attribut ist mit dem Namen **ToolbarButtonClick** dynamisierbar. Der Datentyp ist LONG.

ToolbarButtonClick FunctionTrendControl-Eigenschaft

ToolbarButtonClick

Löst die mit der Taste der Symbolleiste verbundene Funktion aus. Die "ID" kann der Programmierer nutzen, um die entsprechende Tastenfunktion aufzurufen.

ID	Tastenfunktion	ID	Tastenfunktion
1	"Hilfe".	13	"Zeitbereich wählen"
2	"Konfigurationsdialog"	14	"Vorhergehende Kurve"
4	"Zoomen Ausschnitt ".	15	"Nächste Kurve"
5	"Zoomen +/-"	16	"Stopp"
6	"X-Achse zoomen +/-".	16	"Start"
7	"Y-Achse zoomen +/-"	17	"Drucken"
8	"Kurvenbereich verschieben"	20	"Daten exportieren"
9	"Achsenbereich verschieben"	3	"Lineal"
10	"Originalansicht"	18	"Backup verbinden"
11	"Datenanbindung wählen"	19	"Backup trennen"
12	"Kurven wählen"	1001	"Benutzerdefiniert 1"

Das Attribut ist mit dem Namen **ToolbarButtonClick** dynamisierbar. Der Datentyp ist LONG.

ToolbarButtonClick OnlineTableControl-Eigenschaft

ToolbarButtonClick

Löst die mit der Taste der Symbolleiste verbundene Funktion aus. Die "ID" kann der Programmierer nutzen, um die entsprechende Tastenfunktion aufzurufen.

ID	Tastenfunktion	ID	Tastenfunktion
1	"Hilfe"	12	"Vorhergehende Spalte"
2	"Konfigurationsdialog"	13	"Nächste Spalte"
3	"Erster Datensatz"	14	"Stopp"
4	"Vorhergehender Datensatz"	14	"Start"
5	"Nächster Datensatz".	15	"Drucken"
6	"Letzter Datensatz"	20	"Daten exportieren"
7	"Bearbeiten"	16	"Statistikbereich festlegen"
8	"Zeilen kopieren"	17	"Statistik berechnen"
9	"Datenabindung wählen".	18	"Backup verbinden"
10	"Spalten wählen"	19	"Backup trennen"
11	"Zeitbereich wählen"	1001	"Benutzerdefiniert 1"

Das Attribut ist mit dem Namen **ToolbarButtonClick** dynamisierbar. Der Datentyp ist LONG.

ToolbarButtonClick OnlineTrendControl-Eigenschaft

ToolbarButtonClick

Löst die mit der Taste der Symbolleiste verbundene Funktion aus. Die "ID" kann der Programmierer nutzen, um die entsprechende Tastenfunktion aufzurufen.

ID	Tastenfunktion	ID	Tastenfunktion
1	"Hilfe"	17	"Zeitbereich wählen"
2	"Konfigurationsdialog".	18	"Vorhergehende Kurve"
3	"Erster Datensatz"	19	"Nächste Kurve"
4	"Vorhergehender Datensatz"	20	"Stopp"
5	"Nächster Datensatz"	20	"Start"
6	"Letzter Datensatz"	21	"Drucken"
8	"Zoomen Ausschnitt"	26	"Daten exportieren"
9	"Zoomen +/-"	7	"Lineal"
10	"Zeitachse zoomen +/-"	22	"Statistikbereich festlegen"
11	"Wertachse zoomen +/-"	23	"Statistik berechnen"
12	"Kurvenbereich verschieben"	24	"Backup verbinden"
13	"Achsenbereich verschieben"	25	"Backup trennen"
14	"Originalansicht"	27	"Relative Achse"

15	"Datenanbindung wählen"	1001	"Benutzerdefiniert 1"
16	"Kurven wählen"		

Das Attribut ist mit dem Namen **ToolbarButtonClick** dynamisierbar. Der Datentyp ist LONG.

ToolbarButtonClick RulerControl-Eigenschaft

ToolbarButtonClick

Löst die mit der Taste der Symbolleiste verbundene Funktion aus. Die "ID" kann der Programmierer nutzen, um die entsprechende Tastenfunktion aufzurufen.

ID	Tastenfuntion
1	"Hilfe".
2	"Konfigurationsdialog"
3	"Linealfenster ".
4	"Statistikbereich"
5	"Statistik".
6	"Drucken"
7	"Daten exportieren"
1001	"Benutzerdefiniert 1"

Das Attribut ist mit dem Namen **ToolbarButtonClick** dynamisierbar. Der Datentyp ist LONG.

ToolbarButtonClick UserArchiveControl-Eigenschaft

ToolbarButtonClick

Löst die mit der Taste der Symbolleiste verbundene Funktion aus. Die "ID" kann der Programmierer nutzen, um die entsprechende Tastenfunktion aufzurufen.

ID	Tastenfunktion	ID	Tastenfunktion
1	"Hilfe"	12	"Variablen lesen"
2	"Konfigurationsdialog"	13	"Variablen schreiben"
3	"Datenanbindung wählen"	14	"Archiv importieren"
4	"Erste Zeile"	15	"Archiv exportieren"
5	"Vorhergehende Zeile"	16	"Sortier-Dialog"
6	"Nächste Zeile"	17	"Selektions-Dialog"
7	"Letzte Zeile"	18	"Drucken"
8	"Zeilen löschen"	20	"Daten exportieren"
9	"Zeilen ausschneiden"	19	"Zeitbasis-Dialog"
10	"Zeilen kopieren"	1001	"Benutzerdefiniert 1"
11	"Zeilen einfügen"		

Das Attribut ist mit dem Namen **ToolbarButtonClick** dynamisierbar. Der Datentyp ist LONG.

ToolbarButtonCount-Eigenschaft

ToolbarButtonCount

Gibt die Anzahl der projektierbaren Tastenfunktionen an.

Das Attribut ist mit dem Namen **ToolbarButtonCount** dynamisierbar. Der Datentyp ist LONG.

ToolbarButtonEnabled-Eigenschaft

ToolbarButtonEnabled

Legt fest, ob eine benutzerdefinierte Taste der Symbolleiste bedienbar ist.

Das Attribut ist mit dem Namen **ToolbarButtonEnabled** dynamisierbar. Der Datentyp ist BOOLEAN.

ToolbarButtonHotKey-Eigenschaft

Hotkey - ToolbarButtonHotKey

Zeigt den Hotkey für die ausgewählte Tastenfunktion an.

Um einen Hotkey anzulegen oder zu ändern, klicken Sie auf das Feld "Hotkey" und drücken die gewünschte Taste oder Tastenkombination.

Das Attribut ist mit dem Namen **ToolbarButtonHotKey** dynamisierbar. Der Datentyp ist LONG.

ToolbarButtonID-Eigenschaft

Objekt-ID - ToolbarButtonID

Eindeutige Ident-Nummer für die gewählte Tastenfunktion. Die Ident-Nummer wird von WinCC vergeben und kann nicht geändert werden.

Das Attribut ist mit dem Namen **ToolbarButtonID** dynamisierbar. Der Datentyp ist LONG.

ToolbarButtonIndex-Eigenschaft

ToolbarButtonIndex

Referenziert eine Tastenfunktion. Unter Verwendung des Attributs können Sie einer bestimmten Tastenfunktion die Werte anderer Attribute zuweisen.

Gültige Werte für "ToolbarButtonIndex" liegen zwischen 0 und "ToolbarButtonCount" minus 1. Das Attribut "ToolbarButtonCount" gibt die Anzahl der projektierbaren Tastenfunktionen an.

Das Attribut "ToolbarButtonIndex" ist über das Attribut **ToolbarButtonRepos** dynamisierbar. Der Datentyp ist LONG.

ToolbarButtonLocked-Eigenschaft

ToolbarButtonLocked

Legt für eine benutzerdefinierte Taste der Symbolleiste fest, ob der eingerastete, gedrückte Zustand der Taste dargestellt wird.

Das Attribut ist mit dem Namen **ToolbarButtonLocked** dynamisierbar. Der Datentyp ist BOOLEAN.

ToolbarButtonName-Eigenschaft

Objektname - ToolbarButtonName

Zeigt den Namen für die ausgewählte Tastenfunktion an. Den Namen für benutzerdefinierte Tastenfunktionen können Sie ändern.

Das Attribut "ToolbarButtonName" für benutzerdefinierte Tastenfunktionen ist über das Attribut **ToolbarButtonRename** dynamisierbar. Der Datentyp ist STRING.

ToolbarButtonPasswordLevel-Eigenschaft

Bedienberechtigung - ToolbarButtonPasswordLevel

Zeigt die Berechtigung für die ausgewählte Tastenfunktion an. Über die Auswahlschaltfläche können Sie die Berechtigung ändern.

Die Berechtigungen werden im Editor "User Administrator" projektiert.

Das Attribut ist mit dem Namen **ToolbarButtonPasswordLevel** dynamisierbar. Der Datentyp ist LONG.

ToolbarButtonRemove-Eigenschaft

Entfernen - ToolbarButtonRemove

Entfernt die ausgewählte Tastenfunktion aus der Liste. Nur benutzerdefinierte Tastenfunktionen können Sie entfernen.

Das Attribut ist mit dem Namen **ToolbarButtonRemove** dynamisierbar. Der Datentyp ist STRING.

ToolbarButtonRename-Eigenschaft

ToolbarButtonRename

Ändert den Namen des benutzerdefinierten Elements der Symbolleiste, das über das Attribut "ToolbarButtonIndex" referenziert wird.

Für benutzerdefinierte Elemente ist das Attribut mit dem Namen **ToolbarButtonRename** dynamisierbar. Mit "ToolbarButtonRename" dynamisieren Sie auch das Attribut "ToolbarButtonName". Der Datentyp ist STRING.

ToolbarButtonRepos-Eigenschaft

Auf/Ab - ToolbarButtonRepos

Ändert die Reihenfolge der Tastenfunktionen. "Auf" und "Ab" bewegen die ausgewählte Tastenfunktion in der Liste nach oben oder unten. Dadurch wird die Tastenfunktion in der Symbolleiste des Controls weiter vorne oder hinten platziert.

Das Attribut ist mit dem Namen **ToolbarButtonRepos** dynamisierbar. Der Datentyp ist LONG.

ToolbarButtonTooltipText-Eigenschaft

ToolbarButtonTooltipText

Legt den Text für den Tooltip der Taste fest.

Das Attribut ist mit dem Namen **ToolbarButtonTooltipText** dynamisierbar. Der Datentyp ist STRING.

ToolbarButtonUserDefined-Eigenschaft

ToolbarButtonUserDefined

Zeigt, ob die Taste der Symbolleiste vom Projekteur als neue, benutzerdefinierte Taste hinzugefügt wurde.

Wert	Erklärung
TRUE	Die Taste der Symbolleiste ist benutzerdefiniert.
FALSE	Die Taste der Symbolleiste ist vom System vorgegeben.

Das Attribut ist mit dem Namen **ToolbarButtonUserDefined** dynamisierbar. Der Datentyp ist BOOLEAN.

ToolbarButtonVisible-Eigenschaft

Tastenfunktionen - ToolbarButtonVisible

Aktivieren Sie in der Liste die Tastenfunktionen, die Sie in der Symbolleiste anzeigen wollen.

Klicken Sie auf einen Eintrag in der Liste, um die Eigenschaften anzupassen oder um die Position in der Symbolleiste des Controls mit den Tasten "Auf" und "Ab" zu ändern.

Das Attribut ist mit dem Namen **ToolbarButtonVisible** dynamisierbar. Der Datentyp ist BOOLEAN.

ToolbarButtons-Eigenschaft

Beschreibung

Legt die in der Symbolleiste enthaltenen Schaltflächen fest oder gibt sie zurück, durch setzen oder rücksetzen des entsprechenden Bits. Jeder Schaltfläche ist ein Bit zugeordnet. Es gibt keine Beschränkung bei der Kombination der Bits.

Bit - Wert (hex) ; Wert (dez) ; Schaltfläche:

- 0 - 0x00000001; 1; Meldeliste
- 1 - 0x00000002; 2; Kurzzeitarchivliste
- 2 - 0x00000004; 4; Langzeitarchivliste
- 3 - 0x00000008; 8; Quittierung zentraler Melder
- 4 - 0x00000010; 16; Einzelquittierung
- 5 - 0x00000020; 32; Sammelquittierung
- 6 - 0x00000040; 64; Autoscroll
- 7 - 0x00000080; 128; Selektions-Dialog
- 8 - 0x00000100; 256; Sperr-Dialog
- 9 - 0x00000200; 512; Druck Meldeprotokoll
- 11 - 0x00000800; 2048; Not-Quittierung
- 12 - 0x00001000; 4096; Erste Meldung
- 13 - 0x00002000; 8192; Letzte Meldung
- 14 - 0x00004000; 16384; Nächste Meldung
- 15- 0x00008000; 32768; Vorherige Meldung
- 16 - 0x00010000; 65536; Infotext-Dialog
- 17 -0x00020000; 131072; Kommentar-Dialog
- 18 - 0x00040000; 262144; Loop in Alarm
- 20 - 0x00100000; 1048576; Druck aktuelle Ansicht
- 21 - 0x00200000; 2097152; Sperrliste
- 22 - 0x00400000; 4194304; Meldung sperren/freigeben
- 23 - 0x00800000; 8388608; Sortier-Dialog
- 24 - 0x01000000; 16777216; Zeitbasis-Dialog
- 25 - 0x02000000; 33554432; Hitliste

Um mehrere Schaltflächen anzuzeigen, müssen deren Werte logisch verODERt werden.
Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

ToolbarHotKeys-Eigenschaft

Beschreibung

Legt die Hotkeys der Schaltflächen in der Symbolleiste fest oder gibt sie zurück. Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

ToolbarShowTooltips-Eigenschaft

Tooltips - ToolbarShowTooltips

Legt fest, ob in Runtime die Tooltips zu den Tastenfunktionen angezeigt werden.

Wert	Erklärung
TRUE	Die Tooltips werden angezeigt.
FALSE	Die Tooltips werden nicht angezeigt.

Das Attribut ist mit dem Namen **ToolbarShowTooltips** dynamisierbar. Der Datentyp ist BOOLEAN.

Das Attribut zum Festlegen des Tooltip-Textes ist "ToolbarButtonTooltipText".

ToolbarUseBackColor-Eigenschaft

anzeigen Hintergrundfarbe - ToolbarUseBackColor

Legt fest, ob die Hintergrundfarbe der Symbolleiste angezeigt wird.

Wert	Erklärung
TRUE	Die Hintergrundfarbe der Symbolleiste wird angezeigt.
FALSE	Die Hintergrundfarbe der Symbolleiste wird nicht angezeigt.

Das Attribut ist mit dem Namen **ToolbarUseBackColor** dynamisierbar. Der Datentyp ist BOOLEAN.

ToolbarUseHotKeys-Eigenschaft

Hotkeys - ToolbarUseHotKeys

Legt fest, ob die Hotkeys für die Tastenfunktionen in Runtime aktiviert sind. Die Hotkeys für die Tastenfunktion fügen Sie im Feld "Hotkey" ein.

Wert	Erklärung
TRUE	Die Hotkeys sind aktiviert.
FALSE	Die Hotkeys sind nicht aktiviert.

Das Attribut ist mit dem Namen **ToolbarUseHotKeys** dynamisierbar. Der Datentyp ist BOOLEAN.

ToolbarVisible-Eigenschaft

Symbolleiste anzeigen - ToolbarVisible

Legt fest, ob die Symbolleiste des Controls angezeigt wird.

Wert	Erklärung
TRUE	Die Symbolleiste wird angezeigt.
FALSE	Die Symbolleiste wird nicht angezeigt.

Das Attribut ist mit dem Namen **ToolbarVisible** dynamisierbar. Der Datentyp ist BOOLEAN.

ToolTip - TrendLower

ToolTipText-Eigenschaft

Beschreibung

Legt den Text fest, der als Tooltip angezeigt wird, wenn Sie mit der Maus über das Objekt fahren oder gibt ihn zurück.

STRING (Schreib-Lese-Zugriff)

Beispiel

Das folgende Beispiel weist jedem Objekt des Bilds "NewPDL1" einen Tooltiptext zu. Im Bild "NewPDL1" sind nur Objekte enthalten, die die Eigenschaft ToolTipText enthalten:

```
'VBS89
Dim objScreen
Dim objScrItem
Dim lngIndex
Dim strName
lngIndex = 1
Set objScreen = HMIRuntime.Screens("NewPDL1")
For lngIndex = 1 To objScreen.ScreenItems.Count
strName = objScreen.ScreenItems(lngIndex).ObjectName
Set objScrItem = objScreen.ScreenItems(strName)
'
'Assign tooltiptexts to the objects
objScrItem.ToolTipText = "Name of object is " & strName
Next
```

Siehe auch

Radio-Box (Seite 217)
Zustandsanzeige (Seite 210)
Verbinder (Seite 180)
Textliste (Seite 208)
Statischer Text (Seite 178)
Slider (Seite 221)
Sammelanzeige (Seite 205)
Rundrechteck (Seite 175)
Rundbutton (Seite 219)
Rechteck (Seite 172)
Polygonzug (Seite 170)
Polygon (Seite 168)
OLE-Objekt (Seite 203)
Linie (Seite 166)
Kreissegment (Seite 164)
Kreisbogen (Seite 162)
Kreis (Seite 160)
Gruppe (Seite 294)
Grafik-Objekt (Seite 198)
Ellipsensegment (Seite 158)
Ellipsenbogen (Seite 156)
Ellipse (Seite 153)
EA-Feld (Seite 195)
Check-Box (Seite 215)
Button (Seite 212)
Balken (Seite 186)
Anwender-Objekt (Seite 293)
3D-Balken (Seite 181)

Top-Eigenschaft

Funktion

Legt die y-Koordinate eines Objektes (gemessen vom linken, oberen Bildrand) in Pixel fest oder gibt sie zurück. Die y-Koordinate bezieht sich auf die Ecke links oben des objektumfassenden Rechteckes.

LONG (Schreib-Lese-Zugriff)

Beispiel

Das folgende Beispiel verschiebt alle Objekte des Bildes "NewPDL1" um 5 Pixel nach oben:

```
'VBS90
Dim objScreen
Dim objScrItem
Dim lngIndex
Dim strName
lngIndex = 1
Set objScreen = HMIRuntime.Screens("NewPDL1")
For lngIndex = 1 To objScreen.ScreenItems.Count
strName = objScreen.ScreenItems(lngIndex).ObjectName
Set objScrItem = objScreen.ScreenItems(strName)
objScrItem.Top = objScrItem.Top - 5
Next
```

Siehe auch

Left-Eigenschaft (Seite 454)

ScreenItem-Objekt (Seite 134)

TopConnectedConnectionPointIndex-Eigenschaft

Beschreibung

Gibt die Indexnummer des oberen Verbinderpunktes an oder setzt sie.

Long Schreib-Lese-Zugriff

Siehe auch

Verbinder (Seite 180)

ScreenItem-Objekt (Seite 134)

TopConnectedObjectName-Eigenschaft

Beschreibung

Gibt den Objektnamen des Objektes an, das am oberen Verbinderpunkt angedockt ist oder setzt ihn.

Long Schreib-Lese-Zugriff

Siehe auch

Verbinder (Seite 180)

ScreenItem-Objekt (Seite 134)

Transparency-Eigenschaft

Beschreibung

Legt den Prozentsatz der Transparenz des Objekts fest und gibt ihn zurück.

0 = keine Transparenz; 100 = vollständige Transparenz (Unsichtbarkeit)

Die Texte und Felder der grafischen Objekte werden nur bei "100" transparent dargestellt.

In Runtime funktioniert auch ein völlig transparentes und damit unsichtbares Objekt.

Transparent-Eigenschaft

Beschreibung

TRUE, wenn die Schaltfläche in der Farbe "BackColor" deckend erscheint. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Push Button Control (Seite 271)

ScreenItem-Objekt (Seite 134)

Trend-Eigenschaft

Beschreibung

TRUE, wenn die Tendenz (steigend oder fallend) des zu überwachenden Messwertes mit einem kleinen Pfeil angezeigt werden soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

TrendActualize-Eigenschaft

Aktualisieren -TrendActualize

Legt fest, ob die ausgewählte Kurve aktualisiert wird.

Wert	Erklärung
TRUE	Die ausgewählte Kurve wird immer aktualisiert.
FALSE	Die ausgewählte Kurve wird nicht aktualisiert. Diese Einstellung ist sinnvoll, wenn eine archivierte Kurve mit einer aktuellen Kurve verglichen wird.

Das Attribut ist mit dem Namen **TrendActualize** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendAdd-Eigenschaft

Neu - TrendAdd

Legt eine neue Kurve an.

Das Attribut ist mit dem Namen **TrendAdd** dynamisierbar. Der Datentyp ist STRING.

TrendAutoRangeBeginTagName-Eigenschaft

TrendAutoRangeBeginTagName

Wenn der automatische Wertebereich über Online-Variablen ermittelt wird, legt das Attribut die Variable für die Untergrenze des Wertebereichs fest.

Das Attribut ist mit dem Namen **TrendAutoRangeBeginTagName** dynamisierbar. Der Datentyp ist STRING.

TrendAutoRangeBeginValue-Eigenschaft

TrendAutoRangeBeginValue

Wenn der automatische Wertebereich über die Projektierung der Untergrenze und Obergrenze ermittelt wird, legt das Attribut den Wert für die Untergrenze des Wertebereichs fest.

Das Attribut ist mit dem Namen **TrendAutoRangeBeginValue** dynamisierbar. Der Datentyp ist DOUBLE.

TrendAutoRangeEndTagName-Eigenschaft

TrendAutoRangeEndTagName

Wenn der automatische Wertebereich über Online-Variablen ermittelt wird, legt das Attribut die Variable für die Obergrenze des Wertebereichs fest.

Das Attribut ist mit dem Namen **TrendAutoRangeEndTagName** dynamisierbar. Der Datentyp ist STRING.

TrendAutoRangeEndValue-Eigenschaft

TrendAutoRangeEndValue

Wenn der automatische Wertebereich über die Projektierung der Untergrenze und Obergrenze ermittelt wird, legt das Attribut den Wert für die Obergrenze des Wertebereichs fest.

Das Attribut ist mit dem Namen **TrendAutoRangeEndValue** dynamisierbar. Der Datentyp ist DOUBLE.

TrendAutoRangeSource-Eigenschaft

TrendAutoRangeSource

Legt fest, wie der automatische Wertebereich der Kurvendaten ermittelt wird.

Wert	Beschreibung	Erklärung
0	Anzeigedaten	Der Wertebereich wird automatisch über die angezeigten Daten ermittelt.
1	Wertebereich	Der Wertebereich wird durch die Projektierung der Untergrenze und Obergrenze des Wertebereichs festgelegt. Die Werte der Untergrenze und Obergrenze werden in den Attributen "TrendAutoRangeBeginValue" und "TrendAutoRangeEndValue" abgebildet.
2	Online-Variablen	Die Untergrenze und Obergrenze des Wertebereichs wird aus den Werten von verbundenen Online-Variablen gebildet. Die Variablen der Untergrenze und Obergrenze werden in den Attributen "TrendAutoRangeBeginTagName" und "TrendAutoRangeEndTagName" abgebildet.

Das Attribut ist mit dem Namen **TrendAutoRangeSource** dynamisierbar. Der Datentyp ist LONG.

TrendBeginTime-Eigenschaft

Anfangszeitpunkt - TrendBeginTime

Legt den Anfangszeitpunkt des Zeitbereichs für die Datenversorgung der ausgewählten Kurve fest.

Das Attribut ist mit dem Namen **TrendBeginTime** dynamisierbar. Der Datentyp ist Date.

TrendColor-Eigenschaft

Kurvenfarbe - TrendColor

Gibt die Farbe der Kurve an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **TrendColor** dynamisierbar. Der Datentyp ist LONG.

TrendColor-Eigenschaft (vor WinCC V7)

Beschreibung

Legt die Farbe der Trendanzeige fest oder gibt sie zurück.
Die Trendanzeige stellt die Tendenz (steigend oder fallend) des zu überwachenden Messwertes mit einem kleinen Pfeil dar. Um die Trendanzeige zu aktivieren, muss die Eigenschaft Trend auf "True" gesetzt sein. LONG Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

TrendCount-Eigenschaft

TrendCount

Gibt die Anzahl der projektierten Kurven an.

Das Attribut ist mit dem Namen **TrendCount** dynamisierbar. Der Datentyp ist LONG.

TrendEndTime-Eigenschaft

Endzeitpunkt - TrendEndTime

Legt den Endzeitpunkt des Zeitbereichs für die Datenanbindung der ausgewählten Kurve fest.

Das Attribut ist mit dem Namen **TrendEndTime** dynamisierbar. Der Datentyp ist Date.

TrendExtendedColorSet-Eigenschaft

Erweitert - TrendExtendedColorSet

Legt fest, ob Sie die Punktfarbe und die Füllfarbe projektieren können und ob die Farben in Runtime dargestellt werden.

Wert	Erklärung
TRUE	Die Einstellungen in den Feldern "Punktfarbe" und "Füllfarbe" sind projektierbar und in Runtime wirksam.
FALSE	Die Einstellungen in den Feldern "Punktfarbe" und "Füllfarbe" sind nicht projektierbar und in Runtime nicht wirksam.

Das Attribut ist mit dem Namen **TrendExtendedColorSet** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendFill-Eigenschaft

Gefüllt - TrendFill

Legt fest, ob die Fläche unterhalb der Kurve gefüllt dargestellt wird.

Wert	Erklärung
TRUE	Die Fläche unterhalb der Kurve wird gefüllt dargestellt. Wenn die Option "Erweitert" nicht aktiviert ist, wird die Kurvenfarbe als Füllfarbe verwendet. Bei der Kurvenart "Werte darstellen" wird der Texthintergrund in der Kurvenfarbe dargestellt. Als Textfarbe wird die Hintergrundfarbe des Controls verwendet.
FALSE	Die Kurve wird nicht gefüllt dargestellt.

Das Attribut ist mit dem Namen **TrendFill** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendFillColor-Eigenschaft

Füllfarbe - TrendFillColor

Gibt die Füllfarbe der Kurve an. Bei der Kurvenart "Werte darstellen" wird die Texthintergrundfarbe festgelegt.

Die Füllfarbe wird verwendet, wenn die Option "Gefüllt" aktiviert bzw. "TrendFill" "TRUE" ist. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Projektierung ist nur möglich, wenn die Option "Erweitert" aktiviert bzw. "TrendExtendedColorSet" "TRUE" ist.

Das Attribut ist mit dem Namen **TrendFillColor** dynamisierbar. Der Datentyp ist LONG.

TrendIndex-Eigenschaft

TrendIndex

Referenziert eine projektierte Kurve. Unter Verwendung des Attributs können Sie einer bestimmten Kurve die Werte anderer Attribute zuweisen. Der Index muss immer gesetzt werden, bevor Sie die Eigenschaften einer Kurve in Runtime ändern.

Gültige Werte für "TrendIndex" liegen zwischen 0 und "TrendCount" minus 1. Das Attribut "TrendCount" gibt die Anzahl der projektierten Kurven an.

Das Attribut "TrendIndex" ist über das Attribut **TrendRepos** dynamisierbar. Der Datentyp ist LONG.

TrendLabel-Eigenschaft

Bezeichnug - TrendLabel

Legt die Bezeichnung der ausgewählten Kurve fest. Die Bezeichnung wird in Runtime angezeigt, wenn das Attribut "UseTrendNameAsLabel" den Wert "FALSE" hat.

Das Attribut ist mit dem Namen **TrendLabel** dynamisierbar. Der Datentyp ist STRING.

TrendLineStyle-Eigenschaft

Linienart - TrendLineStyle

Legt fest, welche Linienart zur Darstellung der Kurve verwendet wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	durchgezogen	Die Kurve wird durchgezogen dargestellt.
1	gestrichelt	Die Kurve wird gestrichelt dargestellt.
2	Punkte	Die Kurve wird mit einer punktierten Linie dargestellt.
3	strichpunktiert	Die Kurve wird mit einer strichpunktierten Linie dargestellt.
4	Strich-Punkt-Punkt	Die Kurve wird mit einer Strich-Punkt-Punkt-Linie dargestellt.

Das Attribut ist mit dem Namen **TrendLineStyle** dynamisierbar. Der Datentyp ist LONG.

TrendLineType-Eigenschaft

Kurvenart - TrendLineType

Legt fest, wie die Kurve dargestellt wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	keine	Nur die Punkte werden dargestellt.
1	Punkte linear verbinden	Eine Kurve mit den linear verbundenen Punkten wird dargestellt.
2	Treppenkurve	Eine Treppenkurve mit den verbundenen Punkten wird dargestellt.
3	Werte darstellen	Nur beim OnlineTrendControl projektierbar. Anstatt von Kurvenpunkten wird ein Wert an jedem Zeitstempel bzw. an der Haupt-Gitternetzlinie der Zeitachse angezeigt.

Das Attribut ist mit dem Namen **TrendLineType** dynamisierbar. Der Datentyp ist LONG.

TrendLineWidth-Eigenschaft

Linienstärke - TrendLineWidth

Legt die Linienstärke der dargestellten Linie fest.

Das Attribut ist mit dem Namen **TrendLineWidth** dynamisierbar. Der Datentyp ist LONG.

TrendLowerLimit-Eigenschaft

TrendLowerLimit

Gibt den unteren Grenzwert für eine Variable an. Wenn die Variable den Wert von "TrendLowerLimit" unterschreitet, werden die Werte mit der in "TrendLowerLimitColor" eingestellten Farbe gekennzeichnet. Die Angabe ist wirksam, wenn das Attribut "TrendLowerLimitColoring" den Wert "TRUE" hat.

Das Attribut ist mit dem Namen **TrendLowerLimit** dynamisierbar. Der Datentyp ist DOUBLE.

TrendLowerLimitColor-Eigenschaft

TrendLowerLimitColor

Legt die Farbe fest, die Variablenwerte kennzeichnet, die unterhalb des Werts von "TrendLowerLimit" liegen. Die Einstellung ist wirksam, wenn das Attribut "TrendLowerLimitColoring" den Wert "TRUE" hat.

Das Attribut ist mit dem Namen **TrendLowerLimitColor** dynamisierbar. Der Datentyp ist LONG.

TrendLowerLimitColoring-Eigenschaft

TrendLowerLimitColoring

Legt fest, ob das Attribut "TrendLowerLimitColor" verwendet wird, um die Variablenwerte zu kennzeichnen, die unterhalb des Werts von "TrendLowerLimit" liegen.

Wert	Erklärung
TRUE	Das Attribut "TrendLowerLimitColor" ist wirksam.
FALSE	Das Attribut "TrendLowerLimitColor" ist unwirksam.

Das Attribut ist mit dem Namen **TrendLowerLimitColoring** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendMeasure - TrendVisible

TrendMeasurePoints-Eigenschaft

Anzahl der Messpunkte - TrendMeasurePoints

Legt die Anzahl der Messpunkte fest, die zur Darstellung der ausgewählten Kurve verwendet werden.

Legt die Anzahl der Wertepaare fest, wenn die Kurve über ein Anwenderarchiv versorgt wird.

Das Attribut ist mit dem Namen **TrendMeasurePoints** dynamisierbar. Der Datentyp ist LONG.

TrendName-Eigenschaft

Objektname - TrendName

Zeigt den Namen der ausgewählten Kurve an. Den Namen legen Sie auf der Registerkarte "Kurven" fest.

Das Attribut "TrendName" ist über das Attribut **TrendRename** dynamisierbar. Der Datentyp ist STRING.

TrendPointColor-Eigenschaft

Punktfarbe - TrendPointColor

Gibt die Farbe der Punkte auf der Kurve an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Projektierung ist nur möglich, wenn die Option "Erweitert" aktiviert bzw. "TrendExtendedColorSet" "TRUE" ist.

Das Attribut ist mit dem Namen **TrendPointColor** dynamisierbar. Der Datentyp ist LONG.

TrendPointStyle-Eigenschaft

Punkteart - TrendPointStyle

Legt fest, wie die Punkte auf der Kurve dargestellt werden.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	kein	Die Punkte werden nicht dargestellt.
1	Punkte	Die Punkte der Kurve werden als 1 Pixel große Punkte dargestellt. Die Einstellung im Feld "Punktbreite" ist nicht wirksam.

Wert	Beschreibung	Erklärung
2	Quadrate	Die Punkte werden als Quadrate dargestellt. Die Einstellung im Feld "Punktbreite" ist wirksam.
3	Kreise	Die Punkte werden als Kreise dargestellt. Die Einstellung im Feld "Punktbreite" ist wirksam.

Das Attribut ist mit dem Namen **TrendPointStyle** dynamisierbar. Der Datentyp ist LONG.

TrendPointWidth-Eigenschaft

Punktbreite - TrendPointWidth

Legt die Punktbreite in Pixel fest. Die Punktbreite können Sie nur für die Punktearten "Quadrate" und "Kreise" projektieren.

Das Attribut ist mit dem Namen **TrendPointWidth** dynamisierbar. Der Datentyp ist LONG.

TrendProvider -Eigenschaft

Datenversorgung - TrendProvider

Legt die Datenversorgung der ausgewählten Kurve fest.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Keine	Keine projektierte Datenversorgung, um eine Anbindung in Runtime per Skript herzustellen.
1	Archivvariablen	Datenversorgung mit Archivvariablen eines Prozesswertarchivs.
2	Onlinevariablen	Datenversorgung mit Onlinevariablen aus dem Variablenhaushalt.
3	Anwenderarchiv	Datenversorgung mit Spalten eines Anwenderarchivs.

Das Attribut ist mit dem Namen **TrendProvider** dynamisierbar. Der Datentyp ist LONG.

TrendProviderCLSID_FunctionTrend-Eigenschaft

TrendProviderCLSID_FunctionTrend

Zeigt die Datenversorgung der ausgewählten Kurve an.

Wert	Erklärung
	Keine projektierte Datenversorgung, um eine Anbindung in Runtime per Skript herzustellen.
{416A09D2-8B5A-11D2-8B81-006097A45D48}	Datenversorgung mit Archivvariablen eines Prozesswertarchivs.

Wert	Erklärung
{A3F69593-8AB0-11D2-A440-00A0C9DBB64E}	Datenversorgung mit Onlinevariablen aus dem Variablenhaushalt.
{2DC9B1C8-4FC1-41B1-B354-3E469A13FBFD}	Datenversorgung mit Spalten eines Anwenderarchivs.

Das Attribut ist mit dem Namen **TrendProviderCLSID** dynamisierbar. Der Datentyp ist STRING.

TrendProviderCLSID_OnlineTrend-Eigenschaft

TrendProviderCLSID_OnlineTrend

Zeigt die Datenversorgung der ausgewählten Kurve an.

Wert	Erklärung
	Keine projektierte Datenversorgung, um eine Anbindung in Runtime per Skript herzustellen.
{416A09D2-8B5A-11D2-8B81-006097A45D48}	Datenversorgung mit Archivvariablen eines Prozesswertarchivs.
{A3F69593-8AB0-11D2-A440-00A0C9DBB64E}	Datenversorgung mit Onlinevariablen aus dem Variablenhaushalt.

Das Attribut ist mit dem Namen **TrendProviderCLSID** dynamisierbar. Der Datentyp ist STRING.

TrendRangeType-Eigenschaft

Einstellung Zeitbereich - TrendRangeType

Legt den Zeitbereich für die ausgewählte Kurve fest, in dem die Kurve mit Daten versorgt wird.

Bei einer Datenversorgung über Anwenderarchive können Sie nur die Anzahl der Messpunkte festlegen.

Folgende Projektiermöglichkeiten stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Zeitbereich	Für die Datenanbindung werden Anfangszeitpunkt und Zeitbereich festgelegt.
1	Anfangs- bis Endzeitpunkt	Für die Datenanbindung werden Anfangs- und Endzeitpunkt festgelegt.
2	Anzahl der Messpunkte	Für die Datenanbindung werden Anfangszeitpunkt und Anzahl der Messpunkte festgelegt.

Das Attribut ist mit dem Namen **TrendRangeType** dynamisierbar. Der Datentyp ist LONG.

TrendRemove-Eigenschaft

Entfernen - TrendRemove

Entfernt die ausgewählte Kurven aus der Liste.

Das Attribut ist mit dem Namen **TrendRemove** dynamisierbar. Der Datentyp ist STRING.

TrendRename-Eigenschaft

TrendRename

Ändert den Namen der Kurve, die über das Attribut "TrendIndex" referenziert wird.

Das Attribut ist mit dem Namen **TrendRename** dynamisierbar. Mit "TrendRename" dynamisieren Sie auch das Attribut "TrendName". Der Datentyp ist STRING.

TrendRepos-Eigenschaft

Auf/Ab - TrendRepos

Ändert die Reihenfolge der ausgewählten Kurve im Kurvenfenster. "Auf" und "Ab" bewegen die ausgewählte Kurve in der Liste nach oben oder unten. Dadurch wird die Kurve in Runtime weiter im Vordergrund oder Hintergrund dargestellt.

Das Attribut ist mit dem Namen **TrendRepos** dynamisierbar. Der Datentyp ist LONG.

TrendSelectTagName-Eigenschaft

TrendSelectTagName

Öffnet den Dialog zur Auswahl des Variablennamens für die Datenversorgung der Y-Achse im WinCC OnlineTrendControl. Das Attribut können Programmierer nutzen, um z. B. über eine Schaltfläche den Benutzer einen Variablennamen auswählen zu lassen.

Das Attribut ist mit dem Namen **TrendSelectTagName** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendSelectTagNameX-Eigenschaft

TrendSelectTagNameX

Öffnet den Dialog zur Auswahl des Variablennamens für die Datenversorgung der X-Achse im WinCC FunctionTrendControl. Das Attribut können Programmierer nutzen, um z. B. über eine Schaltfläche den Benutzer einen Variablennamen auswählen zu lassen.

Das Attribut ist mit dem Namen **TrendSelectTagNameX** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendSelectTagNameY-Eigenschaft

TrendSelectTagNameY

Öffnet den Dialog zur Auswahl des Variablennamens für die Datenversorgung der Y-Achse im WinCC FunctionTrendControl. Das Attribut können Programmierer nutzen, um z. B. über eine Schaltfläche den Benutzer einen Variablennamen auswählen zu lassen.

Das Attribut ist mit dem Namen **TrendSelectTagNameY** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendState-Eigenschaft

TrendState

Zeigt den Zustand der Datenanbindung der ausgewählten Kurve in Runtime an.

Das Attribut ist mit dem Namen **TrendState** dynamisierbar. Der Datentyp ist LONG.

TrendTagName-Eigenschaft

Variablenname - TrendTagName

Zeigt den Variablennamen der angebundenen Variablen an. Über die Schaltfläche öffnen Sie einen Dialog zur Auswahl einer Onlinevariablen bzw. einer Archivvariablen.

Das Attribut ist mit dem Namen **TrendTagName** dynamisierbar. Der Datentyp ist STRING.

TrendTagNameX-Eigenschaft

Variablenname X / Spalte X - TrendTagNameX

Zeigt den Namen der angebundenen Variablen bzw. Spalte für die X-Achse an. Über die Auswahlschaltfläche wählen Sie bezüglich der projektierten Datenversorgung eine Variable bzw. eine Spalte aus.

Das Attribut ist mit dem Namen **TrendTagNameX** dynamisierbar. Der Datentyp ist STRING.

TrendTagNameY-Eigenschaft

Variablenname Y / Spalte Y - TrendTagNameY

Zeigt den Namen der angebundenen Variablen bzw. Spalte für die Y-Achse an. Über die Auswahlschaltfläche wählen Sie bezüglich der projektierten Datenversorgung eine Variable bzw. eine Spalte aus.

Das Attribut ist mit dem Namen **TrendTagNameY** dynamisierbar. Der Datentyp ist STRING.

TrendTimeAxis-Eigenschaft

Zeitachse - TrendTimeAxis

Legt fest, welche Zeittachse für die ausgewählte Kurve verwendet wird. Die zur Verfügung stehenden Zeitachsen legen Sie auf der Registerkarte "Zeitachsen" fest.

Das Attribut ist mit dem Namen **TrendTimeAxis** dynamisierbar. Der Datentyp ist STRING.

TrendTimeRangeBase-Eigenschaft

Zeitbereich - TrendTimeRangeBase

Legt die Zeiteinheit zur Bestimmung des Zeitbereichs fest.

Folgende Zeiteinheiten stehen zur Verfügung:

Wert	Beschreibung
500	500 ms
1000	1 Sekunde
60000	1 Minute
3600000	1 Stunde
86400000	1 Tag

Das Attribut ist mit dem Namen **TrendTimeRangeBase** dynamisierbar. Der Datentyp ist LONG.

TrendTimeRangeFactor-Eigenschaft

Zeitbereich - TrendTimeRangeFactor

Legt den Faktor zur Bestimmung des Zeitbereichs fest. Nur ganzzahlige Faktoren sind zulässig.

Das Attribut ist mit dem Namen **TrendTimeRangeFactor** dynamisierbar. Der Datentyp ist SHORT.

TrendTrendWindow-Eigenschaft

Kurvenfenster - TrendTrendWindow

Legt fest, in welchem Kurvenfenster die ausgewählte Kurve dargestellt wird. Die zur Verfügung stehenden Kurvenfenster legen Sie auf der Registerkarte "Kurvenfenster" fest.

Das Attribut ist mit dem Namen **TrendTrendWindow** dynamisierbar. Der Datentyp ist STRING.

TrendUncertainColor-Eigenschaft

TrendUncertainColor

Werte haben einen unsicheren Status, wenn der Anfangswert nach dem Aktivieren von Runtime unbekannt ist oder ein Ersatzwert verwendet wird. Mit dem Attribut "TrendUncertainColor" legen Sie die Farbe fest, die für die Kennzeichnung dieser Werte verwendet wird. Ob die Angabe ausgewertet wird, ist abhängig vom Attribut "TrendUncertainColoring".

Das Attribut ist mit dem Namen **TrendUncertainColor** dynamisierbar. Der Datentyp ist LONG.

TrendUncertainColoring-Eigenschaft

TrendUncertainColoring

Werte haben einen unsicheren Status, wenn der Anfangswert nach dem Aktivieren von Runtime unbekannt ist oder ein Ersatzwert verwendet wird. Über "TrendUncertainColoring" legen Sie fest, ob derartige Werte, mit der in "TrendUncertainColor" eingestellten Farbe gekennzeichnet werden.

Wert	Erklärung
TRUE	Die Einstellung des Attributs "TrendUncertainColor" ist wirksam.
FALSE	Die Einstellung des Attributs " TrendUncertainColor " ist unwirksam.

Das Attribut ist mit dem Namen **TrendUncertainColoring** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendUpperLimit-Eigenschaft

TrendUpperLimit

Gibt den oberen Grenzwert für eine Variable an. Wenn die Variable den Wert von "TrendUpperLimit" unterschreitet, werden die Werte mit der in "TrendUpperLimitColor" eingestellten Farbe gekennzeichnet. Die Angabe ist wirksam, wenn das Attribut "TrendUpperLimitColoring" den Wert "TRUE" hat.

Das Attribut ist mit dem Namen **TrendUpperLimit** dynamisierbar. Der Datentyp ist DOUBLE.

TrendUpperLimitColor-Eigenschaft

TrendUpperLimitColor

Legt die Farbe fest, die Variablenwerte kennzeichnet, die unterhalb des Werts von "TrendLowerLimit" liegen. Die Einstellung ist wirksam, wenn das Attribut "TrendUpperLimitColoring" den Wert "TRUE" hat.

Das Attribut ist mit dem Namen **TrendUpperLimitColor** dynamisierbar. Der Datentyp ist LONG.

TrendUpperLimitColoring-Eigenschaft

TrendUpperLimitColoring

Legt fest, ob das Attribut "TrendUpperLimitColor" verwendet wird, um die Variablenwerte zu kennzeichnen, die unterhalb des Werts von "TrendUpperLimit" liegen.

Wert	Erklärung
TRUE	Die Einstellung des Attributs "TrendUpperLimitColor" ist wirksam.
FALSE	Die Einstellung des Attributs "TrendUpperLimitColor" ist unwirksam.

Das Attribut ist mit dem Namen **TrendUpperLimitColoring** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendValueAlignment-Eigenschaft

Ausrichtung - TrendValueAlignment

Legt für die Kurvenart "Werte darstellen" die Ausrichtung der dargestellten Werte fest.

Folgende Einstellungen stehen in Abhängigkeit von der Schreiberrichtung der Kurve zur Verfügung:

- Die Schreiberrichtung der Kurve ist "von rechts" bzw. "von links"

Wert	Beschreibung	Erklärung
0	unten	Die Werte werden unten im Kurvenfenster angezeigt.
1	zentriert	Die Werte werden zentriert im Kurvenfenster angezeigt.
2	oben	Die Werte werden oben im Kurvenfenster angezeigt.

- Die Schreiberrichtung der Kurve ist "von oben" bzw. "von unten"

Wert	Beschreibung	Erklärung
0	links	Die Werte werden links im Kurvenfenster angezeigt.
1	zentriert	Die Werte werden zentriert im Kurvenfenster angezeigt.
2	rechts	Die Werte werden rechts im Kurvenfenster angezeigt.

Das Attribut ist mit dem Namen **TrendValueAlignment** dynamisierbar. Der Datentyp ist LONG.

TrendValueAxis-Eigenschaft

Wertachse - TrendValueAxis

Legt fest, welche Wertachse für die ausgewählte Kurve verwendet wird. Die zur Verfügung stehenden Wertachsen legen Sie auf der Registerkarte "Wertachsen" fest.

Das Attribut ist mit dem Namen **TrendValueAxis** dynamisierbar. Der Datentyp ist STRING.

TrendValueUnit-Eigenschaft

Einheit - TrendValueUnit

Legt für die Kurvenart "Werte darstellen" eine Einheit der Werte fest, die dem darzustellenden Wert angehängt wird. Z. B. "%" oder "°C".

Das Attribut ist mit dem Namen **TrendValueUnit** dynamisierbar. Der Datentyp ist STRING.

TrendVisible-Eigenschaft

Kurven - TrendVisible

In der Liste werden die Kurven aufgelistet, die Sie angelegt haben.

Aktivieren Sie in der Liste die Kurven, die Sie in den Kurvenfenstern anzeigen wollen.

Klicken Sie auf eine Kurve in der Liste, um die Eigenschaften anzupassen und um Achsen und Kurvenfenster der Kurve zuzuordnen.

Das Attribut ist mit dem Namen **TrendVisible** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendWindow - TrendYAxis

TrendWindowAdd-Eigenschaft

Neu - TrendWindowAdd

Legt ein neues Kurvenfenster an.

Das Attribut ist mit dem Namen **TrendWindowAdd** dynamisierbar. Der Datentyp ist STRING.

TrendWindowCoarseGrid-Eigenschaft

Hauptskalierung - TrendWindowCoarseGrid

Legt fest, ob die Gitternetzlinien für die Hauptskalierung angezeigt werden.

Wert	Erklärung
TRUE	Die Gitternetzlinien für die Hauptskalierung werden angezeigt.
FALSE	Die Gitternetzlinien für die Hauptskalierung werden nicht angezeigt.

Das Attribut ist mit dem Namen **TrendWindowCoarseGrid** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendWindowCoarseGridColor-Eigenschaft

Farbe Hauptskalierung - TrendWindowCoarseGridColor

Gibt die Farbe der Gitternetzlinien für die Hauptskalierung an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **TrendWindowCoarseGridColor** dynamisierbar. Der Datentyp ist LONG.

TrendWindowCount-Eigenschaft

TrendWindowCount

Gibt die Anzahl der projektierten Kurvenfenster an.

Das Attribut ist mit dem Namen **TrendWindowCount** dynamisierbar. Der Datentyp ist LONG.

TrendWindowFineGrid-Eigenschaft

Hilfsskalierung - TrendWindowFineGrid

Legt fest, ob die Gitternetzlinien für die Hilfsskalierung angezeigt werden.

Wert	Erklärung
TRUE	Die Gitternetzlinien für die Hilfsskalierung werden angezeigt.
FALSE	Die Gitternetzlinien für die Hilfsskalierung werden nicht angezeigt.

Das Attribut ist mit dem Namen **TrendWindowFineGrid** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendWindowFineGridColor-Eigenschaft

Farbe Hilfsskalierung - TrendWindowFineGridColor

Gibt die Farbe der Gitternetzlinien für die Hilfsskalierung an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Das Attribut ist mit dem Namen **TrendWindowFineGridColor** dynamisierbar. Der Datentyp ist LONG.

TrendWindowForegroundTrendGrid-Eigenschaft

nur für Vordergrundkurve - TrendWindowForegroundTrendGrid

Legt fest, ob nur die Gitternetzlinien für die Vordergrundkurve im ausgewählten Kurvenfenster angezeigt werden.

Wert	Erklärung
TRUE	Die Gitternetzlinien für die Vordergrundkurve werden im Kurvenfenster angezeigt.
FALSE	Die Gitternetzlinien für alle Kurven werden im Kurvenfenster angezeigt.

Das Attribut ist mit dem Namen **TrendWindowForegroundTrendGrid** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendWindowGridInTrendColor-Eigenschaft

in Kurvenfarbe - TrendWindowGridInTrendColor

Legt fest, ob die Gitternetzlinien für die Hauptskalierung in der Kurvenfarbe dargestellt werden.

Wert	Erklärung
TRUE	Die Gitternetzlinien werden in der Kurvenfarbe dargestellt.
FALSE	Die Gitternetzlinien werden mit der im Feld "Farbe" eingestellten Farbe dargestellt.

Das Attribut ist mit dem Namen **TrendWindowGridInTrendColor** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendWindowHorizontalGrid-Eigenschaft

für X-Achse - TrendWindowHorizontalGrid

Legt fest, ob die horizontalen Gitternetzlinien angezeigt werden.

Wert	Erklärung
TRUE	Die horizontalen Gitternetzlinien werden angezeigt.
FALSE	Die horizontalen Gitternetzlinien werden nicht angezeigt.

Das Attribut ist mit dem Namen **TrendWindowHorizontalGrid** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendWindowIndex-Eigenschaft

TrendWindowIndex

Referenziert ein projektiertes Kurvenfenster. Unter Verwendung des Attributs können Sie einem bestimmten Kurvenfenster die Werte anderer Attribute zuweisen.

Gültige Werte für "TrendWindowIndex" liegen zwischen 0 und "TrendWindowCount" minus 1. Das Attribut "TrendWindowCount" gibt die Anzahl der projektierten Kurvenfenster an.

Das Attribut "TrendWindowIndex" ist über das Attribut **TrendWindowRepos** dynamisierbar. Der Datentyp ist LONG.

TrendWindowName-Eigenschaft

Objektname - TrendWindowName

Legt den Namen des ausgewählten Kurvenfensters fest.

Das Attribut "TrendWindowName" ist über das Attribut **TrendWindowRename** dynamisierbar. Der Datentyp ist STRING.

TrendWindowRemove-Eigenschaft

Entfernen - TrendWindowRemove

Entfernt das ausgewählte Kurvenfenster aus der Liste.

Das Attribut ist mit dem Namen **TrendWindowRemove** dynamisierbar. Der Datentyp ist STRING.

TrendWindowRename-Eigenschaft

TrendWindowRename

Ändert den Namen des Kurvenfensters, das über das Attribut "TrendWindowIndex" referenziert wird.

Das Attribut ist mit dem Namen **TrendWIndowRename** dynamisierbar. Mit "TrendWindowRename" dynamisieren Sie auch das Attribut "TrendWindowName". Der Datentyp ist STRING.

TrendWindowRepos-Eigenschaft

Auf/Ab - TrendWindowRepos

Ändert die Reihenfolge der Kurvenfenster. "Auf" und "Ab" bewegen die ausgewählte Kurve in der Liste nach oben oder unten.

Die Reihenfolge in der Liste bestimmt die Position im Control. Das erste Kurvenfenster wird an der untersten Position, das letzte Kurvenfenster an der obersten Position dargestellt.

Das Attribut ist mit dem Namen **TrendWindowRepos** dynamisierbar. Der Datentyp ist LONG.

TrendWindowRulerColor-Eigenschaft

Farbe des Lineals - TrendWindowRulerColor

Gibt die Farbe des Lineals an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Farbe ist projektierbar und darstellbar, wenn für die Darstellung des Lineals bzw. "TrendWindowRulerStyle" "1 - grafisch" eingestellt ist.

Das Attribut ist mit dem Namen **TrendWindowRulerColor** dynamisierbar. Der Datentyp ist LONG.

TrendWindowRulerLayer-Eigenschaft

Ebene der Lineale - TrendWindowRulerLayer

Legt die Darstellungsebene des Lineals im Kurvenfenster fest.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	hinter Gitternetz	Das Lineal wird hinter das Gitternetz gelegt.
1	hinter Kurven	Das Lineal wird hinter die Kurven und vor das Gitternetz gelegt.
2	vor Kurven	Das Lineal wird vor die Kurven gelegt.

Das Attribut ist mit dem Namen **TrendWindowRulerLayer** dynamisierbar. Der Datentyp ist LONG.

TrendWindowRulerStyle-Eigenschaft

Darstellung des Lineals - TrendWindowRulerStyle

Legt die Darstellung des Lineals fest.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	einfach	Das Lineal wird als eine einfache schwarze Linie dargestellt.
1	grafisch	Das Lineal wird in der projektierten "Farbe" und "Stärke" dargestellt.

Das Attribut ist mit dem Namen **TrendWindowRulerStyle** dynamisierbar. Der Datentyp ist LONG.

TrendWindowRulerWidth-Eigenschaft

Stärke des Lineals - TrendWindowRulerWidth

Legt die Stärke des Lineals in Pixel fest.

Die Stärke ist projektierbar und darstellbar, wenn für die Darstellung des Lineals bzw. "TrendWindowRulerStyle" "1 - grafisch" eingestellt ist.

Das Attribut ist mit dem Namen **TrendWindowRulerWidth** dynamisierbar. Der Datentyp ist LONG.

TrendWindowSpacePortion-Eigenschaft

Bereichsanteil - TrendWindowSpacePortion

Legt den Anteil des ausgewählten Kurvenfensters bei der Darstellung im Control fest.

Das Attribut ist mit dem Namen **TrendWindowSpacePortion** dynamisierbar. Der Datentyp ist LONG.

TrendWindowStatisticRulerColor-Eigenschaft

Farbe Lineal für Statistikbereich - TrendWindowStatisticRulerColor

Gibt die Farbe des Lineals für den Statistikbereich an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Farbe ist projektierbar und darstellbar, wenn für die Darstellung des Lineals für den Statistikbereich bzw. "TrendWindowStatisticRulerStyle" "1 - grafisch" eingestellt ist.

Das Attribut ist mit dem Namen **TrendWindowStatisticRulerColor** dynamisierbar. Der Datentyp ist LONG.

TrendWindowStatisticRulerStyle-Eigenschaft

Darstellung Lineal für Statistikbereich - TrendWindowStatisticRulerStyle

Legt die Darstellung des Lineals für die Festlegung des Statistikbereich fest.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	einfach	Das Lineal wird als eine einfache schwarze Linie dargestellt.
1	grafisch	Das Lineal wird in der projektierten "Farbe" und "Stärke" dargestellt.

Das Attribut ist mit dem Namen **TrendWindowStatisticRulerStyle** dynamisierbar. Der Datentyp ist LONG.

TrendWindowStatisticRulerWidth-eigenschaft

Stärke Lineal für Statistikbereich - TrendWindowStatisticRulerWidth

Legt die Stärke des Lineals für den Statistikbereich in Pixel fest.

Die Stärke des Lineals ist projektierbar und darstellbar, wenn für die Darstellung des Lineals für den Statistikbereich bzw. "TrendWindowStatisticRulerStyle" "1 - grafisch" eingestellt ist.

Das Attribut ist mit dem Namen **TrendWindowStatisticRulerWidth** dynamisierbar. Der Datentyp ist LONG.

TrendWindowVerticalGrid-Eigenschaft

für Y-Achse - TrendWindowVerticalGrid

Legt fest, ob die vertikalen Gitternetzlinien angezeigt werden.

Wert	Erklärung
TRUE	Die vertikalen Gitternetzlinien werden angezeigt.
FALSE	Die vertikalen Gitternetzlinien werden nicht angezeigt.

Das Attribut ist mit dem Namen **TrendWindowVerticalGrid** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendWindowVisible-Eigenschaft

Kurvenfenster - TrendWindowVisible

In der Liste werden die Kurvenfenster aufgelistet, die Sie angelegt haben.

Aktivieren Sie in der Liste die Kurvenfenster, die Sie im Control anzeigen wollen.

Klicken Sie auf einen Eintrag in der Liste, um die Eigenschaften des Lineals und die Gitternetzlinien anzupassen.

Das Attribut ist mit dem Namen **TrendWindowVisible** dynamisierbar. Der Datentyp ist BOOLEAN.

TrendXAxis-Eigenschaft

X-Achse - TrendXAxis

Legt fest, welche X-Achse für die ausgewählte Kurve verwendet wird. Die zur Verfügung stehenden X-Achsen legen Sie auf der Registerkarte "X-Achsen" fest.

Das Attribut ist mit dem Namen **TrendXAxis** dynamisierbar. Der Datentyp ist STRING.

TrendYAxis-Eigenschaft

Y-Achse - TrendYAxis

Legt fest, welche Y-Achse für die ausgewählte Kurve verwendet wird. Die zur Verfügung stehenden Y-Achsen legen Sie auf der Registerkarte "Y-Achsen" fest.

Das Attribut ist mit dem Namen **TrendYAxis** dynamisierbar. Der Datentyp ist STRING.

Type

Type-Eigenschaft

Beschreibung

Liest den Typ eines Objektes aus, z.B. "Rectangle", "Circle" oder "Line".

Der Objekttyp wird als String zurückgegeben. Readonly

Bei allen von WinCC bereitgestellten Graphikelementen wird eine spezielle Kennung als Typ zurückgeliefert. Sie ist unter der Rubrik "Typkennzeichnung in VBS" bei jeder Einzelbeschreibung der WinCC-Objekttypen zu finden.

Besonderheit

Bei WinCC-fremden Controls und OLE-Objekten wird die versionsunabhängige ProgID als Typ zurückgeliefert.

Aus der ProgID können Sie die versionsabhängige ProgID oder den "User friendly Name" ermitteln: In folgendem Beispiel ist "Control1" ein im Bild eingebettetes Control, das über die Type-Eigenschaft bereits die versionsunabhängige ProgID zurückgibt.

Hinweis

Da nicht jedes Control eine versionsabhängige ProgID besitzt, sollte zum Abfragen der versionsabhängigen ProgID oder des UserFriendlyName eine Fehlerbehandlung eingebaut werden. Wird keine Fehlerbehandlung verwendet, wird der Code sofort ohne Ergebnis beendet, wenn keine ProgID gefunden wird.

Ermitteln Sie die versionsabhängige ProgID wie folgt:

```
'VBS91
Dim objControl
Dim strCurrentVersion
Set objControl = ScreenItems("Control1")
strCurrentVersion = CreateObject("WScript.Shell").RegRead("HKCR\" & objControl.Type &
"\CurVer\")
```

```
MsgBox strCurrentVersion
```

Hinweis

Damit obiges Beispiel funktioniert, sollte ein Multimediacontrol ins Bild eingefügt werden.

Ermitteln Sie den User friendly Name wie folgt:

```
'VBS92
Dim objControl
Dim strFriendlyName
Set objControl = ScreenItems("Control1")
strFriendlyName = CreateObject("WScript.Shell").RegRead("HKCR\" & objControl.Type & "\")
MsgBox strFriendlyName
```

Hinweis

Damit obiges Beispiel funktioniert, sollte ein Multimediacontrol ins Bild eingefügt werden.

Beispiel

Das folgende Beispiel gibt für alle Objekte des Bildes "NewPDL1" den Typ aus:

```
'VBS93
Dim objScreen
Dim objScrItem
Dim lngIndex
Dim lngAnswer
Dim strName
lngIndex = 1
Set objScreen = HMIRuntime.Screens("NewPDL1")
For lngIndex = 1 To objScreen.ScreenItems.Count
strName = objScreen.ScreenItems(lngIndex).ObjectName
Set objScrItem = objScreen.ScreenItems(strName)
lngAnswer = MsgBox(objScrItem.Type, vbOKCancel)
If vbCancel = lngAnswer Then Exit For
Next
```

Siehe auch

ScreenItem-Objekt (Seite 134)

Objekt-Typen des Objekts ScreenItem (Seite 152)

TypeAlarmHigh-Eigenschaft

Beschreibung

TRUE, wenn der obere Grenzwert, bei dem Alarm ausgelöst wird, prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

ScreenItem-Objekt (Seite 134)

Balken (Seite 186)

TypeAlarmLow-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert, bei dem Alarm ausgelöst wird, prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

TypeLimitHigh4-Eigenschaft

Beschreibung

TRUE, wenn der obere Grenzwert "Reserve 4" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

TypeLimitHigh5-Eigenschaft

Beschreibung

TRUE, wenn der obere Grenzwert "Reserve 5" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

TypeLimitLow4-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Reserve 4" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

TypeLimitLow5-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Reserve 5" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

TypeToleranceHigh-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Toleranz high" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

TypeToleranceLow-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Toleranz low" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

TypeWarningHigh-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Warning high" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

TypeWarningLow-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Warning low" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

1.14.4.20 U

Un - Up

UnitColor-Eigenschaft

Beschreibung

Legt die Textfarbe für die Bezeichnung der Maßeinheit fest. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

UnitFont-Eigenschaft

Beschreibung

Steuert die Darstellung der Beschriftung für die Maßeinheit. Nur Lese-Zugriff.

Folgende Eigenschaften sind einstellbar:

- Schriftart
- Schriftstil
- Schriftgröße
- Effekt "durchgestrichen"
- Effekt "unterstrichen"

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

UnitOffset-Eigenschaft

Beschreibung

Das Attribut legt den Abstand des Textes für die Maßeinheit in Bezug auf die Oberkante des Objekts fest. Der Schriftzug ist nur längs des vertikalen Durchmessers der Skalenscheibe positionierbar. Der Wert der Eigenschaft ist bezogen auf die Höhe des Objekts und wird gemessen von der Oberkante des Objekts zur Unterkante des Schriftzugs.

Der Wertebereich ist 0 bis 1.

0: Die Unterkante des Schriftzugs liegt auf der oberen Begrenzung des Objekts. Der Text ist nicht mehr sichtbar, da er außerhalb des Objekts liegt.

1: Die Unterkante des Schriftzugs liegt auf der unteren Begrenzung des Objekts.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

UnitText-Eigenschaft

Beschreibung

Legt den Text für die Maßeinheit fest. Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

UnselBGColor-Eigenschaft

Beschreibung

Legt beim Objekt TextList die Hintergrundfarbe der Auswahlliste bei nicht ausgewählten Einträgen fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Textliste (Seite 208)

ScreenItem-Objekt (Seite 134)

UnselTextColor-Eigenschaft

Beschreibung

Legt beim Objekt Textliste die Textfarbe in der Auswahlliste bei nicht ausgewählten Einträgen fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Siehe auch

Textliste (Seite 208)

ScreenItem-Objekt (Seite 134)

UpdateCycle-Eigenschaft

Beschreibung

Gibt die Art und Häufigkeit für die Aktualisierung des Bildfensters in Runtime zurück. Nur Lese-Zugriff.

Siehe auch

Bildfenster (Seite 190)

ScreenItem-Objekt (Seite 134)

UpperLimit-Eigenschaft

Beschreibung

TRUE, wenn die Angabe von "UpperLimitColor" verwendet wird um die Variablenwerte (einer über "Index" referenzierten Kurve) zu kennzeichnen, die oberhalb des Wertes von "UpperLimitValue" liegen. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

UpperLimitColor-Eigenschaft

Beschreibung

Legt die Farbe fest, die verwendet wird, um die Variablenwerte (einer über "Index" referenzierten Kurve) zu kennzeichnen, die oberhalb des Wertes von "UpperLimitValue" liegen. Ob die Angabe ausgewertet wird ist abhängig vom Wert der Eigenschaft "UpperLimit". Die Angabe der Farbe erfolgt als RGB-Wert. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

ScreenItem-Objekt (Seite 134)

UpperLimitTagName-Eigenschaft

Beschreibung

Legt die Obergrenze des Kurvenbereichs fest, die automatisch aus den in PCS 7 projektierten Eigenschaften von Variablen übernommen wird. Schreib-Lese-Zugriff.

UpperLimitValue-Eigenschaft

Beschreibung

Variablenwerte (einer über "Index" referenzierten Kurve), die den Wert von "UpperLimitValue" überschreiten, werden mit der in "UpperLimitColor" festgelegten Farbe gekennzeichnet. Ob die Angabe ausgewertet wird ist abhängig vom Wert der Eigenschaft "UpperLimit".

Siehe auch

ScreenItem-Objekt (Seite 134)

WinCC Online Table Control (vor WinCC V7) (Seite 289)

WinCC Online Trend Control (vor WinCC V7) (Seite 291)

WinCC Function Trend Control (vor WinCC V7) (Seite 287)

Us

UseColumnBackColor-Eigenschaft

Spaltenfarbe verwenden / Hintergrund - UseColumnBackColor

Legt fest, welche Einstellungen für die Hintergrundfarbe der Spalten wirksam sind.

Wert	Erklärung
TRUE	Die Einstellungen der Hintergrundfarben auf der Registerkarte "Zeitspalten" bzw. in "TimeColumnBackColor" und auf der Registerkarte "Wertspalten" bzw. in "ValueColumnBackColor" sind wirksam.
FALSE	Die Einstellungen der Hintergrundfarben auf der Registerkarte "Darstellung" sind wirksam.

Das Attribut ist mit dem Namen **UseColumnBackColors** dynamisierbar. Der Datentyp ist BOOLEAN.

UseColumnForeColor-Eigenschaft

Spaltenfarbe verwenden / Schrift - UseColumnForeColor

Legt fest, welche Einstellungen für die Schriftfarben der Spalten wirksam sind.

Wert	Erklärung
TRUE	Die Einstellungen der Schriftfarben auf der Registerkarte "Zeitspalten" bzw. in "TimeColumnForeColor" und auf der Registerkarte "Wertspalten" bzw. in "ValueColumnForeColor" sind wirksam.
FALSE	Die Einstellungen der Schriftfarben der Registerkarte "Darstellung" sind wirksam.

Das Attribut ist mit dem Namen **UseColumnForeColors** dynamisierbar. Der Datentyp ist BOOLEAN.

UseMessageColor-Eigenschaft

Meldungsfarben anzeigen - UseMessageColor

Legt fest, ob die vereinbarten Meldungsfarben angezeigt werden.

Wert	Erklärung
TRUE	Die Meldungsfarben werden angezeigt.
FALSE	Die Meldungsfarben werden nicht angezeigt. Stattdessen werden die Farbeinstellungen auf der Registerkarte "Darstellung" wirksam, die für den Tabelleninhalt angegeben sind.

Das Attribut ist mit dem Namen **UseMessageColor** dynamisierbar. Der Datentyp ist BOOLEAN.

UseOnlineTags-Eigenschaft

Beschreibung

Legt fest, ob die in PCS 7 projektierten Eigenschaften von Variablen als Kurvenparameter übernommen werden. Schreib-Lese-Zugriff.

UseRangeSubstitutes-Eigenschaft

Beschreibung

TRUE, wenn für die Kurven des Kurven-Control eine eigene Skalierung der Werteachse angezeigt wird. BOOLEAN Schreib-Lese-Zugriff.

UserData-Eigenschaft

Beschreibung

Enthält den Wert, der beim Ausführen eines benutzerdefinierten Menüeintrags oder Symbols an das VB-Script übergeben wird. STRING-Schreib-Lese-Zugriff.

Beispiel

Verwenden Sie im Editor "Menüs und Symbolleisten" das Feld "Anwender Daten", um an die Prozedur einen Parameter zu übergeben.

Das folgende Beispiel zeigt die Prozedur "ActivateScreen", die einen Bildwechsel ausführt. Den Bildnamen geben Sie im Feld "Anwender Daten" ein:

```
Sub ActivateScreen (ByVal Item)
Dim objScreen
Dim strScreenName
' "UserData" contains the screen name specified
' in editor menus and toolbars.
strScreenName = Item.Userdata
HMIRuntime.BaseScreenName = strScreenName
End Sub
```

UserName-Eigenschaft

Beschreibung

Gibt den Namen des Users zurück, der das Alarm-Objekt ausgelöst hat.

Siehe auch

Alarms-Objekt (Auflistung) (Seite 120)

UserValue1-Eigenschaft

Beschreibung

Legt einen beliebigen Wert fest oder gibt ihn zurück.
Den Wert können Sie z.B. von einem Skript auswerten lassen. In Runtime wird dieser Wert weder ausgewertet noch angezeigt.

Siehe auch

ScreenItem-Objekt (Seite 134)

Sammelanzeige (Seite 205)

UserValue2-Eigenschaft

Beschreibung

Legt einen beliebigen Wert fest oder gibt ihn zurück.
Den Wert können Sie z.B. von einem Skript auswerten lassen. In Runtime wird dieser Wert weder ausgewertet noch angezeigt.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

UserValue3-Eigenschaft

Beschreibung

Legt einen beliebigen Wert fest oder gibt ihn zurück.
Den Wert können Sie z.B. von einem Skript auswerten lassen. In Runtime wird dieser Wert weder ausgewertet noch angezeigt.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

UserValue4-Eigenschaft

Beschreibung

Legt einen beliebigen Wert fest oder gibt ihn zurück.
Den Wert können Sie z.B. von einem Skript auswerten lassen. In Runtime wird dieser Wert weder ausgewertet noch angezeigt.

Siehe auch

Sammelanzeige (Seite 205)

ScreenItem-Objekt (Seite 134)

UseSelectedTitleColor-Eigenschaft

Markierungsfarbe - UseSelectedTitleColor

Legt fest, ob eine Markierungsfarbe für die Tabellenüberschriften von markierten Tabellenzellen verwendet wird.

Wert	Erklärung
TRUE	Eine Markierungsfarbe wird verwendet. Die Einstellungen zu "Hintergrund" bzw. "SelectedTitleColor" und "Schrift" bzw. "SelectedTitleForeColor" sind in Runtime wirksam.
FALSE	Eine Markierungsfarbe wird nicht verwendet. Die Einstellungen zu "Hintergrund" und "Schrift" sind in Runtime nicht wirksam.

Das Attribut ist mit dem Namen **UseSelectedTitleColor** dynamisierbar. Der Datentyp ist BOOLEAN.

UseSourceBackColors-Eigenschaft

Hintergrundfarben übernehmen - UseSourceBackColors

Legt fest, ob die Hintergrundfarbe von dem im Feld "Quelle" angegebenen Control übernommen wird.

Wert	Erklärung
TRUE	Die Hintergrundfarbe wird vom verbundenen Control übernommen.
FALSE	Die Hintergrundfarbe wird nicht vom verbundenen Control übernommen. Die Einstellung auf der Registerkarte "Darstellung" wird verwendet.

Das Attribut ist mit dem Namen **UseSourceBackColors** dynamisierbar. Der Datentyp ist BOOLEAN.

UseSourceForeColors-Eigneschaft

Schriftfarben übernehmen - UseSourceForeColors

Legt fest, ob die Schriftfarbe von dem im Feld "Quelle" angegebenen Control übernommen wird.

Wert	Erklärung
TRUE	Die Schriftfarbe wird vom verbundenen Control übernommen.
FALSE	Die Schriftfarbe wird nicht vom verbundenen Control übernommen. Die Einstellung auf der Registerkarte "Darstellung" wird verwendet.

Das Attribut ist mit dem Namen **UseSourceForeColors** dynamisierbar. Der Datentyp ist BOOLEAN.

UseTableColor2-Eigenschaft

Zeilenfarbe 2 - UseTableColor2

Legt fest, ob eine zweite Zeilenfarbe bei der Darstellung der Tabelle verwendet wird.

Wert	Erklärung
TRUE	Die Einstellungen der "Zeilenfarbe 2" werden im Wechsel mit der "Zeilenfarbe 1" verwendet.
FALSE	Die Einstellungen der "Zeilenfarbe 1" werden für alle Zeilen verwendet.

Das Attribut ist mit dem Namen **UseTableColor2** dynamisierbar. Der Datentyp ist BOOLEAN.

UseTrendNameAsLabel-Eigenschaft

UseTrendNameAsLabel

Legt fest, ob das Attribut "TrendName" oder "TrendLabel" als Bezeichnung der Kurve in Runtime verwendet wird.

Wert	Erklärung
TRUE	Das Attribut "TrendName" wird als Bezeichnung der Kurve in Runtime verwendet.
FALSE	Das Attribut "TrendLabel" wird als Bezeichnung der Kurve in Runtime verwendet.

Das Attribut ist mit dem Namen **UseTrendNameAsLabel** dynamisierbar. Der Datentyp ist BOOLEAN.

1.14.4.21 V

Val - ValueAxis

Value-Eigenschaft

Beschreibung Tag-Objekt

Gibt den Wert der Variablen beim letzten Lesezugriff aus oder den Wert, der geschrieben werden soll oder geschrieben wurde. Value steht für den Wert einer Variablen. Nach dem Aufruf der Methode "Read" enthält sie den gelesenen Wert der Variablen. Vor dem Schreiben kann der gewünschte neue Variablenwert der Eigenschaft zugewiesen werden. Nach dem Aufruf der Methode "Write" enthält die Eigenschaft den zuletzt zu schreiben versuchten Wert.

VARIANT (Schreib-Lese-Zugriff)

Beispiel

Das folgende Beispiel schreibt einen neuen Wert in die Variable "Tag1":

```
'VBS94
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
objTag.Value = 50
objTag.Write
```

Beschreibung WinCC Gauge Control

Bestimmt den Wert, auf den der Zeiger zeigt. Wertebereich: "ValueMin" bis "ValueMax".

Beschreibung DataItem-Objekt

Liefert eine Kopie des Wertes oder die Objektreferenz zurück. Des weiteren kann ein bereits hinzugefügter Wert über die Value Eigenschaft geändert werden.

Beispiel

Das Beispiel zeigt, wie ein Wert in die Auflistung aufgenommen und als Trace ausgegeben wird. Anschließend wird der Wert geändert, erneut ausgegeben und dann entfernt. Sinnvollerweise geschieht dies in verschiedenen Aktionen.

```
'VBS198
HMIRuntime.DataSet.Add "motor1", 23
HMIRuntime.Trace "motor1: " & HMIRuntime.DataSet("motor1").Value & vbNewLine
HMIRuntime.DataSet("motor1").Value = 55
HMIRuntime.Trace "motor1: " & HMIRuntime.DataSet("motor1").Value & vbNewLine
HMIRuntime.DataSet.Remove("motor1")
```

Hinweis

Bei Objektreferenzen muss sichergestellt sein, dass die Objekte multithreadfähig sind.

Siehe auch

WinCC Gauge Control (Seite 261)

Write-Methode (Seite 787)

Read-Methode (Seite 757)

Tag-Objekt (Seite 146)

DataItem-Objekt (Seite 122)

ProcessValues-Objekt (Auflistung) (Seite 133)

ValueAxisAdd-Eigenschaft

Neu - ValueAxisAdd

Legt eine neue Wertachse an.

Das Attribut ist mit dem Namen **ValueAxisAdd** dynamisierbar. Der Datentyp ist STRING.

ValueAxisAlign-Eigenschaft

Ausrichtung - ValueAxisAlign

Legt fest, wie die ausgewählte Wertachse ausgerichtet wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	links	Die ausgewählte Wertachse wird links von der Kurve angezeigt.
1	rechts	Die ausgewählte Wertachse wird rechts von der Kurve angezeigt.

Das Attribut ist mit dem Namen **ValueAxisAlign** dynamisierbar. Der Datentyp ist LONG.

ValueAxisAutoPrecisions-Eigenschaft

Nachkommastellen Automatisch - ValueAxisAutoPrecisions

Legt fest, ob die Anzahl der Nachkommastellen automatisch festgelegt wird.

Wert	Erklärung
TRUE	Anzahl der Nachkommastellen wird automatisch festgelegt. Der Wert im Feld "Nachkommastellen" bzw. "ValueAxisPrecisions" ist nicht wirksam.
FALSE	Der Wert im Feld "Nachkommastellen" bzw. "ValueAxisPrecisions" ist wirksam.

Das Attribut ist mit dem Namen **ValueAxisAutoPrecisions** dynamisierbar. Der Datentyp ist BOOLEAN.

ValueAxisAutoRange-Eigenschaft

Wertebereich Automatisch - ValueAxisAutoRange

Legt fest, ob der Wertebereich automatisch ermittelt wird.

Wert	Erklärung
TRUE	Der Wertebereich wird automatisch ermittelt.
FALSE	Der Wertebereich wird bestimmt durch die projektierten Werte in den Feldern "von" und "bis" bzw. "ValueAxisBeginValue" und "ValueAxisEndValue"

Das Attribut ist mit dem Namen **ValueAxisAutoRange** dynamisierbar. Der Datentyp ist BOOLEAN.

ValueAxisBeginValue-Eigenschaft

Wertebereich von - ValueAxisBeginValue

Legt den Anfangswert der ausgewählten Wertachse fest. Sie können den Wert projektieren, wenn die Option "Automatisch" nicht aktiviert bzw. "ValueAxisAutoRange" "FALSE" ist.

Das Attribut ist mit dem Namen **ValueAxisBeginValue** dynamisierbar. Der Datentyp ist DOUBLE.

ValueAxisColor-Eigenschaft

Farbe Wertachse - ValueAxisColor

Gibt die Farbe der Zeitachse an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Einstellung ist wirksam, wenn die Option "in Kurvenfarbe" nicht aktiviert bzw. "ValueAxisInTrendColor" "FALSE" ist.

Das Attribut ist mit dem Namen **ValueAxisColor** dynamisierbar. Der Datentyp ist LONG.

ValueAxisCount-Eigenschaft

ValueAxisCount

Gibt die Anzahl der projektierten Wertachsen an.

Das Attribut ist mit dem Namen **ValueAxisCount** dynamisierbar. Der Datentyp ist LONG.

ValueAxisEndValue-Eigenschaft

Wertebereich bis - ValueAxisEndValue

Legt den Endwert der ausgewählten Wertachse fest. Sie können den Wert projektieren, wenn die Option "Automatisch" nicht aktiviert bzw. "ValueAxisAutoRange" "FALSE" ist.

Das Attribut ist mit dem Namen **ValueAxisEndValue** dynamisierbar. Der Datentyp ist DOUBLE.

ValueAxisExponentialFormat-Eigenschaft

Exponentialdarstellung - ValueAxisExponentialFormat

Legt fest, ob die Werte der gewählten Wertachse in Exponentialdarstellung angezeigt werden.

Wert	Erklärung
TRUE	Die Werte werden in Exponentialdarstellung angezeigt.
FALSE	Die Werte werden in Dezimaldarstellung angezeigt.

Das Attribut ist mit dem Namen **ValueAxisExponentialFormat** dynamisierbar. Der Datentyp ist BOOLEAN.

ValueAxisIndex-Eigenschaft

ValueAxisIndex

Referenziert eine Wertachse. Unter Verwendung des Attributs können Sie einer bestimmten Wertachse die Werte anderer Attribute zuweisen.

Gültige Werte für "ValueAxisIndex" liegen zwischen 0 und "ValueAxisCount" minus 1. Das Attribut "ValueAxisCount" gibt die Anzahl der projektierten Wertachsen an.

Das Attribut "ValueAxisIndex" ist über das Attribut **ValueAxisRepos** dynamisierbar. Der Datentyp ist LONG.

ValueAxisInTrendColor-Eigenschaft

in Kurvenfarbe - ValueAxisInTrendColor

Legt fest, ob die ausgewählte Wertachse in der Kurvenfarbe angezeigt wird. Wenn mehrere Kurven im Kurvenfenster angezeigt werden, wird die Farbe der ersten Kurve verwendet. Die Reihenfolge der Kurven legen Sie auf der Registerkarte "Kurven" fest.

Wert	Erklärung
TRUE	Die ausgewählte Wertachse wird in der Kurvenfarbe angezeigt. Die Einstellung im Feld "Farbe" bzw. "ValueAxisColor" ist unwirksam.
FALSE	Die ausgewählte Wertachse wird in der Farbe angezeigt, die im Feld "Farbe" bzw. "ValueAxisColor" eingestellt ist.

Das Attribut ist mit dem Namen **ValueAxisInTrendColor** dynamisierbar. Der Datentyp ist BOOLEAN.

ValueAxisInTrendColor-Eigenschaft

in Kurvenfarbe - ValueAxisInTrendColor

Legt fest, ob die ausgewählte Wertachse in der Kurvenfarbe angezeigt wird. Wenn mehrere Kurven im Kurvenfenster angezeigt werden, wird die Farbe der ersten Kurve verwendet. Die Reihenfolge der Kurven legen Sie auf der Registerkarte "Kurven" fest.

Wert	Erklärung
TRUE	Die ausgewählte Wertachse wird in der Kurvenfarbe angezeigt. Die Einstellung im Feld "Farbe" bzw. "ValueAxisColor" ist unwirksam.
FALSE	Die ausgewählte Wertachse wird in der Farbe angezeigt, die im Feld "Farbe" bzw. "ValueAxisColor" eingestellt ist.

Das Attribut ist mit dem Namen **ValueAxisInTrendColor** dynamisierbar. Der Datentyp ist BOOLEAN.

ValueAxisLabel-Eigenschaft

Beschriftung - ValueAxisLabel

Legt die Beschriftung der ausgewählten Wertachse fest.

Das Attribut ist mit dem Namen **ValueAxisLabel** dynamisierbar. Der Datentyp ist STRING.

ValueAxisName-Eigenschaft

Objektname - ValueAxisName

Legt den Namen der ausgewählten Wertachse fest.

Das Attribut "ValueAxisName" ist über das Attribut **ValueAxisRename** dynamisierbar. Der Datentyp ist STRING.

ValueAxisPrecisions-Eigenschaft

Nachkommastellen - ValueAxisPrecisions

Legt fest, mit wie vielen Nachkommastellen die Werte der ausgewählten Wertachse angezeigt werden. Der Wert ist projektierbar und in Runtime wirksam, wenn die Option "Automatisch" nicht aktiviert bzw. "ValueAxisAutoPrecisions" "FALSE" ist.

Das Attribut ist mit dem Namen **ValueAxisPrecisions** dynamisierbar. Der Datentyp ist SHORT.

ValueAxisRemove-Eigenschaft

Entfernen - ValueAxisRemove

Entfernt die ausgewählte Wertachse aus der Liste.

Das Attribut ist mit dem Namen **ValueAxisRemove** dynamisierbar. Der Datentyp ist STRING.

ValueAxisRename-Eigenschaft

ValueAxisRename

Ändert den Namen der Wertachse, die über das Attribut "ValueAxisIndex" referenziert wird.

Das Attribut ist mit dem Namen **ValueAxisRename** dynamisierbar. Mit "ValueAxisRename" dynamisieren Sie auch das Attribut "ValueAxisName". Der Datentyp ist STRING.

ValueAxisRepos-Eigenschaft

Auf/Ab - ValueAxisRepos

Ändert die Reihenfolge der Wertachsen. "Auf" und "Ab" bewegen die ausgewählte Wertachse in der Liste nach oben oder unten.

Die Reihenfolge in der Liste bestimmt in Runtime die Position der Wertachse im Kurvenfenster. Wenn die Ausrichtung gleich ist und die Wertachse in der Liste weiter oben steht, wird sie an einer kurvenferneren Position dargestellt.

Das Attribut ist mit dem Namen **ValueAxisRepos** dynamisierbar. Der Datentyp ist LONG.

ValueAxisScalingType-Eigenschaft

Skalierung - ValueAxisScalingType

Legt fest, wie die ausgewählte Wertachse skaliert wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	Linear	Die ausgewählte Wertachse wird linear skaliert.
1	Logarithmisch	Die ausgewählte Wertachse wird logarithmisch skaliert.
2	Logarithmisch negiert	Die ausgewählte Wertachse wird logarithmisch negiert skaliert.

Das Attribut ist mit dem Namen **ValueAxisScalingType** dynamisierbar. Der Datentyp ist LONG.

ValueAxisTrendWindow-Eigenschaft

Kurvenfenster - ValueAxisTrendWindow

Legt fest, in welchem Kurvenfenster die ausgewählte Wertachse verwendet wird. Die zur Verfügung stehenden Kurvenfenster legen Sie auf der Registerkarte "Kurvenfenster" fest.

Das Attribut ist mit dem Namen **ValueAxisTrendWindow** dynamisierbar. Der Datentyp ist STRING.

ValueAxisVisible-Eigenschaft

Wertachsen - ValueAxisVisible

In der Liste werden die Wertachsen aufgelistet, die Sie angelegt haben. Klicken Sie auf eine Wertachse in der Liste, um die Eigenschaften anzupassen und um ein Kurvenfenster der Wertachse zuzuordnen.

Aktivieren Sie in der Liste die Wertachsen, die Sie in den Kurvenfenstern anzeigen wollen.

Das Attribut ist mit dem Namen **ValueAxisVisible** dynamisierbar. Der Datentyp ist BOOLEAN.

ValueColumn - Vi

ValueColumnAdd-Eigenschaft

Neu - ValueColumnAdd

Legt eine neue Wertspalte an.

Das Attribut ist mit dem Namen **ValueColumnAdd** dynamisierbar. Der Datentyp ist STRING.

ValueColumnAlign-Eigenschaft

Ausrichtung - ValueColumnAlign

Legt fest, wie die ausgewählte Wertspalte ausgerichtet wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	links	Die ausgewählte Wertspalte wird links angezeigt.
1	zentriert	Die ausgewählte Wertspalte wird zentriert angezeigt.
2	rechts	Die ausgewählte Wertspalte wird rechts angezeigt.

Das Attribut ist mit dem Namen **ValueColumnAlign** dynamisierbar. Der Datentyp ist LONG.

ValueColumnAlignment-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert ein Spaltenpaar. "ValueColumnAlignment" legt die Ausrichtung der Variablenwerte dieses Spaltenpaares fest.

- 0: Variablenwerte werden linksbündig eingetragen.
- 1: Variablenwerte werden zentriert eingetragen
- 2: Variablenwerte werden rechtsbündig eingetragen.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

ValueColumnAutoPrecisions-Eigenschaft

Automatisch - ValueColumnAutoPrecisions

Legt fest, ob die Anzahl der Nachkommastellen automatisch festgelegt wird.

Wert	Erklärung
TRUE	Anzahl der Nachkommastellen wird automatisch festgelegt. Der Wert im Feld "Nachkommastellen" bzw. "ValueColumnPrecisions" ist nicht wirksam.
FALSE	Der Wert im Feld "Nachkommastellen" bzw. "ValueColumnPrecisions" ist wirksam.

Das Attribut ist mit dem Namen **ValueColumnAutoPrecisions** dynamisierbar. Der Datentyp ist BOOLEAN.

ValueColumnBackColor-Eigenschaft

Hintergrundfarbe - ValueColumnBackColor

Gibt die Hintergrundfarbe der ausgewählten Wertspalte an. Mit der Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Einstellung ist wirksam, wenn auf der Registerkarte "Allgemeines" im Bereich "Spaltenfarbe verwenden" die Option "Hintergrundfarbe" aktiviert bzw. "UseColumnBackColor" "TRUE" ist.

Das Attribut ist mit dem Namen **ValueColumnBackColor** dynamisierbar. Der Datentyp ist LONG.

ValueColumnCaption-Eigenschaft

Bezeichnung - ValueColumnCaption

Legt die Bezeichnung der Wertspalte fest

Das Attribut ist mit dem Namen **ValueColumnCaption** dynamisierbar. Der Datentyp ist STRING.

ValueColumnCount-Eigenschaft

ValueColumnCount

Gibt die Anzahl der projektierten Wertspalten an.

Das Attribut ist mit dem Namen **ValueColumnCount** dynamisierbar. Der Datentyp ist LONG.

ValueColumnExponentialFormat-Eigenschaft

Exponentialdarstellung - ValueColumnExponentialFormat

Legt fest, ob die Werte der gewählten Wertspalte in Exponentialdarstellung angezeigt werden.

Wert	Erklärung
TRUE	Anzeige in Exponentialdarstellung.
FALSE	Anzeige als Dezimalzahl.

Das Attribut ist mit dem Namen **ValueColumnExponentialFormat** dynamisierbar. Der Datentyp ist BOOLEAN.

ValueColumnForeColor-Eigenschaft

Schriftfarbe - ValueColumnForeColor

Gibt die Schriftfarbe der ausgewählten Wertspalte an. Mit der Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Einstellung ist wirksam, wenn auf der Registerkarte "Allgemeines" im Bereich "Spaltenfarbe verwenden" die Option "Schriftfarbe" aktiviert bzw. "UseColumnForeColor" "TRUE" ist.

Das Attribut ist mit dem Namen **ValueColumnForeColor** dynamisierbar. Der Datentyp ist LONG.

ValueColumnHideText-Eigenschaft

ValueColumnHideText

Legt fest, ob der Inhalt der Wertspalte als Text angezeigt wird.

Wert	Erklärung
TRUE	Der Inhalt wird nicht als Text angezeigt.
FALSE	Der Inhalt wird als Text angezeigt.

Das Attribut ist mit dem Namen **ValueColumnHideText** dynamisierbar. Der Datentyp ist BOOLEAN.

ValueColumnHideTitleText-Eigenschaft

ValueColumnHideTitleText

Legt fest, ob die Überschrift der Wertspalte als Text angezeigt wird.

Wert	Erklärung
TRUE	Die Überschrift wird nicht als Text angezeigt.
FALSE	Die Überschrift wird als Text angezeigt.

Das Attribut ist mit dem Namen **ValueColumnHideTitleText** dynamisierbar. Der Datentyp ist BOOLEAN.

ValueColumnIndex-Eigenschaft

ValueColumnIndex

Referenziert eine projektierte Wertspalte. Unter Verwendung des Attributs können Sie einem bestimmten Wertspalte die Werte anderer Attribute zuweisen.

Gültige Werte für "ValueColumnIndex" liegen zwischen 0 und "ValueColumnCount" minus 1. Das Attribut "ValueColumnCount" gibt die Anzahl der projektierten Wertspalten an.

Das Attribut "ValueColumnIndex" ist über das Attribut **ValueColumnRepos** dynamisierbar. Der Datentyp ist LONG.

ValueColumnLength-Eigenschaft

Länge in Zeichen - ValueColumnLength

Legt die Breite für die ausgewählte Wertspalte fest.

Das Attribut ist mit dem Namen **ValueColumnLength** dynamisierbar. Der Datentyp ist LONG.

ValueColumnName-Eigenschaft

Objektname - ValueColumnName

Legt den Namen der ausgewählten Wertspalte fest.

Das Attribut "ValueColumnName" ist über das Attribut **ValueColumnRename** dynamisierbar. Der Datentyp ist STRING.

ValueColumnPrecisions-Eigenschaft

Nachkommastellen - ValueColumnPrecisions

Legt fest, mit wie vielen Nachkommastellen die Daten der ausgewählten Wertspalte angezeigt werden. Der Wert kann eingegeben werden, wenn die Option "Automatisch" nicht aktiviert bzw. "ValueColumnAutoPrecisions" "FALSE" ist.

Das Attribut ist mit dem Namen **ValueColumnPrecisions** dynamisierbar. Der Datentyp ist SHORT.

ValueColumnProvider-Eigenschaft

Datenversorgung - ValueColumnProvider

Legt die Datenversorgung der ausgewählten Wertspalte fest.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
1	Archivvariablen	Datenversorgung mit Archivvariablen eines Prozesswertarchivs.
2	Onlinevariablen	Datenversorgung mit Onlinevariablen aus dem Variablenhaushalt.

Das Attribut ist mit dem Namen **ValueColumnProvider** dynamisierbar. Der Datentyp ist LONG.

ValueColumnProviderCLSID-Eigenschaft

ValueColumnProviderCLSID

Zeigt die Datenversorgung der ausgewählten Wertspalte an.

Wert	Erklärung
{416A09D2-8B5A-11D2-8B81-006097A45D48}	Datenversorgung mit Archivvariablen eines Prozesswertarchivs.
{A3F69593-8AB0-11D2-A440-00A0C9DBB64E}	Datenversorgung mit Onlinevariablen aus dem Variablenhaushalt.

Das Attribut ist mit dem Namen **ValueColumnProviderCLSID** dynamisierbar. Der Datentyp ist STRING.

ValueColumnRemove-Eigenschaft

Entfernen - ValueColumnRemove

Entfernt die ausgewählte Wertspalte aus der Liste.

Das Attribut ist mit dem Namen **ValueColumnRemove** dynamisierbar. Der Datentyp ist STRING.

ValueColumnRename-Eigenschaft

ValueColumnRename

Ändert den Namen der Wertspalte, die über das Attribut "ValueColumnIndex" referenziert wird.

Das Attribut ist mit dem Namen **ValueColumnRename** dynamisierbar. Mit "ValueColumnRename" dynamisieren Sie auch das Attribut "ValueColumnName". Der Datentyp ist STRING.

ValueColumnRepos-Eigenschaft

Auf/Ab - ValueColumnRepos

Ändert die Reihenfolge der Wertspalten. "Auf" und "Ab" bewegen die ausgewählte Wertspalte in der Liste nach oben oder unten.

Wenn mehrere Wertspalten einer Zeitspalte zugeordnet sind, bestimmt die Reihenfolge in der Liste die Reihenfolge der Werstspalten hinter der Zeitspalte. Je weiter oben die Wertspalte steht, desto näher wird sie an der Zeitspalte platziert.

Die Reihenfolge der Zeitspalten mit den zugeordneten Wertspalten ändern Sie auf der Registerkarte "Zeitspalten".

Das Attribut ist mit dem Namen **ValueColumnRepos** dynamisierbar. Der Datentyp ist LONG.

ValueColumnSelectTagName-Eigenschaft

ValueColumnSelectTagName

Öffnet den Dialog zur Auswahl des Variablennamens für die Datenversorgung der Wertspalte im WinCC OnlineTableControl. Das Attribut können Programmierer nutzen, um z. B. über eine Schaltfläche dem Benutzer einen Variablennamen auswählen zu lassen.

Das Attribut ist mit dem Namen **ValueColumnSelectTagName** dynamisierbar. Der Datentyp ist BOOLEAN.

ValueColumnShowIcon-Eigenschaft

ValueColumnShowIcon

Legt fest, ob der Inhalt der Wertspalte als Symbol angezeigt wird.

Wert	Erklärung
TRUE	Der Inhalt wird als Symbol angezeigt.
FALSE	Der Inhalt wird nicht als Symbol angezeigt.

Das Attribut ist mit dem Namen **ValueColumnShowIcon** dynamisierbar. Der Datentyp ist BOOLEAN.

ValueColumnShowTitleIcon-Eigenschaft

ValueColumnShowTitleIcon

Legt fest, ob die Überschrift der Wertspalte als Symbol angezeigt wird.

Wert	Erklärung
TRUE	Die Überschrift wird als Symbol angezeigt.
FALSE	Die Überschrift wird nicht als Symbol angezeigt.

Das Attribut ist mit dem Namen **ValueColumnShowTitleIcon** dynamisierbar. Der Datentyp ist BOOLEAN.

ValueColumnSort-Eigenschaft

ValueColumnSort

Legt fest, wie die im "ValueColumnIndex" referenzierte Wertspalte sortiert wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung	Erklärung
0	nein	Keine Sortierung
1	aufsteigend	Aufsteigende Sortierung vom kleinsten zum größten Wert.
2	absteigend	Absteigende Sortierung vom größten zum kleinsten Wert.

Das Attribut ist mit dem Namen **ValueColumnSort** dynamisierbar. Der Datentyp ist LONG.

ValueColumnSortIndex-Eigenschaft

ValueColumnSortIndex

Gibt die Sortierreihenfolge der im "ValueColumnIndex" referenzierten Wertspalte an. Wenn Sie den Wert auf "0" setzen, wird das Sortierkriterium in "ValueColumnSort" entfernt.

Das Attribut ist mit dem Namen **ValueColumnSortIndex** dynamisierbar. Der Datentyp ist LONG.

ValueColumnState-Eigenschaft

ValueColumnState

Zeigt den Zustand der Datenanbindung der ausgewählten Wertspalte in Runtime an.

Das Attribut ist mit dem Namen **ValueColumnState** dynamisierbar. Der Datentyp ist LONG.

ValueColumnTagName-Eigenschaft

Variablenname - ValueColumnTagName

Zeigt den Variablennamen der angebundenen Variablen an. Über die Auswahlschaltfläche können Sie die Variablenanbindung ändern.

Das Attribut ist mit dem Namen **ValueColumnTagName** dynamisierbar. Der Datentyp ist STRING.

ValueColumnTimeColumn-Eigenschaft

Zeitspalte - ValueColumnTimeColumn

Legt fest, mit welcher Zeitspalte die ausgewählte Wertspalte dargestellt wird. Die zur Verfügung stehenden Zeitspalten legen Sie auf der Registerkarte "Zeitspalten" fest.

Das Attribut ist mit dem Namen **ValueColumnTimeColumn** dynamisierbar. Der Datentyp ist STRING.

ValueColumnVisible-Eigenschaft

Wertspalten - ValueColumnVisible

In der Liste werden die Wertspalten aufgelistet, die Sie angelegt haben. Klicken Sie auf eine Wertspalte in der Liste, um die Eigenschaften anzupassen, um die Zeitspalte zuzuordnen und um die Datenanbindung festzulegen.

Aktivieren Sie in der Liste die Wertspalten, die Sie in der Tabelle anzeigen wollen. Eine Wertspalte wird angezeigt, wenn sie mit einer Zeitspalte verbunden ist.

Das Attribut ist mit dem Namen **ValueColumnVisible** dynamisierbar. Der Datentyp ist BOOLEAN.

ValueMax-Eigenschaft

Beschreibung

Legt den Wert am Ende der Skala fest. Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

ValueMin-Eigenschaft

Beschreibung

Legt den Wert am Anfang der Skala fest. Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)

ScreenItem-Objekt (Seite 134)

Variable-Eigenschaft

Beschreibung

Die Eigenschaft "Index" referenziert ein Spaltenpaar. "Variable" legt den Namen der Variablen fest, die mit diesem Spaltenpaar verbunden ist.

Siehe auch

WinCC Online Table Control (vor WinCC V7) (Seite 289)

ScreenItem-Objekt (Seite 134)

VerticalGridLines-Eigenschaft

Vertikal - VerticalGridLines

Legt fest, ob vertikale Trennlinien angezeigt werden.

Wert	Erklärung
TRUE	Vertikale Trennlinien werden angezeigt.
FALSE	Vertikale Trennlinien werden nicht angezeigt.

Das Attribut ist mit dem Namen **VerticalGridLines** dynamisierbar. Der Datentyp ist BOOLEAN.

Visible-Eigenschaft

Beschreibung

Schaltet ein Objekt sichtbar oder unsichtbar bzw. gibt einen entsprechenden Wert aus:

- TRUE: Objekt ist sichtbar
- FALSE: Objekt ist unsichtbar

VARIANT_BOOL (Schreib-Lese-Zugriff)

Beispiel

Das folgende Beispiel setzt alle Objekte des Bildes "NewPDL1" auf unsichtbar:

```
'VBS95
Dim objScreen
Dim objScrItem
Dim lngIndex
Dim strName
lngIndex = 1
Set objScreen = HMIRuntime.Screens("NewPDL1")
For lngIndex = 1 To objScreen.ScreenItems.Count
strName = objScreen.ScreenItems(lngIndex).ObjectName
Set objScrItem = objScreen.ScreenItems(strName)
objScrItem.Visible = False
Next
```

Siehe auch

ScreenItem-Objekt (Seite 134)
Layer-Objekt (Seite 129)
HMIRuntime-Objekt (Seite 127)

1.14.4.22 W

Warning-Eigenschaft

Beschreibung

Legt den Beginn des "Warnbereichs" als Skalenwert fest. Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)
ScreenItem-Objekt (Seite 134)

WarningColor-Eigenschaft

Beschreibung

Bestimmt die Farbe des "Warnbereichs" auf der Skala. LONG Schreib-Lese-Zugriff.

Siehe auch

WinCC Gauge Control (Seite 261)
ScreenItem-Objekt (Seite 134)

WarningHigh-Eigenschaft

Beschreibung

Legt den oberen Grenzwert "Warning High" fest oder gibt ihn zurück.
Damit der Grenzwert überwacht wird, muss die Eigenschaft "CheckWarningHigh" auf TRUE gesetzt sein.
Die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften "ColorWarningHigh" und "TypeWarningHigh" festgelegt.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

WarningLow-Eigenschaft

Beschreibung

Legt den unteren Grenzwert "Warning Low" fest oder gibt ihn zurück.
Damit der Grenzwert überwacht wird, muss die Eigenschaft "CheckWarningLow" auf TRUE gesetzt sein.
Die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften "ColorWarningLow" und "TypeWarningLow" festgelegt.

Siehe auch

Balken (Seite 186)

ScreenItem-Objekt (Seite 134)

Width-Eigenschaft

Beschreibung

Setzt die Breite eines Objektes in Pixel oder gibt sie aus.

LONG

Beispiel

Das folgende Beispiel verdoppelt die Breite aller Objekte des Bildes "NewPDL1", deren Name mit "Button" beginnt:

```
'VBS96
Dim objScreen
Dim cmdButton
Dim lngIndex
Dim strName
lngIndex = 1
Set objScreen = HMIRuntime.Screens("NewPDL1")
For lngIndex = 1 To objScreen.ScreenItems.Count
'
'Get all "Buttons"
strName = objScreen.ScreenItems(lngIndex).ObjectName
If "Button" = Left(strName, 6) Then
Set cmdButton = objScreen.ScreenItems(strName)
cmdButton.Width = cmdButton.Width * 2
End If
```

```
Next
```

Siehe auch

Height-Eigenschaft (Seite 421)

ScreenItem-Objekt (Seite 134)

WinCCStyle-Eigenschaft

Beschreibung

Legt fest, in welchem Stil das Objekt dargestellt wird.

Benutzerdefiniert	Stellt das Objekt entsprechend den eigenen Einstellungen dar.
Global	Stellt das Objekt im global eingestellten Design dar.
Windows-Stil	Stelt das Objekt im Windows-Stil dar.

WindowBorder-Eigenschaft

Beschreibung

TRUE, wenn das Fenster in Runtime mit Rahmen dargestellt wird. Nur Lese-Zugriff.

Siehe auch

Bildfenster (Seite 190)

Applikationsfenster (Seite 185)

ScreenItem-Objekt (Seite 134)

WindowPositionMode-Eigenschaft

Beschreibung

Legt die Position und Skalierung des Bildfensters auf dem Bildschirm fest. Es wirkt sich nur aus, wenn das Attribut "Unabhängiges Fenster" (IndependentWindow) auf "ja" (TRUE) gesetzt ist.

Standard	Das Bildfenster wird in Originalgröße an der projektierten Stelle auf dem Bildschirm positioniert.
Zentrieren	Das Bildfenster wird in Originalgröße mittig auf dem Bildschirm positioniert.
Maximieren	Das Bildfenster wird auf die Größe des Bildschirms skaliert.

WindowsStyle-Eigenschaft

Beschreibung

Legt fest, ob das Objekt im Windows-Stil von WinCC Version 6.2 dargestellt wird. Es ist nur auswählbar, wenn als aktuelles Design "WinCC Classic" ausgewählt ist.

TRUE, wenn das Objekt im Windows-Stil von WinCC Version 6.2 dargestellt wird.

FALSE, wenn das Objekt nicht im Windows-Stil von WinCC Version 6.2 dargestellt wird.

WindowsStyle-Eigenschaft

Beschreibung

TRUE, wenn das Objekt dem allgemeinen Windows Stil entspricht (z.B. graue Schaltflächen ohne Rahmen). BOOLEAN Schreib-Lese-Zugriff. Hinweis:

- Wenn diese Eigenschaft auf "True" gesetzt ist, werden Eigenschaften, die dem Windows-Stil nicht entsprechen, ignoriert (z.B. "BorderWidth").
- Andererseits führt die Festlegung einer "BorderWidth" oder einer nicht grauen Hintergrundfarbe dazu, dass "WindowsStyle" den Wert "False" erhält.
- Eine Ausnahme bilden hierbei die Blinkattribute: Die Festlegung von Blinkattributen führt nicht automatisch zur Deaktivierung des Attributs "WindowsStyle".

Siehe auch

Slider (Seite 221)

Button (Seite 212)

ScreenItem-Objekt (Seite 134)

WindowType-Eigenschaft

Beschreibung

Legt den Verwendungszweck des Meldefensters fest.

- 0 - Meldeliste: Zur Darstellung der aktuell anstehenden Meldungen.
- 1 - Kurzzeitarchivliste: Zur Darstellung der archivierten Meldungen.
- 2 - Langzeitarchivliste: Zur Darstellung der archivierten Meldungen.
- 3 - Sperrliste: Zur Darstellung der aktuell gesperrten Meldungen.
- 4 - Hitliste:Zur Darstellung der statistische Informationen der Meldungen.

Siehe auch

WinCC Alarm Control (vor WinCC V7) (Seite 285)

ScreenItem-Objekt (Seite 134)

WithAxes-Eigenschaft

Beschreibung

TRUE, wenn die Skala angezeigt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

WithLabels-Eigenschaft

Beschreibung

TRUE, wenn die Skalenbeschriftung angezeigt wird. BOOLEAN Schreib-Lese-Zugriff.

Siehe auch

WinCC Slider Control (Seite 278)

ScreenItem-Objekt (Seite 134)

1.14.4.23 X - Z

XAxisColor-Eigenschaft (vor WinCC V7)

Beschreibung

Mit dem Attribut legen Sie die verwendete Farbe der gemeinsamen X-Achse fest. Die Angabe der Farbe erfolgt als RGB-Wert. LONG Schreib-Lese-Zugriff.

X/YAxisAdd-Eigenschaft

Neu - X/YAxisAdd

Legt eine neue X-Achse bzw. Y-Achse an.

Das Attribut für die X-Achse ist mit dem Namen **XAxisAdd** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisAdd** dynamisierbar.

Der Datentyp ist STRING.

X/YAxisAlign-Eigenschaft

Ausrichtung - X/YAxisAlign

Legt fest, wie die ausgewählte Achse ausgerichtet wird.

Folgende Einstellungen stehen für die X-Achse zur Verfügung:

Wert	Beschreibung	Erklärung
0	unten	Die ausgewählte X-Achse wird unter der Kurve angezeigt.
1	oben	Die ausgewählte X-Achse wird über der Kurve angezeigt.

Das Attribut für die X-Achse ist mit dem Namen **XAxisAlign** dynamisierbar. Der Datentyp ist LONG.

Folgende Einstellungen stehen für die Y-Achse zur Verfügung:

Wert	Beschreibung	Erklärung
0	links	Die ausgewählte Y-Achse wird links von der Kurve angezeigt.
1	rechts	Die ausgewählte Y-Achse wird rechts von der Kurve angezeigt.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisAlign** dynamisierbar. Der Datentyp ist LONG.

X/YAxisAutoPrecisions-Eigenschaft

Nachkommastellen Automatisch - X/YAxisAutoPrecisions

Legt fest, ob die Anzahl der Nachkommastellen automatisch festgelegt wird.

Wert	Erklärung
TRUE	Die Anzahl der Nachkommastellen wird automatisch festgelegt. Der Wert im Feld "Nachkommastellen" bzw. "X/YAxisPrecisions" ist nicht wirksam.
FALSE	Der Wert im Feld "Nachkommastellen" bzw. "X/YAxisPrecisions" ist wirksam.

Das Attribut für die X-Achse ist mit dem Namen **XAxisAutoPrecisions** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisAutoPrecisions** dynamisierbar.

Der Datentyp ist BOOLEAN.

X/YAxisAutoRange-Eigenschaft

Wertebereich Automatisch - X/YAxisAutoRange

Legt fest, ob der Wertebereich der ausgewählten Achse automatisch ermittelt wird.

Wert	Erklärung
TRUE	Der Wertebereich wird automatisch ermittelt.
FALSE	Der Wertebereich wird bestimmt durch die projektierten Werte in den Feldern "von" und "bis" bzw. "X/YAxisBeginValue" und "X/YAxisEndValue".

Das Attribut für die X-Achse ist mit dem Namen **XAxisAutoRange** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisAutoRange** dynamisierbar.

Der Datentyp ist BOOLEAN.

X/YAxisBeginValue-Eigenschaft

Wertebereich von - X/YAxisBeginValue

Legt den unteren Wertebereich der ausgewählten Achse fest. Sie können den Wert projektieren, wenn die Option "Automatisch" nicht aktiviert bzw. "X/YAxisAutoRange" "FALSE" ist.

Das Attribut für die X-Achse ist mit dem Namen **XAxisBeginValue** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisBeginValue** dynamisierbar.

Der Datentyp ist DOUBLE.

X/YAxisColor-Eigenschaft

Farbe X/Y-Achse - X/YAxisColor

Gibt die Farbe der ausgewählten Achse an. Über die Schaltfläche öffnen Sie den Dialog "Farbauswahl" zur Auswahl der Farbe.

Die Einstellung ist nur wirksam, wenn das Feld "in Kurvenfarbe" nicht aktiviert bzw. "X/YAxisInTrendColor" "FALSE" ist.

Das Attribut für die X-Achse ist mit dem Namen **XAxisColor** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisColor** dynamisierbar.

Der Datentyp ist LONG.

X/YAxisEndValue-Eigenschaft

Wertebereich bis - X/YAxisEndValue

Legt den oberen Wertebereich der ausgewählten Achse fest. Sie können den Wert projektieren, wenn die Option "Automatisch" nicht aktiviert bzw. "X/YAxisAutoRange" "FALSE" ist.

Das Attribut für die X-Achse ist mit dem Namen **XAxisEndValue** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisEndValue** dynamisierbar.

Der Datentyp ist DOUBLE.

X/YAxisExponentialFormat-Eigenschaft

Exponentialdarstellung - X/YAxisExponentialFormat

Legt fest, ob die Werte der gewählten Achse in Exponentialdarstellung angezeigt werden.

Wert	Erklärung
TRUE	Die Werte werden in Exponentialdarstellung angezeigt.
FALSE	Die Werte werden in Dezimaldarstellung angezeigt.

Das Attribut für die X-Achse ist mit dem Namen **XAxisExponentialFormat** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisExponentialFormat** dynamisierbar.

Der Datentyp ist BOOLEAN.

X/YAxisInTrendColor-Eigenschaft

in Kurvenfarbe - X/YAxisInTrendColor

Legt fest, ob die ausgewählte Achse in der Kurvenfarbe angezeigt wird. Wenn mehrere Kurven im Kurvenfenster angezeigt werden, wird die Farbe der ersten Kurve verwendet. Die Reihenfolge der Kurven legen Sie auf der Registerkarte "Kurven" fest.

Wert	Erklärung
TRUE	Die ausgewählte Achse wird in der Kurvenfarbe angezeigt. Die Einstellung im Feld "Farbe" bzw. "X/YAxisColor" ist unwirksam.
FALSE	Die ausgewählte Achse wird in der Farbe angezeigt, die im Feld "Farbe" bzw. "X/YAxisColor" eingestellt ist.

Das Attribut für die X-Achse ist mit dem Namen **XAxisInTrendColor** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisInTrendColor** dynamisierbar.

Der Datentyp ist BOOLEAN.

X/YAxisLabel-Eigenschaft

Beschriftung - X/YAxisLabel

Legt den Text fest, mit dem die ausgewählte Achse beschriftet wird.

Das Attribut für die X-Achse ist mit dem Namen **XAxisLabel** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisLabel** dynamisierbar.

Der Datentyp ist STRING.

X/YAxisName-Eigenschaft

Objektname - X/YAxisName

Legt den Namen der ausgewählten Achse fest.

Für die X-Achse ist das Attribut "XAxisName" über das Attribut **XAxisRename** dynamisierbar.

Für die Y-Achse ist das Attribut "YAxisName" über das Attribut **YAxisRename** dynamisierbar.

Der Datentyp ist STRING.

X/YAxisPrecisions-Eigenschaft

Nachkommastellen - X/YAxisPrecisions

Legt fest, mit wie vielen Nachkommastellen die Werte der ausgewählten Achse angezeigt werden. Der Wert ist projektierbar und in Runtime wirksam, wenn die Option "Automatisch" nicht aktiviert bzw. "X/YAxisAutoPrecisions" "FALSE" ist.

Das Attribut für die X-Achse ist mit dem Namen **XAxisPrecisions** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisPrecisions** dynamisierbar.

Der Datentyp ist SHORT.

X/YAxisRemove-Eigenschaft

Entfernen - X/YAxisRemove

Entfernt die ausgewählte Achse aus der Liste.

Das Attribut für die X-Achse ist mit dem Namen **XAxisRemove** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisRemove** dynamisierbar.

Der Datentyp ist STRING.

X/YAxisRepos-Eigenschaft

Auf/Ab - X/YAxisRepos

Ändert die Reihenfolge der Achsen. "Auf" und "Ab" bewegen die ausgewählte Achse in der Liste nach oben oder unten.

Die Reihenfolge in der Liste bestimmt in Runtime die Position der Achse im Kurvenfenster. Wenn die Ausrichtung gleich ist und die Achse in der Liste weiter oben steht, wird sie an einer kurvenferneren Position dargestellt.

Das Attribut für die X-Achse ist mit dem Namen **XAxisRepos** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisRepos** dynamisierbar.

Der Datentyp ist LONG.

X/YAxisScalingType-Eigenschaft

Skalierung - X/YAxisScalingType

Legt fest, wie die ausgewählte Achse skaliert wird.

Folgende Einstellungen stehen zur Verfügung:

Wert	Beschreibung
0	Linear
1	Logarithmisch
2	Logarithmisch negiert

Das Attribut für die X-Achse ist mit dem Namen **XAxisScalingType** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisScalingType** dynamisierbar.

Der Datentyp ist LONG.

X/YAxisTrendWindow-Eigenschaft

Kurvenfenster - X/YAxisTrendWindow

Legt fest, in welchem Kurvenfenster die ausgewählte Achse verwendet wird. Die zur Verfügung stehenden Kurvenfenster legen Sie auf der Registerkarte "Kurvenfenster" fest.

Das Attribut für die X-Achse ist mit dem Namen **XAxisTrendWindow** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisTrendWindow** dynamisierbar.

Der Datentyp ist STRING.

X/YAxisVisible-Eigenschaft

X/Y-Achsen - X/YAxisVisible

In der Liste werden die Achsen aufgelistet, die Sie angelegt haben. Klicken Sie auf eine Achse in der Liste, um die Eigenschaften anzupassen und um die Achse einem Kurvenfenster zuzuordnen.

Aktivieren Sie in der Liste die Achsen, die Sie in den Kurvenfenstern anzeigen wollen.

Das Attribut für die X-Achse ist mit dem Namen **XAxisVisible** dynamisierbar.

Das Attribut für die Y-Achse ist mit dem Namen **YAxisVisible** dynamisierbar.

Der Datentyp ist BOOLEAN.

XAxisCount-Eigenschaft

XAxisCount

Gibt die Anzahl der projektierten X-Achsen an.

Das Attribut ist mit dem Namen **XAxisCount** dynamisierbar. Der Datentyp ist LONG.

XAxisIndex-Eigenschaft

XAxisIndex

Referenziert eine projektierte X-Achse. Unter Verwendung des Attributs können Sie einer bestimmten X-Achse die Werte anderer Attribute zuweisen.

Gültige Werte für "Index" liegen zwischen 0 und "XAxisCount" minus 1. Das Attribut "XAxisCount" gibt die Anzahl der projektierten X-Achsen an.

Das Attribut "XAxisIndex" ist über das Attribut **XAxisRepos** dynamisierbar. Der Datentyp ist LONG.

XAxisRename-Eigenschaft

XAxisRename

Ändert den Namen der X-Achse, die über das Attribut "XAxisIndex" referenziert wird.

Das Attribut ist mit dem Namen **XAxisRename** dynamisierbar. Mit "XAxisRename" dynamisieren Sie auch das Attribut "XAxisName". Der Datentyp ist STRING.

YAxisCount-Eigenschaft

YAxisCount

Gibt die Anzahl der projektierten Y-Achsen an.

Das Attribut ist mit dem Namen **YAxisCount** dynamisierbar. Der Datentyp ist LONG.

YAxisIndex-Eigenschaft

YAxisIndex

Referenziert eine projektierte Y-Achse. Unter Verwendung des Attributs können Sie einer bestimmten Y-Achse die Werte anderer Attribute zuweisen.

Gültige Werte für "YAxisIndex" liegen zwischen 0 und "YAxisCount" minus 1. Das Attribut "YAxisCount" gibt die Anzahl der projektierten Y-Achsen an.

Das Attribut "YAxisIndex" ist über das Attribut **YAxisRepos** dynamisierbar. Der Datentyp ist LONG.

YAxisRename-Eigenschaft

YAxisRename

Ändert den Namen der Y-Achse, die über das Attribut "YAxisIndex" referenziert wird.

Das Attribut ist mit dem Namen **YAxisRename** dynamisierbar. Mit "YAxisRename" dynamisieren Sie auch das Attribut "YAxisName". Der Datentyp ist STRING.

ZeroPoint-Eigenschaft

Beschreibung

Legt die Lage des Nullpunktes des Balkens fest oder gibt sie zurück.
Geben Sie den Wert in % zur Gesamtbalkenhöhe an. Der Nullpunkt kann auch außerhalb des dargestellten Bereiches liegen.
Die Eigenschaft "ScalingType" muss auf "2" und "Scaling" auf TRUE gesetzt sein.

Siehe auch

ScreenItem-Objekt (Seite 134)

Balken (Seite 186)

ZeroPointValue-Eigenschaft

Beschreibung

Legt den Wert des Nullpunktes der Skalenanzeige fest.

Legt den absoluten Wert für den Nullpunkt fest oder gibt ihn zurück.

Siehe auch

Balken (Seite 186)

3D-Balken (Seite 181)

ScreenItem-Objekt (Seite 134)

Zoom-Eigenschaft

Beschreibung

Setzt den Zoomfaktor eines Bildes bzw. Bildfensters oder liest ihn aus.

Wenn der angegebene Zoomfaktor kleiner ist als der Minimalwert, so wird automatisch der Zoomfaktor auf den Minimalwert gesetzt. Wenn der angegebene Zoomfaktor größer ist als der Maximalwert, so wird der Zoomfaktor auf den Maximalwert gesetzt.

Der Minimalwert des Zoomfaktors liegt bei 2 Prozent, der Maximalwert bei 800 Prozent.

Beim Screen-Objekt wird der Zoomfaktor als numerischer Wert und beim Bildfenster-Objekt in Prozent angegeben.

Beispiel

Das folgende Beispiel verdoppelt den Zoomfaktor des aktuellen Bildes:

```
'VBS97
HMIRuntime.ActiveScreen.Zoom = HMIRuntime.ActiveScreen.Zoom * 2
```

Siehe auch

Bildfenster (Seite 190)

Screen-Objekt (Seite 140)

1.14.5 Methoden

1.14.5.1 Methoden

Übersicht

Mit den Methoden, die Sie auf die einzelnen Objekte anwenden, können Sie z.B. Variablenwerte zur weiteren Verarbeitung auslesen, oder Diagnosemeldungen in Runtime ausgeben.

Verfügbare Methoden in VBS

Activate	GetStatusbarElement	MoveToNext	ShowInfoText
ActivateDynamic	GetStatusbarElementCollection	MoveToNextLine	ShowLockDialog
Add	GetTimeAxis	MoveToNextPage	ShowLockList
AttachDB	GetTimeAxisCollection	MoveToPrevious	ShowLongTermArchiveList
CalculateStatistic	GetTimeColumn	MoveToPreviousLine	ShowMessageList
CopyRows	GetTimeColumnCollection	MoveToPreviousPage	ShowPercentageAxis
CreateTagSet	GetToolbarButton	NextColumn	ShowPropertyDialog
CutRows	GetToolbarButtonCollection	NextTrend	ShowSelectArchive
DeactivateDynamic	GetTrend	OneToOneView	ShowSelection
DeleteRows	GetTrendCollection	PasteRows	ShowSelectionDialog
DetachDB	GetTrendWindow	PreviousColumn	ShowSelectTimeBase
Edit	GetTrendWindowCollection	PreviousTrend	ShowShortTermArchiveList
Export	GetValueAxis	Print	ShowSort
GetColumn	GetValueAxisCollection	QuitHorn	ShowSortDialog
GetColumnCollection	GetValueColumn	QuitSelected	ShowTagSelection
GetHitlistColumn	GetValueColumnCollection	QuitVisible	ShowTimebaseDialog
GetHitlistColumnCollection	GetXAxis	Read	ShowTimeSelection
GetMessageBlock	GetXAxisCollection	ReadTags	ShowTrendSelection
GetMessageBlockCollection	GetYAxis	Refresh	StartStopUpdate
GetMessageColumn	GetYAxisCollection	Remove	Stop
GetMessageColumnCollection	HideAlarm	RemoveAll	Trace
GetOperatorMessage	Item-Methode	Restore	UnhideAlarm

GetOperatorMessageCollection	LockAlarm	SelectedStatisticArea	UnlockAlarm
GetRulerBlock	LoopInAlarm	ServerExport	Write
GetRulerBlockCollection	MoveAxis	ServerImport	WriteTags
GetRulerColumn	MoveRuler (Seite 746)	ShowColumnSelection	ZoomArea
GetRulerColumnCollection	MoveToFirst	ShowComment	ZoomInOut
GetRulerData	MoveToFirstLine	ShowDisplayOptionsDialog	ZoomInOutTime
GetStatisticAreaColumn	MoveToFirstPage	ShowEmergencyQuitDialog	ZoomInOutValues
GetStatisticAreaColumnCollection	MoveToLast	ShowHelp	ZoomInOutX
GetStatisticResultColumn	MoveToLastLine	ShowHideList	ZoomInOutY
GetStatisticResultColumnCollection	MoveToLastPage	ShowHitList	ZoomMove

1.14.5.2 Methoden A bis E

Activate-Methode

Funktion

Aktiviert das angegebene Bild bzw. Bildelement.

Hinweis

Fokuszuweisungen sollten nicht im ButtonDown-Ereignis projektiert werden. Da der Fokus beim ButtonDown-Ereignis explizit angefordert wird, kann es zu ungültigen Zuständen kommen.

Syntax

```
Ausdruck.Activate
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Screen" oder "ScreenItem" zurückgibt.

Parameter

--

Beispiele

Das folgende Beispiel zeigt die Verwendung beim Typ "Screen":

```
'VBS98
Dim objScreen
MsgBox HMIRuntime.ActiveScreen.ObjectName    'Output of active screen
Set objScreen = HMIRuntime.Screens("ScreenWindow1")
objScreen.Activate    'Activate "ScreenWindow1"
MsgBox HMIRuntime.ActiveScreen.ObjectName    'New output of active screen
```

Das folgende Beispiel zeigt die Verwendung beim Typ "ScreenItem":

```
'VBS158
MsgBox HMIRuntime.ActiveScreen.ActiveScreenItem.ObjectName  'Output of active screen item
HMIRuntime.ActiveScreen.ScreenItems("IOField1").Activate
MsgBox HMIRuntime.ActiveScreen.ActiveScreenItem.ObjectName  'New output of active screen
item
```

Siehe auch

ScreenItem-Objekt (Seite 134)

Screen-Objekt (Seite 140)

ActivateDynamic-Methode

Funktion

Aktiviert dynamisch einen Trigger für die festgelegte Eigenschaft und mit dem festgelegten Zyklus während Runtime. Bei jeder Aktivierung des Triggers kann ein anderer Aktivierungszyklus verwendet werden.

Beispiele für die Methode finden Sie im Kapitel "VBS zum Erstellen von Prozeduren und Aktionen > Aktionen erstellen und bearbeiten > Trigger > Animationstrigger".

Syntax

```
Ausdruck.ActivateDynamic (ByVAl bstrPropertyName As String, ByVal
bstrCycleName As String)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

Parameter	Beschreibung
bstrPropertyName	Name der Eigenschaft, auf die sich der Trigger bezieht.
bstrCycleName	Name des Aktivierungszyklus, z.B. "CycleTime1s".

Siehe auch

Animationstrigger (Seite 73)

Add-Methode

Beschreibung für TagSet-Objekt

Fügt eine Variable zur Auflistung hinzu. Die Variable kann über den Namen oder über eine Referenz zu einem Tag-Objekt hinzugefügt werden.

Syntax

```
Ausdruck.Add [Tag]
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "TagSet" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
Tag	Name einer WinCC Variable oder Referenz auf ein Tag-Objekt, das der Auflistung hinzugefügt werden soll.

Beispiel

Im folgenden Beispiel wird ein TagSet-Objekt erzeugt und eine Variable hinzugefügt.

```
'VBS170
Dim group
Set group = HMIRuntime.Tags.CreateTagSet
group.Add "Motor1"
```

Tag-Objekte können auch wie folgt aufgenommen werden.

```
'VBS171
Dim Tag
Set Tag = HMIRuntime.Tags("Motor2")
Dim group2
Set group2 = HMIRuntime.Tags.CreateTagSet
group2.Add Tag
```

Beschreibung für DataSet-Objekt

Fügt der Auflistung einen Wert oder eine Objektreferenz hinzu.

Hinweis

Das DataSet-Objekt unterstützt keine Klassen.
Objekte vom Typ Screen, Screens, ScreenItem, ScreenItems, Tag und TagSet können nicht in die DataSet-Auflistung aufgenommen werden.
Bei Objektreferenzen muss sichergestellt sein, dass die Objekte multithreadfähig sind.

Syntax

```
Ausdruck.Add [vtName], [vtUserData]
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "DataSet" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vtName	Name, unter dem der Wert oder die Variable zu der Auflistung hinzugefügt werden soll.
vtUserData	Wert, der in der Auflistung hinzugefügt werden soll.

Beispiel

In diesem Beispiel wird ein Wert in die DataSet-Auflistung aufgenommen.

```
'VBS172
HMIRuntime.DataSet.Add "Motor1",23
```

Siehe auch

TagSet-Objekt (Auflistung) (Seite 151)

DataSet-Objekt (Auflistung) (Seite 125)

AttachDB-Methode

Funktion

Führt die Tastenfunktion "Backup verbinden" des Control aus.

Syntax

```
Ausdruck.AttachDB()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

CalculateStatistic-Methode

Funktion

Führt die Tastenfunktion "Statistik berechnen" des OnlineTrendControl und OnlineTableControl aus.

Syntax

```
Ausdruck.CalculateStatistic()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

CopyRows-Methode

Funktion

Führt die Tastenfunktion "Zeilen kopieren" des Control aus.

Syntax

```
Ausdruck.CopyRows()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

Create-Methode

Funktion

Erzeugt ein neues Alarm-Objekt.

Syntax

```
Ausdruck.Create (VARIANT vtApplication)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Alarm" zurückgibt.

Parameter

```
VARIANT
```

Parameter	Beschreibung
vtApplication	Name des Alarm-Objekts (optional)

Siehe auch

Alarms-Objekt (Auflistung) (Seite 120)

CreateTagSet-Methode

Funktion

Erzeugt ein neues TagSet-Objekt. Dieses Objekt kann für optimierte Multi-Tag-Zugriffe benutzt werden.

Syntax

```
Ausdruck.CreateTagSet()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "TagSet" zurückgibt.

Parameter

VARIANT

Beispiel

Das folgende Beispiel zeigt wie man ein TagSet-Objekt erzeugt.

```
'VBS168
'Build a Reference to the TagSet Object
Dim group
Set group = HMIRuntime.Tags.CreateTagSet
```

Siehe auch

TagSet-Objekt (Auflistung) (Seite 151)

Tags-Objekt (Auflistung) (Seite 149)

CutRows-Methode

Funktion

Führt die Tastenfunktion "Zeilen ausschneiden" des UserArchiveControl aus.

Syntax

```
Ausdruck.CutRows()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

DeactivateDynamic-Methode

Funktion

Deaktiviert den mit der "ActivateDynamic"-Methode verwendeten Trigger für die festgelegte Eigenschaft während Runtime.

Syntax

```
Ausdruck.DeactivateDynamic(ByVal bstrPropertyName As String)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

String

Parameter	Beschreibung
bstrPropertyName	Name der Eigenschaft, auf die sich der Trigger bezieht.

DeleteRows-Methode

Funktion

Führt die Tastenfunktion "Zeilen löschen" des UserArchiveControl aus.

Syntax

```
Ausdruck.DeleteRows()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

DetachDB-Methode

Funktion

Führt die Tastenfunktion "Backup trennen" des Control aus.

Syntax

```
Ausdruck.DetachDB()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Edit-Methode

Funktion

Führt die Tastenfunktion "Bearbeiten" des OnlineTableControl aus.

Syntax

```
Ausdruck.Edit()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Export-Methode

Funktion

Führt die Tastenfunktion "Archiv exportieren" oder "Daten exportieren" des Control aus.

Syntax

```
Ausdruck.Export()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

1.14.5.3 Methoden Get

GetColumn-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Spalten-Objekt des WinCC UserArchiveControl als Typ "ICCAxUAColumn" zurück.

Syntax

```
Ausdruck.GetColumn(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der Spalte des UserArchiveControl.

Beispiel

```
'VBS312
Dim ctrl
Dim objColumn
Set ctrl = ScreenItems("UAControl")
Set objColumn = ctrl.GetColumn("Field1")
objColumn.Length = 30
Set objColumn = ctrl.GetColumn(3)
objColumn.Align = 2
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "Column" schreiben Sie z. B. "objColumn.Align" statt "objColumn.ColumnAlign".

Siehe auch

Column-Objekt (Auflistung) (Seite 230)

GetColumnCollection-Methode

Funktion

Gibt die Auflistung aller Spalten-Objekte des WinCC UserArchiveControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetColumnCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS313
Dim ctrl
Dim coll
Dim field
Set ctrl = ScreenItems("UAControl")
Set coll = ctrl.GetColumnCollection
HMIRuntime.Trace "Number of fields:" & coll.Count & vbCrLf
For Each field In coll
 HMIRuntime.Trace field.Name & vbCrLf
 HMIRuntime.Trace field.Type & vbCrLf
 HMIRuntime.Trace field.Length & vbCrLf
 HMIRuntime.Trace field.Caption & vbCrLf
```

```
Next
```

Siehe auch

Column-Objekt (Auflistung) (Seite 230)

GetHitlistColumn-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Spalten-Objekt der Hitliste des WinCC AlarmControl als Typ "ICCAxMessageColumn" zurück.

Syntax

```
Ausdruck.GetHitlistColumn(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der Spalte der Hitliste

Beispiel

```
'VBS314
Dim ctrl
Dim objHitlistColumn
Set ctrl = ScreenItems("AlarmControl")
Set objHitlistColumn = ctrl.GetHitlistColumn("Date")
objHitlistColumn.Sort = 2
Set objHitlistColumn = ctrl.GetHitlistColumn("AverageComeGo")
objHitlistColumn.Visible = FALSE
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "HitlistColumn" schreiben Sie z. B. "objHitlistColumn.Visible" statt "objHitlistColumn.HitlistColumnVisible".

Siehe auch

HitlistColumn-Objekt (Auflistung) (Seite 231)

GetHistlistColumnCollection-Methode

Funktion

Gibt die Auflistung aller Spalten-Objekte der Hitliste des WinCC AlarmControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetHitlisteColumnCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS315
Dim ctrl
Dim coll
Dim hitlistcol
Set ctrl = ScreenItems("AlarmControl")
Set coll = ctrl.GetHitlistColumnCollection
HMIRuntime.Trace "Number of hitlist columns:" & coll.Count & vbCrLf
For Each hitlistcol In coll
 HMIRuntime.Trace hitlistcol.Index & vbCrLf
 HMIRuntime.Trace hitlistcol.Name & vbCrLf
 HMIRuntime.Trace hitlistcol.Sort & vbCrLf
 HMIRuntime.Trace hitlistcol.SortIndex & vbCrLf
Next
```

Siehe auch

HitlistColumn-Objekt (Auflistung) (Seite 231)

GetMessageBlock-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Meldeblock-Objekt des WinCC AlarmControl als Typ "ICCAxMessageBlock" zurück.

Syntax

```
Ausdruck.GetMessageBlock(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name des Meldeblocks.

Beispiel

```
'VBS316
Dim ctrl
```

```
Dim objMsgBlock
Set ctrl = ScreenItems("AlarmControl")
Set objMsgBlock = ctrl.GetMessageBlock("Date")
objMsgBlock.Align = 2
Set objMsgBlock = ctrl.GetMessageBlock("Number")
objMsgBlock.LeadingZeros = 4
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "MessageBlock" schreiben Sie z. B. "objMsgBlock.Align" statt "objMsgBlock.MessageBlockAlign".

Siehe auch

MessageBlock-Objekt (Auflistung) (Seite 232)

GetMessageBlockCollection-Methode

Funktion

Gibt die Auflistung aller Meldeblock-Objekte des WinCC AlarmControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetMessageBlockCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS317
Dim ctrl
Dim coll
Dim msgblock
Set ctrl = ScreenItems("AlarmControl")
Set coll = ctrl.GetMessageBlockCollection
For Each msgblock In coll
 msgblock.Align = 1
 msgblock.Length = 12
 msgblock.Selected = TRUE
Next
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "MessageBlock" schreiben Sie z. B. "msgblock.Align" statt "msgblock.MessageBlockAlign".

Siehe auch

MessageBlock-Objekt (Auflistung) (Seite 232)

GetMessageColumn-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Spalten-Objekt des WinCC AlarmControl als Typ "ICCAxMessageColumn" zurück.

Syntax

```
Ausdruck.GetMessageColumn(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der Spalte in der Meldeliste.

Beispiel

```
'VBS318
Dim ctrl
Dim objMessColumn
Set ctrl = ScreenItems("AlarmControl")
Set objMessColumn = ctrl.GetMessageColumn("Date")
objMessColumn.Visible = FALSE
Set objMessColumn = ctrl.GetMessageColumn("Number")
objMessColumn.Sort = 1
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "MessageColumn" schreiben Sie z. B. "objMessColumn.Visible" statt "objMessColumn.MessageColumnVisible".

Siehe auch

MessageColumn-Objekt (Auflistung) (Seite 233)

GetMessageColumnCollection-Methode

Funktion

Gibt die Auflistung aller Spalten-Objekte des WinCC AlarmControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetMessageColumnCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS319
Dim ctrl
Dim coll
Dim msgcol
Set ctrl = ScreenItems("AlarmControl")
Set coll = ctrl.GetMessageColumnCollection
HMIRuntime.Trace "Number of message columns:" & coll.Count & vbCrLf
For Each msgcol In coll
 HMIRuntime.Trace msgcol.Index & vbCrLf
 HMIRuntime.Trace msgcol.Name & vbCrLf
 HMIRuntime.Trace msgcol.Sort & vbCrLf
 HMIRuntime.Trace msgcol.SortIndex & vbCrLf
Next
```

Siehe auch

MessageColumn-Objekt (Auflistung) (Seite 233)

GetOperatorMessage-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Bedienmeldungs-Objekt des WinCC AlarmControl als Typ "ICCAxOperatorMessage" zurück.

Syntax

```
Ausdruck.GetOperatorMessage(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der Bedienmeldung

Beispiel

```
'VBS320
Dim ctrl
Dim objOpMess
Set ctrl = ScreenItems("AlarmControl")
Set objOpMess = ctrl.GetOperatorMessage(0)
objOpMess.Source1 = "Number"
objOpMess.SourceType1 = 1
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "OperatorMessage" schreiben Sie z. B. "objOpMess.Source1" statt "objOpMess.OperatorMessageSource1".

Siehe auch

OperatorMessage-Objekt (Auflistung) (Seite 234)

GetOperatorMessageCollection-Methode

Funktion

Gibt die Auflistung aller Bedienmeldungs-Objekte des WinCC AlarmControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetOperatorMessageCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS321
Dim ctrl
Dim coll
Dim opmsg
Set ctrl = ScreenItems("AlarmControl")
Set coll = ctrl.GetOperatorMessageCollection
For Each opmsg In coll
 HMIRuntime.Trace opmsg.Index & vbCrLf
 HMIRuntime.Trace opmsg.Name & vbCrLf
 HMIRuntime.Trace opmsg.Number & vbCrLf
 HMIRuntime.Trace opmsg.Selected & vbCrLf
Next
```

Siehe auch

OperatorMessage-Objekt (Auflistung) (Seite 234)

GetRow-Methode

Funktion

Gibt das mit der Zeilennummer bezeichnete Zeilen-Objekt der auf Tabellen basierenden Controls als Typ "ICCAxDataRow" zurück.

Syntax

```
Ausdruck.GetRow(ByVal IRow As Long)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

Long

Parameter	Beschreibung
IRow	Nummer der gewünschten Zeile des Controls.

Beispiel

```
'VBS356
Dim ctrl
Dim lIndex
Dim lCellIndex
Set ctrl = ScreenItems("UAControl")
Set coll = ctrl.GetRowCollection
'enumerate and trace out row numbers
For lIndex = 1 To coll.Count
 HMIRuntime.trace "Row: " & (ctrl.GetRow(lIndex).RowNumber) & "  "
 'enumerate and trace out column titles and cell texts
 For lCellIndex = 1 To ctrl.GetRow(lIndex).CellCount
  HMIRuntime.trace ctrl.GetColumn(lCellIndex -1).Name & " "
  HMIRuntime.trace ctrl.GetRow(lIndex).CellText(lCellIndex) & "  "
 Next
 HMIRuntime.trace vbNewLine
Next
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "Row" schreiben Sie z. B. "objRow.CellCount" statt "objRow.RowCellCount".

Siehe auch

Row-Objekt (Auflistung) (Seite 235)

GetRowCollection-Methode

Funktion

Gibt die Auflistung aller Zeilen-Objekte der auf Tabellen basierenden Controls als Typ "ICCAxDataRowCollection" zurück.

Syntax

```
Ausdruck.GetRowCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

Eigenschaften der ICCAxDataRowCollection

Die ICCAxDataRowCollection bezieht sich auf Runtime-Daten. Die Daten können nur gelesen werden. Ein Hinzufügen und Ändern der Daten ist nicht möglich.

Folgende Eigenschaften sind bei der ICCAxDataRowCollection verfügbar:

- Count - ermittelt die Anzahl der Zeilen in der Collection
- Item - Zugriff auf eine einzelne Zeile innerhalb der Collection über die Zeilennummer. Die Nummerierung läuft von 1 bis Count. Zurückgegeben wird ein Row-Objekt.

Beispiel

```
'VBS357
Dim ctrl
Dim coll
Dim lIndex
Dim lCellIndex
Set ctrl = ScreenItems("AlarmControl")
Set coll = ctrl.GetRowCollection
HMIRuntime.Trace "Number of message rows:" & coll.Count & vbCrLf
'enumerate and trace out row numbers
For lIndex = 1 To coll.Count
 HMIRuntime.Trace "Row: " & (ctrl.GetRow(lIndex).RowNumber) & "  "
 'enumerate and trace out column titles and cell texts
 For lCellIndex = 1 To ctrl.GetRow(lIndex).CellCount
  HMIRuntime.Trace ctrl.GetMessageColumn(lCellIndex -1).Name & " "
  HMIRuntime.Trace ctrl.GetRow(lIndex).CellText(lCellIndex) & "  "
 Next
 HMIRuntime.Trace vbNewLine
Next
```

Siehe auch

Row-Objekt (Auflistung) (Seite 235)

GetRulerBlock-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Block-Objekt des WinCC RulerControl als Typ "ICCAxRulerBlock" zurück.

Syntax

```
Ausdruck.GetRulerBlock(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name des Blocks im RulerControl

Beispiel

```
'VBS322
Dim ctrl
Dim objRulerBlock
Set ctrl = ScreenItems("RulerControl")
Set objRulerBlock = ctrl.GetRulerBlock(0)
objRulerBlock.Caption = "RulerBlock1"
Set objRulerBlock = ctrl.GetRulerBlock("Name")
objRulerBlock.Length = 10
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "RulerBlock" schreiben Sie z. B. "objRulerBlock.Caption" statt "objRulerBlock.BlockCaption".

Siehe auch

RulerBlock-Objekt (Auflistung) (Seite 236)

GetRulerBlockCollection-Methode

Funktion

Gibt die Auflistung aller Block-Objekte des WinCC RulerControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetRulerBlockCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS323
Dim ctrl
Dim coll
Dim rulerblock
Set ctrl = ScreenItems("RulerControl")
Set coll = ctrl.GetRulerBlockCollection
For Each rulerblock In coll
 rulerblock.Align = 1
 rulerblock.Length = 12
Next
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "RulerBlock" schreiben Sie z. B. "rulerblock.Align" statt "rulerblock.RulerBlockAlign".

Siehe auch

RulerBlock-Objekt (Auflistung) (Seite 236)

GetRulerColumn-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Spalten-Objekt des WinCC RulerControl als Typ "ICCAxRulerColumn" zurück.

Syntax

```
Ausdruck.GetRulerColumn(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der Spalte des RulerControl.

Beispiel

```
'VBS324
Dim ctrl
Dim objRulercol
Set ctrl = ScreenItems("RulerControl")
Set objRulercol = ctrl.GetRulerColumn("Name")
objRulercol.Sort = 0
Set objRulercol = ctrl.GetRulerColumn("ValueY")
objRulercol.Visible = FALSE
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "RulerColumn" schreiben Sie z. B. "objRulercol.Visible" statt "objRulercol.ColumnVisible".

Siehe auch

RulerColumn-Objekt (Auflistung) (Seite 236)

GetRulerColumnCollection-Methode

Funktion

Gibt die Auflistung aller Spalten-Objekte des WinCC RulerControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetRulerColumnCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS325
Dim ctrl
Dim coll
Dim rulercol
Set ctrl = ScreenItems("RulerControl")
Set coll = ctrl.GetRulerColumnCollection
HMIRuntime.Trace "Number of ruler columns:" & coll.Count & vbCrLf
For Each rulercol In coll
 HMIRuntime.Trace rulercol.Index & vbCrLf
 HMIRuntime.Trace rulercol.Name & vbCrLf
 HMIRuntime.Trace rulercol.Sort & vbCrLf
 HMIRuntime.Trace rulercol.SortIndex & vbCrLf
Next
```

Siehe auch

RulerColumn-Objekt (Auflistung) (Seite 236)

GetRulerData-Methode

Funktion

Gibt den Wert der aufgerufenen Kurve an der Linealposition zurück.

Syntax

```
Ausdruck.GetRulerData(ByVal RulerIndex As Long, pvValue As Variant,
Optional pvTimeStamp As Variant, Optional pvFlags As Varian) Long
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Trend" zurückgibt.

Parameter

Parameter	Beschreibung
RulerIndex	0 =Lineal
pvValue	Wert der X-Achse
pvTimeStamp	Zeitpunkt bzw. Wert der Y-Achse
pvFlags	Qualitycode

Beispiel

```
'VBS326
Dim ctrl
Dim objTrend
Dim objIOField1
Dim objIOField2
Dim value
Dim time
Set ctrl = ScreenItems( "Control1" )
Set objTrend = ctrl.GetTrend( "Trend 1" )
Set objIOField1 = ScreenItems( "I/O Field1" )
Set objIOField2 = ScreenItems( "I/O Field2" )
objTrend.GetRulerData 0, value, time
objIOField1.OutputValue = value
objIOField2.OutputValue = time
```

GetSelectedRow-Methode

Funktion

Gibt das markierte Zeilen-Objekt der auf Tabellen basierenden Controls als Typ "ICCAxDataRow" zurück.

Syntax

```
Ausdruck.GetSelectedRow()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

Beispiel

```
'VBS358
Dim ctrl
 Dim lCellIndex
 Dim lCellCount
 Dim headingRow
 Dim selectedRow
 Set ctrl = ScreenItems("TableControl")
 Set headingRow = ctrl.GetRow(0)
 Set selectedRow = ctrl.GetSelectedRow
 lCellCount = headingRow.CellCount
```

```
'enumerate and trace out column titles and cell texts
For lCellIndex = 1 To lCellCount
 HMIRuntime.trace headingRow.CellText(lCellIndex) & ":  "
 HMIRuntime.trace selectedRow.CellText(lCellIndex)
   HMIRuntime.trace vbNewLine
Next
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "Row" schreiben Sie z. B. "objRow.CellCount" statt "objRow.RowCellCount".

Siehe auch

Row-Objekt (Auflistung) (Seite 235)

GetSelectedRows-Methode

Funktion

Gibt bei Mehrfachselektion die markierten Zeilen-Objekte der auf Tabellen basierenden Controls als Typ "ICCAxDataRow" zurück.

Syntax

```
Ausdruck.GetSelectedRows()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

Beispiel

```
'VBS359
Dim ctrl
 Dim lCellIndex
 Dim lCellCount
 Dim lRowIndex
```

```
Dim lRowCount
Dim headingRow
Dim selectedRow
Dim selectedRows
Set ctrl = ScreenItems("TableControl")
Set headingRow = ctrl.GetRow(0)
Set selectedRows = ctrl.GetSelectedRows
lCellCount = headingRow.CellCount
lRowCount = selectedRows.Count
'enumerate selected rows
For lRowIndex = 1 To lRowCount
 Set selectedRow = selectedRows(lRowIndex)
 HMIRuntime.Trace "Row number: " & CStr(lRowIndex) & vbNewLine
 'enumerate and trace out column titles and cell texts
 For lCellIndex = 1 To lCellCount
  HMIRuntime.trace headingRow.CellText(lCellIndex) & ":  "
  HMIRuntime.trace selectedRow.CellText(lCellIndex)
    HMIRuntime.trace vbNewLine
 Next
Next
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "Row" schreiben Sie z. B. "objRow.CellCount" statt "objRow.RowCellCount".

Siehe auch

Row-Objekt (Auflistung) (Seite 235)

GetStatisticAreaColumn-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Spalten-Objekt des Statistikbereichfensters des WinCC RulerControl als Typ "ICCAxRulerColumn" zurück.

Syntax

```
Ausdruck.GetStatisticAreaColumn(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der Spalte des Statistikbereichfensters.

Beispiel

```
'VBS327
Dim ctrl
Dim objStatAreaCol
Set ctrl = ScreenItems("RulerControl")
Set objStatAreaCol = ctrl.GetStatisticAreaColumn("DatasourceY")
objStatAreaCol.Visible = FALSE
Set objStatAreaCol = ctrl.GetStatisticAreaColumn("ValueY(LL)")
objStatAreaCol.Sort = 1
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "StatisticAreaColumn" schreiben Sie z. B. "objStatAreaCol.Visible" statt "objStatAreaCol.ColumnVisible".

Siehe auch

StatisticAreaColumn-Objekt (Auflistung) (Seite 237)

GetStatisticAreaColumnCollection-Methode

Funktion

Gibt die Auflistung aller Spalten-Objekte des Statistikbereichfensters des WinCC RulerControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetStatisticAreaColumnCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS328
Dim ctrl
Dim coll
Dim statcol
Set ctrl = ScreenItems("RulerControl")
Set coll = ctrl.GetStatisticAreaColumnCollection
HMIRuntime.Trace "Number of statistic Area columns:" & coll.Count & vbCrLf
For Each statcol In coll
 HMIRuntime.Trace statcol.Index & vbCrLf
 HMIRuntime.Trace statcol.Name & vbCrLf
 HMIRuntime.Trace statcol.Sort & vbCrLf
 HMIRuntime.Trace statcol.SortIndex & vbCrLf
Next
```

Siehe auch

StatisticAreaColumn-Objekt (Auflistung) (Seite 237)

GetStatisticResultColumn-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Spalten-Objekt des Statistikfensters des WinCC RulerControl als Typ "ICCAxRulerColumn" zurück.

Syntax

```
Ausdruck.GetStatisticResultColumn(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der Spalte des Statistikfensters.

Beispiel

```
'VBS329
Dim ctrl
Dim objStatResCol
Set ctrl = ScreenItems("RulerControl")
Set objStatResCol = ctrl.GetStatisticResultColumn("MaxValue")
objStatResCol.Visible = FALSE
Set objStatResCol = ctrl.GetStatisticResultColumn("Average")
objStatResCol.Sort = 2
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "StatisticResultColumn" schreiben Sie z. B. "objStatResCol.Visible" statt "objStatResCol.ColumnVisible".

Siehe auch

StatisticResultColumn-Objekt (Auflistung) (Seite 238)

GetStatisticResultColumnCollection-Methode

Funktion

Gibt die Auflistung aller Spalten-Objekte des Statistikfensters des WinCC RulerControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetStatisticResultColumnCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS330
Dim ctrl
Dim coll
Dim statcol
Set ctrl = ScreenItems("RulerControl")
Set coll = ctrl.GetStatisticResultColumnCollection
HMIRuntime.Trace "Number of statistic result columns:" & coll.Count & vbCrLf
For Each statcol In coll
 HMIRuntime.Trace statcol.Index & vbCrLf
 HMIRuntime.Trace statcol.Name & vbCrLf
 HMIRuntime.Trace statcol.Sort & vbCrLf
 HMIRuntime.Trace statcol.SortIndex & vbCrLf
Next
```

Siehe auch

StatisticResultColumn-Objekt (Auflistung) (Seite 238)

GetStatusbarElement-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Element der Statuszeile des Controls als Typ "ICCAxStatusbarElement" zurück.

Syntax

```
Ausdruck.GetStatusbarElement(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name des Elements der Statuszeile.

Beispiel

```
'VBS331
Dim ctrl
Dim objStatusBar
Set ctrl = ScreenItems("Control1")
Set objStatusBar = ctrl.GetStatusbarElement(1)
objStatusBar.Visible = FALSE
Set objStatusBar = ctrl.GetStatusbarElement(3)
objStatusBar.Width = 10
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "StatusbarElement" schreiben Sie z. B. "objStatusBar.Visible" statt "objStatusBar.StatusbarElementVisible".

Siehe auch

StatusbarElement-Objekt (Auflistung) (Seite 239)

GetStatusbarElementCollection-Methode

Funktion

Gibt die Auflistung aller Elemente der Statuszeile des Controls als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetStatusbarElementCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS332
Dim ctrl
Dim coll
Dim statelement
Set ctrl = ScreenItems.Item("Control1")
Set coll = ctrl.GetStatusbarElementCollection
HMIRuntime.Trace "Number of statusbar elements:" & coll.Count & vbCrLf
For Each statelement In coll
 HMIRuntime.Trace statelement.Name & vbCrLf
 HMIRuntime.Trace statelement.Width & vbCrLf
 HMIRuntime.Trace statelement.Text & vbCrLf
Next
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "StatusbarElement" schreiben Sie z. B. "statelement.Name" statt "statelement.StatusbarElementName".

Siehe auch

StatusbarElement-Objekt (Auflistung) (Seite 239)

GetTimeAxis-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Zeitachsen-Objekt des WinCC OnlineTrendControl als Typ "ICCAxTimeAxis" zurück.

Syntax

```
Ausdruck.GetTimeAxis(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der Zeitachse.

Beispiel

```
'VBS333
Dim ctrl
Dim objTimeAxis
Set ctrl = ScreenItems("OnlineTrendControl")
Set objTimeAxis = ctrl.GetTimeAxis(1)
objTimeAxis.Visible = FALSE
Set objTimeAxis = ctrl.GetTimeAxis("axis 2")
objTimeAxis.Label = "Time axis 2"
objTimeAxis.DateFormat = "dd.MM.yy"
objTimeAxis.TimeFormat = "HH:mm:ss.ms"
objTimeAxis.RangeType = 2
objTimeAxis.BeginTime = "06.04.2010 9:33:18"
objTimeAxis.MeasurePoints = 100
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "TimeAxis" schreiben Sie z. B. "objTimeAx.Visible" statt "objTimeAx.TimeAxisVisible".

Siehe auch

TimeAxis-Objekt (Auflistung) (Seite 239)

GetTimeAxisCollection-Methode

Funktion

Gibt dieAuflistung aller Zeitachsen-Objekte des WinCC OnlineTrendControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetTimeAxisCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS334
```

```
Dim ctrl
Dim objTrendWnd
Dim objTimeAxis1
Dim objTimeAxis2
Dim objTrend
Set ctrl = ScreenItems("OnlineTrendControl")
Set objTrendWnd = ctrl.GetTrendWindowCollection.AddItem("myWindow")
Set objTimeAxis1 = ctrl.GetTimeAxisCollection.AddItem("TimeAxis2010")
Set objTimeAxis2 = ctrl.GetTimeAxisCollection.AddItem("TimeAxis2011")
objTimeAxis1.TrendWindow = objTrendWnd.Name
objTimeAxis1.Label = "2010"
objTimeAxis1.RangeType = 1
objTimeAxis1.BeginTime = "01.01.2010 0:00:00"
objTimeAxis1.EndTime = "31.12.2010 11:59:59"
objTimeAxis2.TrendWindow = objTrendWnd.Name
objTimeAxis2.Label = "2011"
objTimeAxis2.RangeType = 1
objTimeAxis2.BeginTime = "01.01.2011 0:00:00"
objTimeAxis2.EndTime = "31.12.2011 11:59:59"
Set objTrend = ctrl.GetTrendCollection.AddItem("myTrend1")
objTrend.TrendWindow = objTrendWnd.Name
objTrend.TimeAxis = objTimeAxis1.Name
Set objTrend = ctrl.GetTrendCollection.AddItem("myTrend2")
objTrend.TrendWindow = objTrendWnd.Name
objTrend.TimeAxis = objTimeAxis2.Name
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "TimeAxis" schreiben Sie z. B. "objTimeAxis1.Label" statt "objTimeAxis1.TimeAxisLabel".

Siehe auch

TimeAxis-Objekt (Auflistung) (Seite 239)

GetTimeColumn-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Zeitspalten-Objekt des WinCC OnlineTableControl als Typ "ICCAxTimeColumn" zurück.

Syntax

```
Ausdruck.GetTimeColumn(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der Zeitspalte.

Beispiel

```
'VBS335
Dim ctrl
Dim objTimeCol
Set ctrl = ScreenItems("TableControl")
Set objTimeCol = ctrl.GetTimeColumn("Timecolumn1")
objTimeCol.ShowDate = FALSE
Set objTimeCol = ctrl.GetTimeColumn("Timecolumn2")
objTimeCol.Visible = FALSE
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "TimeColumn" schreiben Sie z. B. "objTimeColumn.ShowDate" statt "objTimeColumn.TimeColumnShowDate".

Siehe auch

TimeColumn-Objekt (Auflistung) (Seite 240)

GetTimeColumnCollection-Methode

Funktion

Gibt die Auflistung aller Zeitspalten-Objekte des WinCC OnlineTableControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetTimeColumnCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS336
Dim ctrl
Dim objTimeCol1
Dim objTimeCol2
Dim coll
Dim timecol
Set ctrl = ScreenItems("TableControl")
Set objTimeCol1 = ctrl.GetTimeColumnCollection.AddItem("TimeColumn2010")
Set objTimeCol2 = ctrl.GetTimeColumnCollection.AddItem("TimeColumn2011")
objTimeCol1.Caption = "2010"
objTimeCol1.RangeType = 1
objTimeCol1.BeginTime = "01.01.2010 0:00:00"
objTimeCol1.EndTime = "31.12.2010 11:59:59"
objTimeCol2.Caption = "2011"
objTimeCol2.RangeType = 0
objTimeCol2.BeginTime = "01.01.2011 0:00:00"
objTimeCol2.TimeRangeFactor = 1
objTimeCol2.TimeRangeBase = 3600000
Set coll = ctrl.GetTimeColumnCollection
For Each timecol In coll
 timecol.Align = 1
 timecol.Length = 12
 timecol.BackColor = RGB(240,240,0)
 timecol.ForeColor = RGB(130,160,255)
Next
```

Siehe auch

TimeColumn-Objekt (Auflistung) (Seite 240)

GetToolbarButton-Methode

Funktion

Gibt die mit Namen oder Index bezeichnete Tastenfunktion der Symbolleiste des Control als Typ "ICCAxToolbarButton" zurück.

Syntax

```
Ausdruck.GetToolbarButton(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der Tastenfunktion der Symbolleiste.

Beispiel

```
'VBS337
Dim ctrl
Set ctrl = ScreenItems("Control1")
Dim toolbu
Set toolbu = ctrl.GetToolbarButton ("ShowHelp")
HMIRuntime.Trace "Name: " & toolbu.Name & vbCrLf
HMIRuntime.Trace "Index: " & toolbu.Index & vbCrLf
HMIRuntime.Trace "Hotkey: " & toolbu.HotKey & vbCrLf
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "ToolbarButton" schreiben Sie z. B. "toolbu.Index" statt "toolbu.ToolbarButtonIndex".

Siehe auch

ToolbarButton-Objekt (Auflistung) (Seite 241)

GetToolbarButtonCollection-Methode

Funktion

Gibt die Auflistung aller Tastenfunktionen der Symbolleiste des Control als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetToolbarButtonCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Methoden sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS338
Dim ctrl
Dim coll
Dim toolbu
Set ctrl = ScreenItems("Control1")
Set coll = ctrl.GetToolbarButtonCollection
HMIRuntime.Trace "Number of toolbar buttons:" & coll.Count & vbCrLf
For Each toolbu In coll
 HMIRuntime.Trace toolbu.Name & vbCrLf
 HMIRuntime.Trace "Hotkey: " & toolbu.HotKey & vbCrLf
 HMIRuntime.Trace "Authorization: " & toolbu.PasswordLevel & vbCrLf
Next
```

Siehe auch

ToolbarButton-Objekt (Auflistung) (Seite 241)

GetTrend-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Kurven-Objekt des WinCC OnlineTrendControl oder WinCC FunctionTrendControl als Typ "ICCAxTrend" bzw. "ICCAxFunctionTrend" zurück.

Syntax

```
Ausdruck.GetTrend(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der Kurve.

Beispiel

```
'VBS339
Dim ctrl
Dim objTrend
Set ctrl = ScreenItems("OnlineTrendControl")
Set objTrend = ctrl.GetTrend("Trend 1")
objTrend.PointStyle = 1
objTrend.LineWidth = 4
Set objTrend = ctrl.GetTrend(2)
objTrend.Provider = 1
objTrend.TagName = "Archive\ArchiveTag2"
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "Trend" schreiben Sie z. B. "objTrend.PointStyle" statt "objTrend.TrendPointStyle".

Siehe auch

Trend-Objekt (Auflistung) (Seite 242)

GetTrendCollection-Methode

Funktion

Gibt die Auflistung aller Kurven-Objekte des WinCC OnlineTrendControl oder WinCC FunctionTrendControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetTrendCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS340
Dim ctrl
Dim objTrendWnd
Dim objTimeAxis
Dim objValAxis
Dim objTrend
Set ctrl = ScreenItems("OnlineTrendControl")
Set objTrendWnd = ctrl.GetTrendWindowCollection.AddItem("myWindow")
Set objTimeAxis = ctrl.GetTimeAxisCollection.AddItem("myTimeAxis")
Set objValAxis = ctrl.GetValueAxisCollection.AddItem("myValueAxis")
objTimeAxis.TrendWindow = objTrendWnd.Name
objValAxis.TrendWindow = objTrendWnd.Name
Set objTrend = ctrl.GetTrendCollection.AddItem("myTrend1")
objTrend.Provider = 1
objTrend.TagName = "Archive\ArchiveTag1"
objTrend.TrendWindow = objTrendWnd.Name
objTrend.TimeAxis = objTimeAxis.Name
```

```
objTrend.ValueAxis = objValAxis.Name
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "Trend" schreiben Sie z. B. "objTrend.TagName" statt "objTrend.TrendTagName".

Siehe auch

Trend-Objekt (Auflistung) (Seite 242)

GetTrendWindow-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Kurvenfenster-Objekt des WinCC OnlineTrendControl oder WinCC FunctionTrendControl als Typ "ICCAxTrendWindow" zurück.

Syntax

```
Ausdruck.GetTrendWindow(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name des Kurvenfensters.

Beispiel

```
'VBS341
Dim ctrl
Dim objTrendWnd
Set ctrl = ScreenItems("OnlineTrendControl")
Set objTrendWnd = ctrl.GetTrendWindow(1)
```

```
objTrendWnd.Visible = FALSE
Set objTrendWnd = ctrl.GetTrendWindow("trend window 2")
objTrendWnd.VerticalGrid = TRUE
objTrendWnd.FineGrid = TRUE
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "TrendWindow" schreiben Sie z. B. "objTrendWnd.Visible" statt "objTrendWnd.TrendWindowVisible".

Siehe auch

TrendWindow-Objekt (Auflistung) (Seite 244)

GetTrendWindowCollection-Methode

Funktion

Gibt die Auflistung aller Kurvenfenster-Objekte des WinCC OnlineTrendControl oder WinCC FunctionTrendControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetTrendWindowCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS342
Dim ctrl
Dim objTrendWnd
Dim objTimeAxis
Dim objValAxis
Set ctrl = ScreenItems("OnlineTrendControl")
Set objTrendWnd = ctrl.GetTrendWindowCollection.AddItem("myWindow")
Set objTimeAxis = ctrl.GetTimeAxisCollection.AddItem("myTimeAxis")
Set objValAxis = ctrl.GetValueAxisCollection.AddItem("myValueAxis")
objTimeAxis.TrendWindow = objTrendWnd.Name
objValAxis.TrendWindow = objTrendWnd.Name
```

Siehe auch

TrendWindow-Objekt (Auflistung) (Seite 244)

GetValueAxis-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Wertachsen-Objekt des WinCC OnlineTrendControl als Typ "ICCAxValueAxis" zurück.

Syntax

```
Ausdruck.GetValueAxis(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der Wertachse.

Beispiel

```
'VBS343
Dim ctrl
Dim objValAxis
Set ctrl = ScreenItems("OnlineTrendControl")
```

```
Set objValAxis = ctrl.GetValueAxis(1)
objValAxis.Visible = FALSE
Set objValAxis = ctrl.GetValueAxis("axis 2")
objValAxis.Label = "Value axis 2"
objValAxis.ScalingType = 0
objValAxis.Precisions = 2
objValAxis.AutoRange = TRUE
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "ValueAxis" schreiben Sie z. B. "objValueAx.Visible" statt "objValueAx.ValueAxisVisible".

Siehe auch

ValueAxis-Objekt (Auflistung) (Seite 245)

GetValueAxisCollection-Methode

Funktion

Gibt die Auflistung aller Wertachsen-Objekte des WinCC OnlineTrendControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetValueAxisCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS344
Dim ctrl
Dim objTrendWnd
Dim objValAxis1
Dim objValAxis2
Dim objTrend
Set ctrl = ScreenItems("OnlineTrendControl")
Set objTrendWnd = ctrl.GetTrendWindowCollection.AddItem("myWindow")
Set objValAxis1 = ctrl.GetValueAxisCollection.AddItem("myValueAxis1")
Set objValAxis2 = ctrl.GetValueAxisCollection.AddItem("myValueAxis2")
objValAxis1.TrendWindow = objTrendWnd.Name
objValAxis1.Label = "Value1"
objValAxis2.TrendWindow = objTrendWnd.Name
objValAxis2.inTrendColor = TRUE
Set objTrend = ctrl.GetTrendCollection.AddItem("myTrend1")
objTrend.TrendWindow = objTrendWnd.Name
objTrend.ValueAxis = objValAxis1.Name
Set objTrend = ctrl.GetTrendCollection.AddItem("myTrend2")
objTrend.TrendWindow = objTrendWnd.Name
objTrend.ValueAxis = objValAxis2.Name
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "ValueAxis" schreiben Sie z. B. "objValueAxis1.Label" statt "objValueAxis1.ValueAxisLabel".

Siehe auch

ValueAxis-Objekt (Auflistung) (Seite 245)

GetValueColumn-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Wertspalten-Objekt des WinCC OnlineTableControl als Typ "ICCAxValueColumn" zurück.

Syntax

```
Ausdruck.GetValueColumn(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der Wertspalte des OnlineTableControl.

Beispiel

```
'VBS345
Dim ctrl
Dim objValueColumn
Set ctrl = ScreenItems("TableControl")
Set objValueColumn = ctrl.GetValueColumn("Valuecolumn1")
objValueColumn.Precisions = 4
Set objValueColumn = ctrl.GetValueColumn(2)
objValueColumn.ExponentialFormat = TRUE
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "ValueColumn" schreiben Sie z. B. "objValueColumn.Precisions" statt "objValueColumn.ValueColumnPrecisions".

Siehe auch

ValueColumn-Objekt (Auflistung) (Seite 246)

GetValueColumnCollection-Methode

Funktion

Gibt die Auflistung aller Wertspalten-Objekte des WinCC OnlineTableControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetValueColulmnCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS346
Dim ctrl
Dim objValCol1
Dim objValCol2
Dim coll
Dim valcol
Set ctrl = ScreenItems("TableControl")
Set objValCol1 = ctrl.GetValueColumnCollection.AddItem("ValueColumn1")
Set objValCol2 = ctrl.GetValueColumnCollection.AddItem("ValueColumn2")
objValCol1.Caption = "Value Archive"
objValCol1.Provider = 1
objValCol1.TagName = "ProcessValueArchive\arch1"
objValCol1.TimeColumn = "TimeColumn1"
objValCol2.Caption = "Value Tag"
objValCol2.Provider = 2
objValCol2.TagName = "tagxx"
objValCol2.TimeColumn = "TimeColumn2"
Set coll = ctrl.GetValueColumnCollection
For Each valcol In coll
 valcol.Align = 2
 valcol.Length = 10
 valcol.AutoPrecisions = TRUE
Next
```

Siehe auch

ValueColumn-Objekt (Auflistung) (Seite 246)

GetXAxis-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete X-Achsen-Objekt des WinCC FunctionTrendControl als Typ "ICCAxValueAxis" zurück.

Syntax

```
Ausdruck.GetXAxis(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der X-Achse.

Beispiel

```
'VBS347
Dim ctrl
Dim objXAx
Set ctrl = ScreenItems("FunctionTrendControl")
Set objXAx = ctrl.GetXAxis(1)
objXAx.Visible = FALSE
Set objXAx = ctrl.GetXAxis("axis 2")
objXAx.Label = "X axis 2"
objXAx.ScalingType = 0
objXAx.Precisions = 2
objXAx.Color = RGB(109,109,109)
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "XAxis" schreiben Sie z. B. "objXAx.Visible" statt "objXAx.XAxisVisible".

Siehe auch

XAxis-Objekt (Auflistung) (Seite 247)

GetXAxisCollection-Methode

Funktion

Gibt die Auflistung aller X-Achsen-Objekte des WinCC FunctionTrendControl als Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetXAxisCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS348
Dim ctrl
Dim objXAxis1
Dim objXAxis2
Dim coll
Dim axes
Set ctrl = ScreenItems("FunctionTrendControl")
Set objXAxis1 = ctrl.GetXAxisCollection.AddItem("myXAxis1")
objXAxis1.Label = "temperature"
Set objXAxis2 = ctrl.GetXAxisCollection.AddItem("myXAxis2")
objXAxis2.Label = "pressure"
Set coll = ctrl.GetXAxisCollection
```

```
HMIRuntime.Trace "Number of XAxis:" & coll.Count & vbCrLf
For Each axes In coll
 HMIRuntime.Trace axes.Name & vbCrLf
 HMIRuntime.Trace axes.Label & vbCrLf
Next
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "XAxis" schreiben Sie z. B. "objXAxis1.Label" statt "objXAxis1.XAxisLabel".

Siehe auch

XAxis-Objekt (Auflistung) (Seite 247)

GetYAxis-Methode

Funktion

Gibt das mit Namen oder Index bezeichnete Y-Achsen-Objekt des WinCC FunctionTrendControl als Typ "ICCAxValueAxis" zurück.

Syntax

```
Ausdruck.GetYAxis(ByVal vIndex As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
vIndex	Index oder Name der Y-Achse.

Beispiel

```
'VBS349
Dim ctrl
Dim objYAx
```

```
Set ctrl = ScreenItems("FunctionTrendControl")
Set objYAx = ctrl.GetYAxis(1)
objYAx.Visible = FALSE
Set objYAx = ctrl.GetYAxis("axis 2")
objYAx.Label = "Y axis 2"
objYAx.Align = 0
objYAx.Precisions = 3
objYAx.EndValue = 90.000
objYAx.BeginValue = 10.000
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "YAxis" schreiben Sie z. B. "objYAx.Visible" statt "objYAx.YAxisVisible".

Siehe auch

YAxis-Objekt (Auflistung) (Seite 247)

GetYAxisCollection-Methode

Funktion

Gibt die Auflistung aller Y-Achsen-Objekte des WinCC FunctionTrendControl vom Typ "ICCAxCollection" zurück.

Syntax

```
Ausdruck.GetYAxisCollection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Eigenschaften und Funktionen der ICCAxCollection

Folgende Eigenschaften sind bei der ICCAxCollection verfügbar:

- Count
- Item

Folgende Funktionen sind bei der ICCAxCollection verfügbar:

- AddItem(vName) As Object
- RemoveItem(vIndex)

Beispiel

```
'VBS350
Dim ctrl
Dim objYAxis1
Dim objYAxis2
Dim coll
Dim axes
Set ctrl = ScreenItems("FunctionTrendControl")
Set objYAxis1 = ctrl.GetXAxisCollection.AddItem("myYAxis1")
objYAxis1.Label = "temperature"
Set objYAxis2 = ctrl.GetXAxisCollection.AddItem("myYAxis2")
objYAxis2.Label = "pressure"
Set coll = ctrl.GetYAxisCollection
HMIRuntime.Trace "Number of YAxis:" & coll.Count & vbCrLf
For Each axes In coll
 HMIRuntime.Trace axes.Name & vbCrLf
 HMIRuntime.Trace axes.Label & vbCrLf
Next
```

Hinweis

Wenn Sie mithilfe des Auflistungsobjekts auf die Eigenschaften zugreifen, müssen Sie nicht den Namen der Auflistung angeben.

Bei der Auflistung "YAxis" schreiben Sie z. B. "objYAxis1.Label" statt "objYAxis1.YAxisLabel".

Siehe auch

YAxis-Objekt (Auflistung) (Seite 247)

1.14.5.4 Methoden H bis M

HideAlarm-Methode

Funktion

Führt die Tastenfunktion "Meldung ausblenden" des AlarmControl aus.

Syntax

```
Ausdruck.HideAlarm()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

InsertData-Methode

Funktion

Fügt Daten zur aufgerufenen Kurve hinzu.

Syntax

```
Ausdruck.InsertData(dblAxisX As Variant, dblAxisY As Variant)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Trend" zurückgibt.

Parameter

Parameter	Beschreibung
dblAxisX	Wert der X-Achse
dblAxisY	Wert der Y-Achse

Beispiel

```
'VBS300
Dim lngFactor
Dim dblAxisX
Dim dblAxisY
Dim objTrendControl
Dim objTrend
Set objTrendControl = ScreenItems("Control1")
Set objTrend = objTrendControl.GetTrend("Trend 1")
For lngFactor = -100 To 100
dblAxisX = CDbl(lngFactor * 0.02)
dblAxisY = CDbl(dblAxisX * dblAxisX + 2 * dblAxisX + 1)
objTrend.InsertData dblAxisX, dblAxisY
Next
```

Item-Methode

Funktion

Holt ein Objekt aus einer Collection und ermöglicht den Zugriff darauf über Index.

Beschreibung für DataItem-Objekt

Der Zugriff erfolgt über den Namen, unter dem der Wert in die Auflistung hinzugefügt wurde. Ein Einzelzugriff über Index ist nicht empfehlenswert, da sich der Index beim Hinzufügen und Löschen von Werten ändert.

Syntax

```
Ausdruck.Item()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Screens", "Layers" (oder "Tags") zurückgibt.

Hinweis

Bei "Tags" eingeschränkter Funktionsumfang! Es fehlen die Standard-Methoden get_Count und get_NewEnum, so dass kein Zugriff über Index und keine Aufzählung sämtlicher Variablen möglich ist.

Parameter

VARIANT

Beispiel

Das folgende Beispiel gibt die Namen aller Objekte des Bilds "NewPDL1" aus:

```
'VBS99
Dim objScreen
Dim objScrItem
Dim lngIndex
Dim lngAnswer
Dim strName
lngIndex = 1
Set objScreen = HMIRuntime.Screens("NewPDL1")
For lngIndex = 1 To objScreen.ScreenItems.Count
'
'The objects will be indicate by Item()
strName = objScreen.ScreenItems.Item(lngIndex).ObjectName
Set objScrItem = objScreen.ScreenItems(strName)
lngAnswer = MsgBox(objScrItem.ObjectName, vbOKCancel)
```

```
If vbCancel = lngAnswer Then Exit For
Next
```

Siehe auch

ScreenItems-Objekt (Auflistung) (Seite 138)

ScreenItem-Objekt (Seite 134)

Tags-Objekt (Auflistung) (Seite 149)

Alarms-Objekt (Auflistung) (Seite 120)

ProcessValues-Objekt (Auflistung) (Seite 133)

LockAlarm-Methode

Funktion

Führt die Tastenfunktion "Meldung sperren" des AlarmControl aus.

Syntax

```
Ausdruck.LockAlarm()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

LoopInAlarm-Methode

Funktion

Führt die Tastenfunktion "Loop in Alarm" des AlarmControl aus.

Syntax

```
Ausdruck.LoopInAlarm()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

MoveAxis-Methode

Funktion

Führt die Tastenfunktion "Achsenbereich verschieben" des OnlineTrendControl und FunctionTrendControl aus.

Syntax

```
Ausdruck.MoveAxis()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

MoveRuler

Funktion

Bewegt das Lineal von einem angegebenen Referenzpunkt um eine festgelegte Distanz.

Syntax

```
Ausdruck.MoveRuler( RulerIndex As Long, RulerMoveRef As Long,
MoveDistance As Long, Optional vTrendWindow As Variant )
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

Parameter	Beschreibung
RulerIndex	Gibt das Lineal an, welches bewegt wird: 0 = Lineal 1 = Lineal bei Beginn des Statistikbereichs 2 = Lineal bei Ende des Statistikbereichs
RulerMoveRef	Gibt den Referenzpunkt an, an dem sich der dritte Parameter "MoveDistance" orientiert: 0 = Beginnposition der Zeitachse 1 = Aktuelle Position des Lineals 2 = Endposition der Zeitachse
MoveDistance	Anzahl an Pixel, um die das Lineal vom angegebenen Referenzpunkt "RulerMoveRef" bewegt wird.
vTrendWindow	Optionaler Parameter bei der Verwendung mehrerer, unabhängiger Kurvenfenster. Legt das Kurvenfenster fest, in dem das Lineal bewegt wird. Wenn der Parameter nicht angegeben wird, bewegt sich das Lineal in allen Kurvenfenstern.

Rückgabewert

Die Funktion gibt die neue Position des Lineals zurück.

Beispiel

Tabelle 1-1 Lineal um 10 Pixel nach links bewegen

```
'VBS367
Sub OnClick(ByVal Item)
Dim ctrl
Set ctrl = ScreenItems.Item("Control1")
ctrl.MoveRuler (0, 1, -10)
End Sub
```

Im Beispiel wird das Lineal vom Referenzpunkt 1 (Aktuelle Position des Lineals) um -10 Pixel verschoben. Als Resultat steht das Lineal 10 Pixel links von seiner Ausgangsposition.

Beispiel

Tabelle 1-2 Lineal um 10 Pixel nach rechs bewegen

```
'VBS368
Sub OnClick(ByVal Item)
Dim ctrl
Set ctrl = ScreenItems.Item("Control1")
ctrl.MoveRuler (0, 1, 10)
End Sub
```

Im Beispiel wird das Lineal vom Referenzpunkt 1 (Aktuelle Position des Lineals) um 10 Pixel verschoben. Als Resultat steht das Lineal 10 Pixel rechts von seiner Ausgangsposition.

Beispiel

Tabelle 1-3 Lineal bei Bildaufschlag ans Ende stellen

```
'VBS369
Sub OnOpen()
Dim ctrl
Set ctrl = ScreenItems.Item("Control1")
ctrl.MoveRuler (0, 2, 0)
End Sub
```

Im Beispiel wird das Lineal vom Referenzpunkt 2 (Endposition der Zeitachse) um 0 Pixel bewegt. Als Resultat steht das Lineal an der Endposition der Zeitachse.

Beispiel

Tabelle 1-4 Die aktuelle Position des Lineals ermitteln

```
'VBS370
Sub OnClick(ByVal Item)
Dim ctrl
Set ctrl = ScreenItems.Item("Control1")
Dim pos
pos = ctrl.MoveRuler (0, 1, 0)
HmiRuntime.Trace "RulerPosition=" & pos & vbCrLf
End Sub
```

Im Beispiel wird das Lineal vom Referenzpunkt 1 (Aktuelle Position des Lineals) um 0 Pixel verschoben. Als Resultat steht das Lineal immer noch an seiner Ausgangsposition. Als Wert wird die Position des Lineals zurückgegeben.

MoveToFirst-Methode

Funktion

Führt die Tastenfunktion "Erste Zeile" des Control aus.

Syntax

```
Ausdruck.MoveToFirst()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

MoveToFirstLine-Methode

Funktion

Führt die Tastenfunktion "Erste Meldung" des AlarmControl aus.

Syntax

```
Ausdruck.MoveToFirstLine()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

MoveToFirstPage-Methode

Funktion

Führt die Tastenfunktion "Erste Seite" des AlarmControl aus.

Syntax

```
Ausdruck.MoveToFirstPage()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

MoveToLast-Methode

Funktion

Führt die Tastenfunktion "Letzter Datensatz" des Control aus.

Syntax

```
Ausdruck.MoveToLast()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

MoveToLastLine-Methode

Funktion

Führt die Tastenfunktion "Letzte Meldung" des AlarmControl aus.

Syntax

```
Ausdruck.MoveToLastLine()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

MoveToLastPage-Methode

Funktion

Führt die Tastenfunktion "Letzte Seite" des AlarmControl aus.

Syntax

```
Ausdruck.MoveToLastPage()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

MoveToNext-Methode

Funktion

Führt die Tastenfunktion "Nächster Datensatz" des Control aus.

Syntax

```
Ausdruck.MoveToNext()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

MoveToNextLine-Methode

Funktion

Führt die Tastenfunktion "Nächste Meldung" des AlarmControl aus.

Syntax

```
Ausdruck.MoveToNextLine()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

MoveToNextPage-Methode

Funktion

Führt die Tastenfunktion "Nächste Seite" des AlarmControl aus.

Syntax

```
Ausdruck.MoveToNextPage()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

MoveToPrevious-Methode

Funktion

Führt die Tastenfunktion "Vorhergehender Datensatz" des Control aus.

Syntax

```
Ausdruck.MoveToPrevious()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

MoveToPreviousLine-Methode

Funktion

Führt die Tastenfunktion "Vorhergehende Meldung" des AlarmControl aus.

Syntax

```
Ausdruck.MoveToPreviousLine()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

MoveToPreviousPage-Methode

Funktion

Führt die Tastenfunktion "Vorhergehende Seite" des AlarmControl aus.

Syntax

```
Ausdruck.MoveToPreviousPage()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

1.14.5.5 Methoden N bis R

NextColumn-Methode

Funktion

Führt die Tastenfunktion "Nächste Spalte" des OnlineTableControl aus.

Syntax

```
Ausdruck.NextColumn()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

NextTrend-Methode

Funktion

Führt die Tastenfunktion "Nächste Kurve" des OnlineTrendControl und FunctionTrendControl aus.

Syntax

```
Ausdruck.NextTrend()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

OneToOneView-Methode

Funktion

Führt die Tastenfunktion "Originalansicht" des OnlineTrendControl und FunctionTrendControl aus.

Syntax

```
Ausdruck.OneToOneView()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

PasteRows-Methode

Funktion

Führt die Tastenfunktion "Zeilen einfügen" des UserArchiveControl aus.

Syntax

```
Ausdruck.PasteRows()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

PreviousColumn-Methode

Funktion

Führt die Tastenfunktion "Vorhergehende Spalte" des OnlineTableControl aus.

Syntax

```
Ausdruck.PreviousColumn()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

PreviousTrend-Methode

Funktion

Führt die Tastenfunktion "Vorhergehende Kurve" des OnlineTrendControl und FunctionTrendControl aus.

Syntax

```
Ausdruck.PreviousTrend()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

Print-Methode

Funktion

Führt die Tastenfunktion "Drucken" des Control aus.

Syntax

```
Ausdruck.Print()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

QuitHorn-Methode

Funktion

Führt die Tastenfunktion "Quittierung zentraler Melder" des AlarmControl aus.

Syntax

```
Ausdruck.QuitHorn()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

QuitSelected-Methode

Funktion

Führt die Tastenfunktion "Einzelquittierung" des AlarmControl aus.

Syntax

```
Ausdruck.QuitSelected()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

QuitVisible-Methode

Funktion

Führt die Tastenfunktion "Sammelquittierung" des AlarmControl aus.

Syntax

```
Ausdruck.QuitVisible()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Read-Methode

Beschreibung für Tag-Objekt

Liest den Zustand einer Variablen (Tag-Objekt) kurz nach dem Aufrufzeitpunkt aus. Gleichzeitig wird das Tag-Objekt mit den gelesenen Werten versorgt. Beim Lesen der Variablen wird der Wert, der Quality Code und der Zeitstempel der Variablen ermittelt. Über die Eigenschaft "LastError" kann ermittelt werden, ob der Aufruf erfolgreich war.

Die Eigenschaften "Name", "Serverprefix" und "Tagprefix" werden dadurch nicht verändert.

Wird der Wert der Variablen erfolgreich gelesen, werden die Eigenschaften des Tag-Objektes mit folgenden Werten belegt:

Eigenschaft	Belegung
Value	Wert der Variablen
Name	Variablenname (unverändert)
QualityCode	Qualitätseinstufung
TimeStamp	aktueller Zeitstempel der Variablen
LastError	0
ErrorDescription	" "

Wird der Variablenwert nicht erfolgreich gelesen, werden die Eigenschaften des Tag-Objektes mit folgenden Werten belegt:

Eigenschaft	Belegung
Value	VT_Empty
Name	Variablenname (unverändert)
QualityCode	Bad Out of Service
TimeStamp	0

Eigenschaft	Belegung
LastError	Fehlercode der Leseoperation
ErrorDescription	Fehlerbeschreibung zu LastError

Hinweis

Eine Zusammenfassung der möglichen Quality Codes finden Sie im WinCC Information-System unter dem Stichwort "Kommunikation" > "Diagnose" oder "Kommunikation" > "Quality Codes".

Syntax

```
Ausdruck.Read([Readmode])
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Tag-Objekt zurückgibt. Der Rückgabewert der Read-Methode ist der Wert der ausgelesenen Variablen.

Parameter

Durch den optionalen Parameter "Readmode" wird zwischen zwei Lesearten unterschieden:

Parameter	Beschreibung
0	Der Wert der Variablen wird aus dem Prozessabbild gelesen (cache). 0 ist der Defaultwert.
1	Der Wert der Variablen wird direkt aus dem AS oder dem Kanal gelesen (direct).

Wird der Parameter "Readmode" weggelassen, wird standardmäßig aus dem Prozessabbild gelesen. Der Rückgabewert der Read-Methode ist der ausgelesene Variablenwert als VARIANT.

Lesen aus dem Prozessabbild

Beim Lesen aus dem Prozessabbild wird die Variable angemeldet und von da an zyklisch aus dem Automatisierungssystem angefordert. Der Zyklus der Anmeldung ist dabei vom projektierten Trigger abhängig. Der Wert wird aus dem Variablenabbild von WinCC gelesen. Bei Bildabwahl werden die Variablen wieder abgemeldet. Der Aufruf zeichnet sich durch Folgendes aus:

- Der Wert wird aus dem Variablenabbild von WinCC gelesen
- Der Aufruf ist im Vergleich zum direkten Lesen schneller (Außer beim ersten Aufruf: Der erste Aufruf dauert prinzipiell länger, da der Wert aus dem AS gelesen und angemeldet werden muss.)
- Die Dauer des Aufrufes ist nicht von Buslast oder vom AS abhängig

Verhalten in Aktionen mit Variablentrigger

Alle im Variablentrigger enthaltenen Variablen sind bereits bei Bildanwahl bekannt und werden mit der angegebenen Überwachungszeit angemeldet. Da alle Variablen auf einmal angefordert werden, kann vom Kanal die bestmögliche Optimierung erzielt werden. Wird innerhalb einer Aktion eine Variable mit Read angefordert, die im Trigger enthalten ist, liegt der Wert bereits vor und wird dem Aufruf direkt übergeben. Wird eine Variable angefordert, die nicht im Trigger enthalten ist, ist das Verhalten wie beim Standardtrigger.

Verhalten in Aktionen mit zyklischem Trigger

Beim ersten Aufruf wird die Variable mit der halben Zykluszeit angemeldet. Bei jedem weiteren Aufruf liegt dann der Wert vor.

Verhalten in ereignisgetriggerten Aktionen

Beim ersten Aufruf wird die Variable im Modus "bei Änderung" angemeldet. Prozessvariablen, die im Modus "bei Änderung" angemeldet sind, entsprechen einem zyklischen Leseauftrag mit einer Zykluszeit von 1s.

Wenn mit einem Ereignis (z.B. Mausklick) ein Wert asynchron angefordert wird, wird die Variable in das Variablenabbild aufgenommen. Die Variable wird ab diesem Zeitpunkt zyklisch aus dem AS angefordert und erhöht somit die Grundlast. Um diese Erhöhung der Grundlast zu umgehen, können Sie den Wert auch synchron lesen. Der synchrone Aufruf verursacht zwar einmalig eine höhere Kommunikationslast, aber die Variable wird nicht in das Variablenabbild aufgenommen.

Direktes Lesen

Beim direkten Lesen wird der aktuelle Wert zurückgeliefert. Die Variable wird nicht zyklisch angemeldet, sondern der Wert wird einmalig aus dem AS angefordert. Direktes Lesen hat folgende Eigenschaften:

- Der Wert wird explizit aus dem AS gelesen
- Der Aufruf dauert im Vergleich zum Lesen aus dem Prozessabbild länger
- Die Dauer des Aufrufes ist u.a. von der Buslast und vom AS abhängig

Beispiel

Lesen einer Variablen direkt aus dem AS oder dem Kanal

```
'VBS100
Dim objTag
Dim vntValue
Set objTag = HMIRuntime.Tags("Tagname")
vntValue = objTag.Read(1)    'Read direct
MsgBox vntValue
```

Lesen einer Variablen aus dem Prozessabbild

```
'VBS101
Dim objTag
Dim vntValue
Set objTag = HMIRuntime.Tags("Tagname")
vntValue = objTag.Read    'Read from cache
MsgBox vntValue
```

Beschreibung für TagSet-Objekt

Das TagSet-Objekt bietet die Möglichkeit, mehrere Variablen mit einem Aufruf zu lesen.

Die Funktionsweise ist dabei weitgehend mit der eines Tag-Objektes identisch. Nachfolgend werden nur die Abweichungen beschrieben.

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "TagSet" zurückgibt.

Lesen aus dem Prozessabbild

Das TagSet-Objekt bietet den Vorteil, dass es mit einem Leseauftrag mehrere Variablen anfordern kann. Dabei werden die Variablen gesammelt im Prozessabbild angemeldet und somit die Performance verbessert.

Direktes Lesen

Da mit einem Aufruf mehrere Leseaufträge abgearbeitet werden können, wird die Performance gegenüber mehreren Einzelaufrufen verbessert.

Beispiel

Das folgende Beispiel zeigt wie Variablen in die TagSet-Auflistung aufgenommen und danach gelesen werden.

```
'VBS174
Dim group
Set group = HMIRuntime.Tags.CreateTagSet
group.Add "Motor1"
group.Add "Motor2"
group.Read
HMIRuntime.Trace "Motor1: " & group("Motor1").Value & vbNewLine
HMIRuntime.Trace "Motor2: " & group("Motor2").Value & vbNewLine
```

Setzt man den optionalen Parameter "Readmode" auf 1, so werden Prozess-Variablen nicht angemeldet sondern direkt aus dem AS oder dem Kanal gelesen.

```
group.Read 1
```

Siehe auch

Beispiel: So lesen Sie Variablenwerte (Seite 803)

Beispiel: So schreiben Sie Variablenwerte (Seite 801)

LastError-Eigenschaft (Seite 435)

ErrorDescription-Eigenschaft (Seite 389)

TagSet-Objekt (Auflistung) (Seite 151)

Tag-Objekt (Seite 146)

Read Tags-Methode

Funktion

Führt die Tastenfunktion "Variablen lesen" des UserArchiveControl aus.

Syntax

```
Ausdruck.ReadTags()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Refresh-Methode

Funktion

Zeichnet alle sichtbaren Bilder neu.

Syntax

```
Ausdruck.Refresh
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Screens" oder "Screen" zurückgibt.

Parameter

--

Beispiele

Das erste Beispiel erzwingt ein sofortiges Neuzeichnen aller sichtbaren Bilder:

```
'VBS149
HMIRuntime.Screens.Refresh
```

Das zweite Beispiel erzwingt ein sofortiges Neuzeichnen des Grundbildes:

```
'VBS150
HMIRuntime.Screens(1).Refresh
```

Siehe auch

Screen-Objekt (Seite 140)

Screens-Objekt (Auflistung) (Seite 143)

HMIRuntime-Objekt (Seite 127)

Remove-Methode

Beschreibung für TagSet-Objekt

Entfernt eine Variable aus einer TagSet-Auflistung. Die Variable kann über den Namen oder über eine Referenz auf ein Tag-Objekt entfernt werden.

Syntax

```
Ausdruck.Remove [Tag]
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "TagSet" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
Tag	Name einer WinCC Variable oder Referenz auf ein Tag-Objekt, das aus der Auflistung entfernt werden soll.

Beispiel

Das folgende Beispiel zeigt, wie mehrere Variablen in die TagSet-Auflistung aufgenommen werden und eine Variable wieder entfernt wird.

```
'VBS175
Dim group
Set group = HMIRuntime.Tags.CreateTagSet
group.Add "Motor1"
group.Add "Motor2"
group.Remove "Motor1"
```

Beschreibung für DataSet-Objekt

Löscht das im Parameter "Name" angegebene Element aus einer Auflistung.

Syntax

```
Ausdruck.Remove [Name]
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "DataSet" zurückgibt.

Parameter

VARIANT

Parameter	Beschreibung
Name	Name des Objekts, das aus der Auflistung entfernt werden soll.

Beispiel

Das Beispiel zeigt, wie das Objekt "motor1" aus der Auflistung entfernt wird.

```
'VBS166
HMIRuntime.DataSet.Remove("motor1")
```

Beschreibung für Objekte Logging, AlarmLogs, DataLogs

Die Methode löscht zuvor eingelagerte Archivsegmente aus dem Runtime-Projekt.

Mit der Methode "Remove" gelöschte Archivsegmente werden aus dem Common Archiving-Verzeichnis des Projektes entfernt.

Der Aufruf kann in Abhängigkeit von den Archivdaten einen längeren Zeitraum in Anspruch nehmen. Dies kann die Abarbeitung der nachfolgenden Skripte blockieren. Eine Blockade der Aktionen im Bild können Sie vermeiden, indem Sie den Aufruf in einer Aktion im Global Scripting starten, zum Beispiel durch das Starten der Aktion durch eine Triggervariable.

Durch das Trennen / Löschen der Archive wird CPU-Last erzeugt. Dies wirkt sich auf die Performance aus.

Hinweis

Der Aufruf der Methode "Remove" ist derzeit nur am Server möglich. Es existiert aber ein Beispiel, das zeigt, wie die Methode vom Client aus auf dem Server gestartet werden kann.

Bei Redundanz gilt: Wieder eingelagerte Archive werden mit der Methode "Remove" nur auf dem Rechner gelöscht, an dem auch die Methode aufgerufen wurde.

Syntax

Objekte Logging, AlarmLogs

```
Ausdruck.Remove [TimeFrom] [TimeTo] [TimeOut] [ServerPrefix]
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Logging" oder "AlarmLogs" zurückgibt.

Objekt DataLogs

```
Ausdruck.Remove [TimeFrom] [TimeTo] [TimeOut] [Type] [ServerPrefix]
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "DataLogs" zurückgibt.

Parameter

TimeFrom

Zeitpunkt, von dem an die Archive gelöscht werden sollen.
Bei der Angabe des Zeitformats ist auch eine Kurzform möglich, wie im Absatz "Zeitformat" beschrieben.

TimeTo

Zeitpunkt, bis zu dem die Archivsegmente gelöscht werden sollen.
Bei der Angabe des Zeitformats ist auch eine Kurzform möglich, wie im Absatz "Zeitformat" beschrieben.

Timeout

Timeout in Millisekunden.
Wenn Sie als Wert "-1" eingeben, wird für immer gewartet (infinite). Wenn Sie den Wert "0" eingeben, wird nicht gewartet.

Type

Typ des Archivs.
Dieser Parameter kann (optional) nur für das Löschen von Archivsegmenten des Tag Logging verwendet werden.
Folgende Werte können eingegeben werden:

zugeordneter Wert	Typ	Beschreibung
1	hmiDataLogFast	Tag Logging Fast data
2	hmiDataLogSlow	Tag Logging Slow data
3	hmiDataLogAll	Tag Logging Fast and Slow data

ServerPrefix

Reserviert für zukünftige Versionen.

Rückgabewert

Wenn beim Löschen der Archivsegmente ein Fehler aufgetreten ist, gibt die Methode eine Fehlermeldung zurück. Weitere Informationen finden Sie unter dem Thema "Fehlermeldungen aus dem Bereich Datenbanken".

Zeitformat

Das Zeitformat ist wie folgt definiert: YYYY-MM-DD hh:mm:ss, wobei YYYY das Jahr bezeichnet, MM den Monat, DD den Tag, hh die Stunde, mm die Minute und ss die Sekunde. Beispielsweise wird die Uhrzeit 2 Minuten und eine Sekunde nach 11 Uhr am 26. Juli 2004 so dargestellt: 2004-07-26 11:02:01.

Bei den Parametern "TimeFrom" und "TimeTo" ist die Angabe von Datum und Uhrzeit in einer Kurzform möglich. Dabei sind nicht alle Felder des Formats zu belegen. Die Kurzform bedeutet, dass bei der Datum/Zeit-Angabe ein oder mehrere Parameter, mit dem Sekunden-Wert beginnend, entfallen können. So kann die Angabe z.B. in den Formaten "YYYY-MM" oder "YYYY-MM-DD hh" erfolgen. Mit der Angabe "TimeFrom" = "2004-09" und "TimeTo" =

"2004-10-04" werden alle Archivsegmente von September 2004 bis einschließlich 4. Oktober eingelagert.

Beispiel

Im folgenden Beispiel werden nachträglich (wieder) eingelagerte Archivsegmente eines bestimmten Zeitraumes entfernt und der Rückgabewert als Trace ausgegeben.

```
'VBS182
HMIRuntime.Trace "Ret: " & HMIRuntime.Logging.Remove("2004-08-22","2004-09-22",-1) &
vbNewLine
```

Im folgenden Beispiel werden alle nachträglich (wieder) eingelagerten Archivsegmente entfernt und der Rückgabewert als Trace ausgegeben.

```
'VBS183
HMIRuntime.Trace "Ret: " & HMIRuntime.Logging.Remove("","",-1) & vbNewLine
```

Siehe auch

Fehlermeldungen aus dem Bereich Datenbanken (Seite 794)

Beispiel: So starten Sie eine Aktion am Server (Logging-Objekt) (Seite 809)

Logging-Objekt (Seite 131)

DataSet-Objekt (Auflistung) (Seite 125)

DataLogs-Objekt (Seite 124)

AlarmLogs-Objekt (Seite 121)

TagSet-Objekt (Auflistung) (Seite 151)

RemoveAll-Methode

Beschreibung für TagSet-Objekt

Löscht alle Variablen aus einer TagSet-Auflistung.

Syntax

```
Ausdruck.RemoveAll
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "TagSet" zurückgibt.

Parameter

--

Beispiel

Das folgende Beispiel zeigt, wie mehrere Variablen in die TagSet-Auflistung aufgenommen und alle wieder entfernt werden.

```
'VBS176
Dim group
Set group = HMIRuntime.Tags.CreateTagSet
group.Add "Motor1"
group.Add "Motor2"
group.RemoveAll
```

Beschreibung für DataSet-Objekt

Löscht alle Werte oder Objektreferenzen aus einer DataSet-Auflistung.

Syntax

```
Ausdruck.RemoveAll
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "DataSet" zurückgibt.

Parameter

--

Beispiel

Das Beispiel zeigt, wie alle Objekte aus der Auflistung entfernt werden.

```
'VBS167
HMIRuntime.DataSet.RemoveAll
```

Siehe auch

DataSet-Objekt (Auflistung) (Seite 125)

TagSet-Objekt (Auflistung) (Seite 151)

Tag-Objekt (Seite 146)

RemoveData-Methode

Funktion

Löscht die Daten der aufgerufenen Kurve.

Syntax

```
Ausdruck.RemoveData
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Trend" zurückgibt.

Beispiel

```
'VBS310
Dim objTrendControl
Dim objTrend
Set objTrendControl = ScreenItems("Control1")
Set objTrend = objTrendControl.GetTrend("Trend 1")
objTrend.RemoveData
```

Restore-Methode

Beschreibung für Objekte Logging, AlarmLogs, DataLogs

Die Methode fügt ausgelagerte Archivsegmente dem Runtime-Projekt hinzu.

Beim Einlagern der Archivsegmente werden diese in das Common Archiving-Verzeichnis des Projektes kopiert. Daher muss entsprechend Speicherkapazität vorhanden sein.

Der Aufruf kann in Abhängigkeit von den Archivdaten einen längeren Zeitraum in Anspruch nehmen. Dies kann die Abarbeitung der nachfolgenden Skripte blockieren. Eine Blockade der Aktionen im Bild können Sie vermeiden, indem Sie den Aufruf in einer Aktion im Global Scripting starten, zum Beispiel durch das Starten der Aktion durch eine Triggervariable.

Durch das Verbinden / Kopieren der Archive entsteht CPU-Last weil insbesondere bei eingeschalteter Signaturprüfung der SQL-Server mehr belastet wird. Durch das Kopieren der Archivsegmente wird der Festplattenzugriff verlangsamt.

Bei eingeschalteter Signaturprüfung wird eine Fehlermeldung zurückgegeben, wenn ein nicht signiertes oder verändertes Archiv eingelagert werden soll. Es wird immer nur eine Fehlermeldung zurückgegeben, auch beim Auftreten mehrerer Fehler während eines Einlagerungsvorgangs. Zusätzlich wird für jedes Archivsegment eine WinCC Systemmeldung erzeugt. In der Windows Ereignisanzeige wird unter "Anwendung" ein Eintrag hinzugefügt. Dadurch besteht die Möglichkeit zu überprüfen, welche Archivsegmente den Fehler erzeugen.

- Bei einem nicht signierten Archiv wird der Rückgabewert "0x8004720F" zurückgegeben. Das Archiv wird eingelagert. In der Ereignisanzeige erfolgt folgender Eintrag: "Validation of database <db_name> failed! No signature found!"
- Bei einem veränderten Archiv wird der Rückgabewert "0x80047207" zurückgegeben. In der Ereignisanzeige erfolgt der Eintrag "Validation of database <db_name> failed !". Das Archiv wird nicht eingelagert.

Hinweis

Der Aufruf der Methode "Restore" ist derzeit nur am Server möglich. Es existiert aber ein Beispiel, das zeigt, wie die Methode vom Client aus auf dem Server gestartet werden kann.

Bei Redundanz gilt: Bei Wiedereinlagerung von Archiven mit der Methode "Restore" werden die Archivsegmente nur dem Runtime-Projekt auf dem Rechner hinzugefügt, an dem auch die Methode aufgerufen wurde.

Syntax

Objekte Logging, AlarmLogs

```
Ausdruck.Restore [SourcePath] [TimeFrom] [TimeTo] [TimeOut]
[ServerPrefix]
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Logging" oder "AlarmLogs" zurückgibt.

Objekt DataLogs

```
Ausdruck.Restore [SourcePath] [TimeFrom] [TimeTo] [TimeOut] [Type]
[ServerPrefix]
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "DataLogs" zurückgibt.

Parameter

SourcePath

Pfad zu den Archivdaten.

TimeFrom

Zeitpunkt, von dem an die Archive eingelagert werden sollen.
Bei der Angabe des Zeitformats ist auch eine Kurzform möglich, wie im Absatz "Zeitformat" beschrieben.

TimeTo

Zeitpunkt, bis zu dem die Archivsegmente eingelagert werden sollen.
Bei der Angabe des Zeitformats ist auch eine Kurzform möglich, wie im Absatz "Zeitformat" beschrieben.

Timeout

Timeout in Millisekunden.
Wenn Sie als Wert "-1" eingeben, wird für immer gewartet (infinite). Wenn Sie den Wert "0" eingeben, wird nicht gewartet.

Type

Typ des Archivs.
Dieser Parameter kann (optional) nur für die Einlagerung von Archivsegmenten des Tag Logging verwendet werden.
Folgende Werte können eingegeben werden:

zugeordneter Wert	Typ	Beschreibung
1	hmiDataLogFast	Tag Logging Fast data
2	hmiDataLogSlow	Tag Logging Slow data
3	hmiDataLogAll	Tag Logging Fast and Slow data

ServerPrefix

Reserviert für zukünftige Versionen.

Rückgabewert

Wenn beim Einlagern der Archivsegmente ein Fehler aufgetreten ist, gibt die Methode eine Fehlermeldung zurück. Weitere Informationen finden Sie unter dem Thema "Fehlermeldungen aus dem Bereich Datenbanken".

Zeitformat

Das Zeitformat ist wie folgt definiert: YYYY-MM-DD hh:mm:ss, wobei YYYY das Jahr bezeichnet, MM den Monat, DD den Tag, hh die Stunde, mm die Minute und ss die Sekunde. Beispielsweise wird die Uhrzeit 2 Minuten und eine Sekunde nach 11 Uhr am 26. Juli 2004 so dargestellt: 2004-07-26 11:02:01.

Bei den Parametern "TimeFrom" und "TimeTo" ist die Angabe von Datum und Uhrzeit in einer Kurzform möglich. Dabei sind nicht alle Felder des Formats zu belegen. Die Kurzform bedeutet, dass bei der Datum/Zeit-Angabe ein oder mehrere Parameter, mit dem Sekunden-Wert beginnend, entfallen können. So kann die Angabe z.B. in den Formaten "YYYY-MM" oder "YYYY-MM-DD hh" erfolgen. Mit der Angabe "TimeFrom" = "2004-09" und "TimeTo" = "2004-10-04" werden alle Archivsegmente von September 2004 bis einschließlich 4. Oktober eingelagert.

Beispiel

Im folgenden Beispiel werden alle Archivsegmente ab dem angegebenen Zeitraum wieder eingelagert und der Rückgabewert als Trace ausgegeben.

```
'VBS184
HMIRuntime.Trace "Ret: " & HMIRuntime.Logging.Restore("D:\Folder","2004-09-14","",-1) &
vbNewLine
```

Im folgenden Beispiel werden Tag Logging Slow Archivsegmente des angegebenen Zeitraums wieder eingelagert und der Rückgabewert als Trace ausgegeben.

```
'VBS185
HMIRuntime.Trace "Ret: " & HMIRuntime.Logging.DataLogs.Restore("D:\Folder","2004-09-14
12:30:05","2004-09-20 18:30",-1,2) & vbNewLine
```

Im folgenden Beispiel werden Alarm Logging Archivsegmente bis zu dem angegebenen Zeitraum wieder eingelagert und der Rückgabewert als Trace ausgegeben.

```
'VBS186
HMIRuntime.Trace "Ret: " & HMIRuntime.Logging.AlarmLogs.Restore("","2004-09-20",-1) &
vbNewLine
```

Siehe auch

Fehlermeldungen aus dem Bereich Datenbanken (Seite 794)

Beispiel: So starten Sie eine Aktion am Server (Logging-Objekt) (Seite 809)

Logging-Objekt (Seite 131)

DataLogs-Objekt (Seite 124)

AlarmLogs-Objekt (Seite 121)

1.14.5.6 Methoden S bis T

SelectAll

Funktion

Markiert alle Zeilen im Control, das auf eine Tabelle basiert.

Syntax

```
Ausdruck.SelectAll()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Siehe auch

Row-Objekt (Auflistung) (Seite 235)

SelectRow

Funktion

Markiert eine bestimmte Zeile im Control, das auf eine Tabelle basiert.

Syntax

```
Ausdruck.SelectRow(ByVal IRow As Long, Optional bExtendSelection As
Boolean)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

Parameter	Beschreibung
IRow	Nummer der Zeile, die zu markieren ist.
bExtendSelection	Gibt optional an, ob die aktuelle Markierung erweitert wird. Ist nur relevant, wenn die Mehrfachselektion möglich ist.

Beispiel

- Aktuell ist Zeile 1 markiert. Wenn SelectRow(2, True) aufgerufen wird, sind danach Zeile 1 und Zeile 2 markiert.
- Aktuell ist Zeile 1 markiert. Wenn SelectRow(2, False) oder SelectRow(2) ohne optionalen Parameter aufgerufen wird, ist danach nur Zeile 2 markiert.

Siehe auch

Row-Objekt (Auflistung) (Seite 235)

SelectedStatisticArea-Methode

Funktion

Führt die Tastenfunktion "Statistikbereich festlegen" des OnlineTableControl aus.

Syntax

```
Ausdruck.SelectedStatisticArea()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ServerExport-Methode

Funktion

Führt die Tastenfunktion "Archiv exportieren" des UserArchiveControl aus.

Syntax

```
Ausdruck.ServerExport()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ServerImport-Methode

Funktion

Führt die Tastenfunktion "Archiv importieren" des UserArchiveControl aus.

Syntax

```
Ausdruck.ServerImport()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ShowColumnSelection-Methode

Funktion

Führt die Tastenfunktion "Spalten wählen" des OnlineTableControl aus.

Syntax

```
Ausdruck.ShowColumnSelection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ShowComment-Methode

Funktion

Führt die Tastenfunktion "Kommentar-Dialog" des AlarmControl aus.

Syntax

```
Ausdruck.ShowComment()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ShowDisplayOptionsDialog-Methode

Funktion

Führt die Tastenfunktion "Anzeigeoptions-Dialog" des AlarmControl aus.

Syntax

```
Ausdruck.ShowDisplayOptionsDialog()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

ShowEmergencyQuitDialog-Methode

Funktion

Führt die Tastenfunktion "Not-Quittierung" des AlarmControl aus.

Syntax

```
Ausdruck.ShowEmergencyQuitDialog()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

ShowHelp-Methode

Funktion

Führt die Tastenfunktion "Hilfe" des Control aus.

Syntax

```
Ausdruck.ShowHelp()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

ShowHideList-Methode

Funktion

Führt die Tastenfunktion "Liste auszublendender Meldungen" des AlarmControl aus.

Syntax

```
Ausdruck.ShowHideList()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ShowHitList-Methode

Funktion

Führt die Tastenfunktion "Hitliste" des AlarmControl aus.

Syntax

```
Ausdruck.ShowHitList()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ShowInfoText-Methode

Funktion

Führt die Tastenfunktion "Infotext-Dialog" des AlarmControl aus.

Syntax

```
Ausdruck.ShowInfoText()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

ShowLockDialog-Methode

Funktion

Führt die Tastenfunktion "Sperr-Dialog" des AlarmControl aus.

Syntax

```
Ausdruck.ShowLockDialog()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

ShowLockList-Methode

Funktion

Führt die Tastenfunktion "Sperrliste" des AlarmControl aus.

Syntax

```
Ausdruck.ShowLockList()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ShowLongTermArchiveList-Methode

Funktion

Führt die Tastenfunktion "Langzeitarchivliste" des AlarmControl aus.

Syntax

```
Ausdruck.ShowLongTermArchiveList()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ShowMessageList-Methode

Funktion

Führt die Tastenfunktion "Meldeliste" des AlarmControl aus.

Syntax

```
Ausdruck.ShowMessageList()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ShowPercentageAxis-Methode

Funktion

Führt die Tastenfunktion "Relative Achse" des OnlineTrendControl aus.

Syntax

```
Ausdruck.ShowPercentageAxis()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

ShowPropertyDialog-Methode

Funktion

Führt die Tastenfunktion "Konfigurationsdialog" des Control aus.

Syntax

```
Ausdruck.ShowPropertyDialog()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

VARIANT

ShowSelectArchive-Methode

Funktion

Führt die Tastenfunktion "Datenanbindung wählen" des UserArchiveControl aus.

Syntax

```
Ausdruck.ShowSelectArchive()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ShowSelection-Methode

Funktion

Führt die Tastenfunktion "Selektions-Dialog" des UserArchiveControl aus.

Syntax

```
Ausdruck.ShowSelection ()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ShowSelectTimeBase-Methode

Funktion

Führt die Tastenfunktion "Zeitbasis-Dialog" des UserArchiveControl aus.

Syntax

```
Ausdruck.ShowSelectTimeBase()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ShowSelectionDialog-Methode

Funktion

Führt die Tastenfunktion "Selektions-Dialog" des AlarmControl aus.

Syntax

```
Ausdruck.ShowSelectionDialog()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

ShowShortTermArchiveList-Methode

Funktion

Führt die Tastenfunktion "Kurzzeitarchivliste" des AlarmControl aus.

Syntax

```
Ausdruck.ShowShortTermArchiveList()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

ShowSort-Methode

Funktion

Führt die Tastenfunktion "Sortier-Dialog" des UserArchiveControl aus.

Syntax

```
Ausdruck.ShowSort()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ShowSortDialog-Methode

Funktion

Führt die Tastenfunktion "Sortier-Dialog" des AlarmControl aus.

Syntax

```
Ausdruck.ShowSortDialog()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ShowTagSelection-Methode

Funktion

Führt die Tastenfunktion "Datenanbindung wählen" des Control aus.

Syntax

```
Ausdruck.ShowTagSelection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ShowTimebaseDialog-Methode

Funktion

Führt die Tastenfunktion "Zeitbasis-Dialog" des AlarmControl aus.

Syntax

```
Ausdruck.ShowTimebaseDialog()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

ShowTimeSelection-Methode

Funktion

Führt die Tastenfunktion "Zeitbereich wählen" des Control aus.

Syntax

```
Ausdruck.ShowTimeSelection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

ShowTrendSelection-Methode

Funktion

Führt die Tastenfunktion "Kurven wählen" des OnlineTrendControl und FunctionTrendControl aus.

Syntax

```
Ausdruck.ShowTrendSelection()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

-- -

StartStopUpdate-Methode

Funktion

Führt die Tastenfunktion "Start" oder "Stopp" des Control aus.

Syntax

```
Ausdruck.StartStopUpdate()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Stop-Methode

Funktion

Beendet WinCC Runtime.

Syntax

```
HMIRuntime.Stop
```

Parameter

Beispiel

Das folgende Beispiel beendet WinCC Runtime:

```
'VBS124
HMIRuntime.Stop
```

Siehe auch

HMIRuntime-Objekt (Seite 127)

Trace-Methode

Beschreibung

Gibt Meldungen im Diagnosefenster aus.

Syntax

```
HMIRuntime.Trace
```

Parameter

STRING

Beispiel

Das folgende Beispiel schreibt einen Text ins Diagnosefenster:

```
'VBS103
HMIRuntime.Trace "Customized error message"
```

Siehe auch

HMIRuntime-Objekt (Seite 127)

1.14.5.7 Methoden U bis Z

UnhideAlarm-Methode

Funktion

Führt die Tastenfunktion "Meldung einblenden" des AlarmControl aus.

Syntax

```
Ausdruck.UnhideAlarm()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

UnlockAlarm-Methode

Funktion

Führt die Tastenfunktion "Meldung freigeben" des AlarmControl aus.

Syntax

```
Ausdruck.UnlockAlarm()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

UnselectAll

Funktion

Entfernt die Markierung an allen Zeilen im Control, das auf eine Tabelle basiert.

Syntax

```
Ausdruck.UnselectAll()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

Siehe auch

Row-Objekt (Auflistung) (Seite 235)

UnselectRow

Funktion

Entfernt die Markierung einer bestimmten Zeile im Control, das auf eine Tabelle basiert.

Syntax

```
Ausdruck.UnselectRow(ByVal IRow As Long)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

Long

Parameter	Beschreibung
IRow	Nummer der Zeile, die zu markieren ist.

Siehe auch

Row-Objekt (Auflistung) (Seite 235)

Write-Methode

Beschreibung für Tag-Objekt

Schreibt einen Wert synchron oder asynchron in eine Variable. Über die Eigenschaft "LastError" kann festgestellt werden, ob der Aufruf erfolgreich war.

Wird der Wert der Variablen erfolgreich gesetzt, werden die Eigenschaften des Tag-Objektes mit folgenden Werten belegt:

Eigenschaft	Belegung
Value	Vom Benutzer gesetzter Wert der Variablen (unverändert)
Name	Variablenname (unverändert)
QualityCode	Bad Out of Service
TimeStamp	0
LastError	0
ErrorDescription	" "

Wird der Variablenwert nicht erfolgreich gesetzt, werden die Eigenschaften des Tag-Objektes mit folgenden Werten belegt:

Eigenschaft	Belegung
Value	Vom Benutzer gesetzter Wert der Variablen (unverändert)
Name	Variablenname (unverändert)
QualityCode	Bad Out of Service
TimeStamp	0
LastError	Fehlercode der Schreiboperation
ErrorDescription	Fehlerbeschreibung zu LastError

Syntax

```
Ausdruck.Write [Value],[Writemode]
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Tags-Objekt zurückgibt.

Parameter

Der zu schreibende Wert kann der Methode als Parameter direkt übergeben werden. Wird der Parameter nicht angegeben, wird der Wert der Eigenschaft "Value" verwendet. Mit dem optionalen Parameter "Writemode" können Sie wählen, ob der Variablenwert synchron oder asynchron geschrieben werden soll. Wird der Parameter "Writemode" nicht verwendet, wird als Defaultwert asynchron geschrieben.

Bei einem Schreibvorgang werden keine Informationen zum Zustand der Variablen geliefert.

In der Eigenschaft "Value" steht der Wert, der vor oder bei der Schreiboperation gesetzt wurde, er muss also nicht dem tatsächlichen aktuellen Wert der Variable entsprechen. Wenn Sie die Informationen zu der Variable aktualisieren möchten, verwenden Sie die Read-Methode.

Parameter	Beschreibung
Value (optional)	Wert der Variablen wird angegeben. Der angegebene Wert überschreibt den Wert der Eigenschaft "Value" im Tag-Objekt. Wert der Variablen wird nicht angegeben. Die Variable erhält den aktuellen Wert aus der Eigenschaft "Value" des Tag-Objektes.
Writemode (optional)	0 oder leer: Der Variablenwert wird asynchron geschrieben. 0 ist der Defaultwert. 1: Der Variablenwert wird synchron geschrieben.

Beim asynchronen Schreiben wird sofort in das Variablenabbild geschrieben. Der Anwender erhält keine Rückmeldung darüber, ob der Wert auch in das AS geschrieben wurde.

Beim synchronen Schreiben (direkt in das AS) wird erst geschrieben, wenn das Automatisierungsgerät bereit ist. Der Anwender erhält eine Rückmeldung, wenn das Schreiben nicht erfolgreich war.

Beispiel

Asynchrones Schreiben

```
'VBS104
Dim objTag
Set objTag = HMIRuntime.Tags("Var1")
objTag.Value = 5
objTag.Write
MsgBox objTag.Value
```

oder

```
'VBS105
Dim objTag
Set objTag = HMIRuntime.Tags("Var1")
objTag.Write 5
MsgBox objTag.Value
```

Synchrones Schreiben

```
'VBS106
Dim objTag
Set objTag = HMIRuntime.Tags("Var1")
objTag.Value = 5
objTag.Write ,1
MsgBox objTag.Value
```

oder

```
'VBS107
Dim objTag
Set objTag = HMIRuntime.Tags("Var1")
objTag.Write 5, 1
MsgBox objTag.Value
```

Beschreibung für TagSet-Objekt

Das TagSet-Objekt bietet die Möglichkeit, mehrere Variablen mit einem Aufruf zu schreiben.

Die Funktionsweise ist dabei weitgehend mit der eines Tag-Objektes identisch. Nachfolgend werden nur die Abweichungen beschrieben.

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "TagSet" zurückgibt.

Parameter

Um unterschiedliche Werte zu schreiben muss die Eigenschaft "Value" der einzelnen Tag-Objekte gesetzt und danach Write ohne Parameter "Value" aufgerufen werden. Da die Schreibaufträge zu einem Aufruf zusammengefasst werden ergibt sich eine verbesserte Performance gegenüber den Einzelaufrufen.

Beim TagSet-Objekt ist es nicht möglich mit der "Write"-Methode einen Wert mitzugeben. Die einzelnen Werte müssen über die Eigenschaft "Value" der einzelnen Tag-Objekte gesetzt werden.

Beispiel

Das folgende Beispiel zeigt wie man Variablen in die TagSet Collection aufnimmt, die Variablenwerte setzt und danach schreibt.

```
'VBS173
Dim group
Set group = HMIRuntime.Tags.CreateTagSet
group.Add "Wert1"
group.Add "Wert2"
group("Wert1").Value = 3
group("Wert2").Value = 9
group.Write
```

Setzt man den optionalen Parameter "Writemode" auf 1, so werden Prozess-Variablen synchron (direkt in das AS) geschrieben.

```
group.Write 1
```

Siehe auch

LastError-Eigenschaft (Seite 435)

ErrorDescription-Eigenschaft (Seite 389)

TagSet-Objekt (Auflistung) (Seite 151)

Tag-Objekt (Seite 146)

WriteTags-Methode

Funktion

Führt die Tastenfunktion "Variablen schreiben" des UserArchiveControl aus.

Syntax

```
Ausdruck.WriteTags()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

ZoomArea-Methode

Funktion

Führt die Tastenfunktion "Zoomen Ausschnitt" des OnlineTrendControl und FunctionTrendControl aus.

Syntax

```
Ausdruck.ZoomArea()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

ZoomInOut-Methode

Funktion

Führt die Tastenfunktion "Zoomen +/-" des OnlineTrendControl und FunctionTrendControl aus.

Syntax

```
Ausdruck.ZoomInOut()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ZoomInOutTime-Methode

Funktion

Führt die Tastenfunktion "Zeitachse zoomen +/-" des OnlineTrendControl aus.

Syntax

```
Ausdruck.ZoomInOutTime()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ZoomInOutValues-Methode

Funktion

Führt die Tastenfunktion "Wertachse zoomen +/-" des OnlineTrendControl aus.

Syntax

```
Ausdruck.ZoomInOutValues()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

ZoomInOutX-Methode

Funktion

Führt die Tastenfunktion "X-Achse zoomen +/-" des FunctionTrendControl aus.

Syntax

```
Ausdruck.ZoomInOutX()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

ZoomInOutY-Methode

Funktion

Führt die Tastenfunktion "Y-Achse zoomen +/-" des FunctionTrendControl aus.

Syntax

```
Ausdruck.ZoomInOutY()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

--

ZoomMove-Methode

Funktion

Führt die Tastenfunktion "Kurvenbereich verschieben" des OnlineTrendControl und FunctionTrendControl aus.

Syntax

```
Ausdruck.ZoomMove()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ScreenItem" zurückgibt.

Parameter

- -

1.14.6 Anhang

1.14.6.1 Fehlermeldungen aus dem Bereich Datenbanken

Einleitung

Bei der Zugriffen auf Datenbanken wird nach der Ausführung ein Wert zurückgeliefert. Dabei stellen Werte im Bereich "0x8..." eine Fehlermeldung dar. Werte ungleich "0x8..." stellen eine Statusmeldung dar.

Statusmeldungen

Folgende Statusmeldungen sind definiert :

0x0	OK
0x1	Die Funktion hat keine Fehler in der Parameterversorgung festgestellt und auch intern keine Fehler erkannt. Folgende Ursachen sind für diesen Rückgabewert möglich. Beim Verbinden von Datenbanken : - Im gegebenen Zeitfenster konnten keine Archive gefunden werden. - Im gegebenen Zeitfenster wurden zwar Archive gefunden, aber diese waren bereits verbunden. Beim Trennen von Datenbanken : - Im gegebenen Zeitfenster konnten keine verbundenen Archive gefunden werden. Es wird dabei nicht überprüft, ob überhaupt Archive verbunden sind.

Fehlermeldungen

Folgende Fehlermeldungen sind definiert (nur in Englisch):

Fehlercode	Fehlermeldung
0x80047200	WinCC is not activated
0x80047201	Invalid archive type
0x80047202	Invalid lower boundary
0x80047203	Invalid upper boundary
0x80047204	Path 'CommonArchiving' could not be created in the project path
0x80047205	Timeout, please retry

Fehlercode	Fehlermeldung
0x80047206	WinCC was deactivated
0x80047207	Wrong signification At least one database had a invalid signature and has not been attached.
0x80047208	Database could not be attached
0x80047209	Copy to 'CommonArchiving' is not possible.
0x8004720A	Invalide syntax for database filename.
0x8004720B	No list of databases.
0x8004720C	Database already detached.
0x8004720D	Database could not be detached.
0x8004720F	Unsigned database attached. At least one database without signature has been attached.
0x80047210	Path error : - Path invalid, - no *.MDF files found in specified path or - no permission to specified path.

Siehe auch

Remove-Methode (Seite 762)

Write-Methode (Seite 787)

Read-Methode (Seite 757)

Restore-Methode (Seite 768)

Logging-Objekt (Seite 131)

DataLogs-Objekt (Seite 124)

AlarmLogs-Objekt (Seite 121)

1.15 Beispiele zu VBScript

1.15.1 Beispiele zu VBScript

Einleitung

Im Folgenden finden Sie Anwendungsbeispiele zu VBS in WinCC. Im Teil "Beispiele in WinCC" sind Codebeispiele enthalten, mit denen Sie Ihre WinCC-Runtimeumgebung dynamisieren können. Diese Beispiele sind so konzipiert, dass Sie sie 1:1 in Ihre Projektierung übernehmen können.

Im Teil "Beispiele allgemein" sind Beispiele enthalten, mit denen Sie Ihre Microsoft-Umgebung beeinflussen können. Für die Ablauffähigkeit dieser Beispiele wird keine Gewährleistung und kein Support gegeben.

Siehe auch

Beispiele in WinCC (Seite 796)

1.15.2 Beispiele in WinCC

1.15.2.1 Beispiele in WinCC

Einleitung

Sie finden hier Beispiele zu Anwendungen von VBScript in WinCC zu folgenden Themen:

- Zugriff auf Objekte im Graphics Designer (z.B. Farb- oder Textänderung)
- Farbe von Objekte über RGB Farben setzen
- Sprachumschaltung projektieren
- Runtime deaktivieren
- Externes Programm starten
- Bildwechsel global projektieren (von Global Skript aus)
- Bildwechsel über Property projektieren
- Trace zur Diagnoseausgabe verwenden
- Wert einer Variablen setzen
- Wert einer Variablen lesen
- Den Erfolg einer Lese-/Schreibaktion in eine Variable prüfen
- Wert einer Variablen setzen asynchron

Siehe auch

Beispiel: So starten Sie eine Fremdapplikation (Seite 828)

Beispiel: So schreiben Sie Objekteigenschaften (Seite 806)

Beispiel: So lesen Sie Variablenwerte (Seite 803)

Beispiel: So schreiben Sie Variablenwerte (Seite 801)

Beispiel: So projektieren Sie eine Diagnoseausgabe über Trace (Seite 800)

Beispiel: So projektieren Sie einen Bildwechsel über Property (Seite 800)

Beispiel: So projektieren Sie einen Bildwechsel global (Seite 799)

Beispiel: So deaktivieren Sie Runtime (Seite 799)

Beispiel: So projektieren Sie eine Sprachumschaltung (Seite 798)

Beispiel: So bestimmen Sie die Farbe von Objekten (Seite 798)

Beispiel: So greifen Sie auf Objekte im Graphics Designer zu (Seite 797)

1.15.2.2 Beispiel: So greifen Sie auf Objekte im Graphics Designer zu

Einleitung

Sie können mit VBS in WinCC auf alle Objekte des Graphics Designer zugreifen, um die grafische Runtime-Umgebung zu dynamisieren. Dabei können Sie Grafikobjekte auf Bedienung dynamisieren (z.B. bei Mausklick auf einen Button), in Abhängigkeit einer Variablen oder zyklisch (z.B. blinken).

Die folgenden Beispiele zeigen Ihnen, wie Sie ein Grafikobjekt auf Mausklick verändern.

Vorgehensweise

Im folgenden Beispiel setzen Sie den Radius eines Kreises in Runtime bei Mausklick auf 20:

```
'VBS121
Dim objCircle
Set objCircle = ScreenItems("Circle1")
objCircle.Radius = 20
```

Hinweis

Die im Beispiel verwendete Ausdrucksweise ist nur im Graphics Designer ablauffähig. Für analoge Aktionen im Global Script adressieren Sie die Objekte über das HMIRuntime-Objekt.

Siehe auch

Beispiele in WinCC (Seite 796)

1.15.2.3 Beispiel: So bestimmen Sie die Farbe von Objekten

Einleitung

Die Farben von Grafikobjekten sind über RGB-Werte definiert (Rot/Grün/Blau). Sie können Farbwerte für Grafikobjekte setzen oder auslesen.

Vorgehensweise

Das folgende Beispiel setzt die Füllfarbe des Bildes "ScreenWindow1" auf Blau:

```
'VBS122
Dim objScreen
Set objScreen = HMIRuntime.Screens("ScreenWindow1")
objScreen.FillStyle = 131075
objScreen.FillColor = RGB(0, 0, 255)
```

Siehe auch

Beispiele in WinCC (Seite 796)

1.15.2.4 Beispiel: So projektieren Sie eine Sprachumschaltung

Einleitung

Sie können die Runtime-Sprache von WinCC mit VBS umschalten. Die üblichste Anwendung sind Buttons mit entsprechenden Länderkennzeichnungen, sie Sie auf der Startseite Ihres Projektes platzieren.

Die Runtime-Sprache geben Sie in VBS mit einer Länder-Kennung an, z.B. 1031 für Deutsch - Standard, 1033 für Englisch - USA etc. Eine Übersicht aller Länder-Kennungen finden Sie in den Grundlagen von VBScript unter dem Thema "Gebietsschema-ID (LCID)-Diagramm".

Vorgehensweise

Erstellen Sie am Ereignis "Mausklick" an einem Button eine VBS-Aktion und geben Sie folgenden Aktionscode ein, um die Runtime-Sprache auf Deutsch umzuschalten:

```
'VBS123
HMIRuntime.Language = 1031
```

Siehe auch

Beispiele in WinCC (Seite 796)

1.15.2.5 Beispiel: So deaktivieren Sie Runtime

Einleitung

Sie können mit VBS WinCC-Runtime beenden, z.B. auf Mausklick oder auch in Abhängigkeit von Variablenwerten oder anderen Ereignissen, z.B. mehrfacher fehlerhafter Eingabe eines Passwortes bei Runtime-Start.

Vorgehensweise

Das folgende Beispiel beendet WinCC Runtime:

```
'VBS124
HMIRuntime.Stop
```

Siehe auch

Beispiele in WinCC (Seite 796)

1.15.2.6 Beispiel: So projektieren Sie einen Bildwechsel global

Einleitung

Sie können globale Bildwechsel mit VBS realisieren, und so z.B. an einem Client in einem Verteilten System ein Bild eines Servers anzeigen. Sie müssen dazu dem Zielbild das Serverprefix des Servers voranstellen.

Vorgehensweise

Projektieren Sie zum Bildwechsel z.B. an einen Button folgenden Code:

```
'VBS125
HMIRuntime.BaseScreenName = "Serverprefix::New screen"
```

Siehe auch

Beispiele in WinCC (Seite 796)

1.15.2.7 Beispiel: So projektieren Sie einen Bildwechsel über Property

Einleitung

Wenn Sie in Ihrer Projektierung aufgeteilte Bilder verwenden, z.B. in einem Grundbild Titel und Bedienleiste der Bedienoberfläche und ein eingebettetes Bildfenster für die eigentliche Bildanzeige, projektieren Sie den Bildwechsel über die Eigenschaften des Bildfensters.

Die Property des Bildfensters "ScreenName" muss geändert werden, damit das andere Bild erscheint. Die Aktion und das Bildfenster müssen in demselben Bild projektiert sein.

Vorgehensweise

Im folgenden Beispiel zeigen Sie in einem Bildfenster "ScreenWindow" bei Ausführen der Aktion das Bild "test.pdl" an:

```
'VBS126
Dim objScrWindow
Set objScrWindow = ScreenItems("ScreenWindow")
objScrWindow.ScreenName = "test"
```

Siehe auch

Beispiele in WinCC (Seite 796)

1.15.2.8 Beispiel: So projektieren Sie eine Diagnoseausgabe über Trace

Einleitung

Wenn Sie ein GSC-Diagnosefenster in Ihr Bild eingefügt haben, können Sie Diagnoseausgaben mit dem Trace-Befehl in Runtime in dem Diagnosefenster ausgeben.

GSC-Diagnose gibt die in den Aktionen enthaltenen Trace-Methoden in der zeitlichen Reihenfolge ihres Aufrufs aus. Das gilt auch für Trace-Anweisungen in Prozeduren, die in Aktionen aufgerufen werden. Durch gezielten Einsatz von Trace-Anweisungen, beispielsweise zur Ausgabe von Variablenwerten, lässt sich so der Ablauf von Aktionen und den darin aufgerufenen Prozeduren verfolgen. Die Trace-Anweisung geben Sie in der Form "HMIRuntime.Trace(<Ausgabe>)" an.

Im GSC-Diagnosefenster werden Trace-Ausgaben von C und VBS ausgegeben.

Vorgehensweise

Das folgende Beispiel schreibt einen Text ins Diagnosefenster:

```
'VBS127
HMIRuntime.Trace "Customized error message"
```

Siehe auch

Beispiele in WinCC (Seite 796)

1.15.2.9 Beispiel: So schreiben Sie Variablenwerte

Einleitung

Sie können mit VBS den Wert einer Variablen in die Steuerung schreiben, z.B. auf Mausklick an einem Button zur Sollwertvorgabe, oder interne Variablenwerte setzen, um andere Aktionen auszulösen.

Nachfolgend werden verschiedene Schreibvarianten aufgeführt und erklärt.

Einfaches Schreiben

Im folgenden Beispiel wird ein Wert in die Variable "Tag1" geschrieben:

```
'VBS128
HMIRuntime.Tags("Tag1").Write 6
```

Dies ist die einfachste Form des Schreibens, da keine Objektreferenz erzeugt wird.

Schreiben mit Objektreferenz

Im folgenden Beispiel wird eine lokale Kopie des Tag-Objektes erzeugt und ein Wert in die Variable "Tag1" geschrieben:

```
'VBS129
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
objTag.Write 7
```

Das Referenzieren bietet den Vorteil, dass Sie vor dem Schreiben mit dem Tag-Objekt arbeiten können. Sie können den Variablenwert lesen, Berechnungen ausführen und wieder schreiben:

```
'VBS130
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
objTag.Read
objTag.Value = objTag.Value + 1
objTag.Write
```

Synchrones Schreiben

Standardmäßig wird der zu schreibende Wert an den Variablenhaushalt übergeben und die Abarbeitung der Aktion fortgesetzt. In manchen Fällen muss jedoch gewährleistet sein, dass der Wert auch tatsächlich geschrieben wurde, bevor die Aktion weiter abgearbeitet wird.

Diese Schreibart realisieren Sie, indem Sie bei dem zusätzlichen, optionalen Parameter den Wert 1 angeben:

```
'VBS131
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
objTag.Write 8,1
```

oder

```
'VBS132
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
objTag.Value = 8
objTag.Write ,1
```

Hinweis

Beachten Sie, dass der Aufruf im Vergleich zum Standardaufruf länger dauert. Die Dauer ist unter anderem vom Kanal und dem AS abhängig.

Diese Schreibart entspricht dem SetTagXXXWait()-Aufruf aus dem C-Skripting.

Schreiben mit Statusbehandlung

Um sicherzustellen, dass ein Wert erfolgreich geschrieben wurde, ist es notwendig, nach dem Schreiben eine Fehlerüberprüfung durchzuführen bzw. den Status der Variablen zu ermitteln.

Dies geschieht, indem nach dem Schreiben des Wertes die Eigenschaft LastError überprüft wird. Wenn die Überprüfung erfolgreich war, d.h. der Auftrag erfolgreich abgesetzt werden konnte, wird der Status der Variablen überprüft.

Bei einem Schreibauftrag wird nicht der aktuelle Status aus dem Prozess ermittelt. Um diesen zu ermitteln ist es erforderlich, die Variable zu lesen. Der nach dem Lesevorgang in der Eigenschaft QualityCode enthaltene Wert gibt Aufschluss über den Status der Variablen und weist ggf. auf eine ausgefallene AS-Verbindung hin.

Im folgenden Beispiel wird die Variable "Tag1" geschrieben. Wenn es beim Schreiben zu einem Fehler kommt, werden Fehlerwert und Fehlerbeschreibung im Global Script Diagnosefenster ausgegeben. Anschließend wird der QualityCode überprüft. Wenn der QualityCode nicht OK (0x80) ist, wird er im Diagnosefenster ausgegeben.

```
'VBS133
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
objTag.Write 9
If 0 <> objTag.LastError Then
HMIRuntime.Trace "Error: " & objTag.LastError & vbCrLf & "ErrorDescription: " &
objTag.ErrorDescription & vbCrLf
Else
objTag.Read
If &H80 <> objTag.QualityCode Then
HMIRuntime.Trace "QualityCode: 0x" & Hex(objTag.QualityCode) & vbCrLf
End If
End If
```

Hinweis

Nach dem Schreiben einer Variablen wird die Eigenschaft QualityCode des lokalen Tag-Objektes auf "BAD Out of Service" gesetzt, da nicht bekannt ist, welchen QualityCode die Variable im Prozess führt.

Der QualityCode kann aus VBS heraus nicht geschrieben werden.

Siehe auch

Write-Methode (Seite 787)

Beispiele in WinCC (Seite 796)

1.15.2.10 Beispiel: So lesen Sie Variablenwerte

Einleitung

Sie können mit VBS den Wert einer Variablen lesen und weiterverarbeiten. So können Sie z.B. auf Mausklick an einem Button etwas über den Zustand der Anlage erfahren oder Berechnungen ausführen.

Nachfolgend werden verschiedene Lesevarianten aufgeführt und erklärt.

Einfaches Lesen

Im folgenden Beispiel wird der Wert der Variablen "Tag1" gelesen und im Global Script Diagnosefenster ausgegeben:

```
'VBS134
HMIRuntime.Trace "Value: " & HMIRuntime.Tags("Tag1").Read & vbCrLf
```

Dies ist die einfachste Form des Lesens, da keine Objektreferenz erzeugt wird.

Lesen mit Objektreferenz

Im folgenden Beispiel wird eine lokale Kopie des Tag-Objektes erzeugt, der Variablenwert gelesen und im Global Script Diagnosefenster ausgegeben:

```
'VBS135
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
HMIRuntime.Trace "Value: " & objTag.Read & vbCrLf
```

Das Referenzieren bietet den Vorteil, dass Sie mit dem Tag-Objekt arbeiten können. Sie können den Variablenwert lesen, Berechnungen ausführen, und wieder schreiben:

```
'VBS136
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
objTag.Read
objTag.Value = objTag.Value + 1
objTag.Write
```

Mit der Read-Methode gelesene Prozessvariablen werden dem Abbild hinzugefügt, sie werden von nun an zyklisch aus dem Automatisierungs-System (AS) angefordert. Befindet sich die Variable bereits im Abbild, wird der darin enthaltene Wert geliefert.

Bei Bildabwahl werden die Variablen wieder abgemeldet.

Hinweis

Wenn eine Variable in einer Global Script Aktion angefordert wird, bleibt sie über die gesamte Runtime-Laufzeit von WinCC angemeldet.

Direktes Lesen

Standardmäßig werden Variablenwerte aus dem Variablenabbild gelesen. In manchen Situationen kann es jedoch erforderlich sein, den Wert direkt aus dem AS zu lesen, z.B. um schnelle Vorgänge zu synchronisieren.

Setzt man beim Lesen den optionalen Parameter auf 1, so wird die Variable nicht zyklisch angemeldet, sondern der Wert wird einmalig aus dem AS angefordert.

```
'VBS137
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
HMIRuntime.Trace "Value: " & objTag.Read(1) & vbCrLf
```

Hinweis

Beachten Sie, dass der Aufruf im Vergleich zum Standardaufruf länger dauert. Die Dauer ist unter anderem vom Kanal und dem AS abhängig.

In zyklischen C-Aktionen ist diese Aufrufart zu vermeiden, da dies der Hauptgrund für Performance-Probleme ist.

Diese Leseart entspricht dem GetTagXXXWait()-Aufruf aus dem C-Scripting

Lesen mit Statusbehandlung

Um sicherzustellen, dass ein Wert gültig ist, sollte nach dem Lesen eine Überprüfung durchgeführt werden. Dies geschieht, indem der QualityCode kontrolliert wird.

Im folgenden Beispiel wird die Variable "myWord" gelesen und anschließend der QualityCode überprüft. Wenn der QualityCode nicht OK (0x80) entspricht, werden die Eigenschaften LastError, ErrorDescription und QualityCode im Global Script Diagnosefenster ausgegeben.

```
'VBS138
Dim objTag
Set objTag = HMIRuntime.Tags("Tag1")
objTag.Read
If &H80 <> objTag.QualityCode Then
HMIRuntime.Trace "Error: " & objTag.LastError & vbCrLf & "ErrorDescription: " &
objTag.ErrorDescription & vbCrLf & "QualityCode: 0x" & Hex(objTag.QualityCode) &vbCrLf
Else
HMIRuntime.Trace "Value: " & objTag.Value & vbCrLf
End If
```

Hinweis

Wenn ein Fehler beim Lesen auftritt, wird der QualityCode auf "BAD Out of Service" gesetzt. Beim Lesen ist es daher ausreichend, den QualityCode zu prüfen.

Siehe auch

Read-Methode (Seite 757)

Beispiele in WinCC (Seite 796)

1.15.2.11 Beispiel: So schreiben Sie Objekteigenschaften

Einleitung

VBS bietet Zugriff auf die Eigenschaften alles Bildobjekte des Graphics Designer. Sie können Eigenschaften auslesen oder zur Runtime verändern.

Die folgenden Beispiele zeigen Ihnen verschiedene Zugriffsformen.

Einfaches Setzen einer Eigenschaft

Im folgenden Beispiel wird die Híntergrundfarbe des im Bild enthaltenen Objektes "Rectangle1" auf rot gesetzt:

```
'VBS139
ScreenItems("Rectangle1").BackColor = RGB(255,0,0)
Dies ist die einfachste Form des Schreibens, da keine Objektreferenz erzeugt wird.
```

Hinweis

Wenn Sie ohne eine Objektreferenz arbeiten, werden im Intellisense nur die Standard-Eigenschaften aufgeführt.

Die in diesem Beispiel verwendete Ausdrucksweise ist nur im Graphics Designer ablauffähig. Für analoge Aktionen im Global Script adressieren Sie die Objekte über das HMIRuntime-Objekt.

Setzen einer Eigenschaft mit Objektreferenz

Im folgenden Beispiel wird eine Referenz auf das im Bild enthaltene Objekt "Rectangle1" erzeugt und mit Hilfe der VBS-Standardfunktion RGB() die Hintergrundfarbe auf rot gesetzt:

```
'VBS140
Dim objRectangle
Set objRectangle = ScreenItems("Rectangle1")
objRectangle.BackColor = RGB(255,0,0)
```

Das Referenzieren bietet sich an, wenn Sie mehrere Eigenschaften eines Objektes ändern wollen. Auf diese Art bekommen Sie im Intellisense alle Eigenschaften des Objektes aufgelistet.

Hinweis

Die in diesem Beispiel verwendete Ausdrucksweise ist nur im Graphics Designer ablauffähig. Für analoge Aktionen im Global Script adressieren Sie die Objekte über das HMIRuntime-Objekt.

Eigenschaften über Bildfenster hinweg setzen

VBS im Graphics Designer bietet zwei Möglichkeiten der bildübergreifenden Adressierung:

- über das Screen-Objekt eines Bildfensters mit "ScreenItems"
- vom Grundbild aus mit "HMIRuntime.Screens"

Referenzierung über das Bildfenster

Im folgendem Beispiel wird in einem unterlagerten Bildfenster die Farbe eines Rechtecks geändert. Das Skript wird in dem Bild "BaseScreen" ausgeführt, in dem sich das Bildfenster "ScreenWindow1" befindet. Im Bildfenster wird ein Bild angezeigt, das ein Objekt vom Typ "Rechteck" mit dem Name "Rectangle1" enthält.

```
'VBS199
Sub OnLButtonUp(ByVal Item, ByVal Flags, ByVal x, ByVal y)
Dim objRectangle
Set objRectangle = ScreenItems("ScreenWindow1").Screen.ScreenItems("Rectangle1")
objRectangle.BackColor = RGB(255,0,0)
End Sub
```

Referenzierung vom Grundbild aus

Sie können das Bild mit dem zu ändernden Objekt über HMIRuntime.Screens referenzieren. Die Angabe des Bildes erfolgt dabei relativ zum Grundbild über den folgenden Zugriffsschlüssel:

[<Grundbildname>.]<Bildfenstername>[:<Bildname>]... .<Bildfenstername>[:<Bildname>]

Im folgenden Beispiel wird eine Referenz auf das im Bild "Screen2" enthaltene Objekt "Rectangle1" erzeugt und die Hintergrundfarbe auf rot gesetzt.

Das Bild "Screen2" befindet sich dabei in "Screen1". "Screen1" wird im Grundbild "BaseScreen" angezeigt.

```
'VBS141
Dim objRectangle
```

```
Set objRectangle =
HMIRuntime.Screens("BaseScreen.ScreenWindow1:Screen1.ScreenWindow1:Screen2").ScreenItems("
Rectangle1")
objRectangle.BackColor = RGB(255,0,0)
```

Es ist nicht notwendig, die Bildnamen mit anzugeben. Über die Bildfensternamen ist es möglich, ein Bild eindeutig zu adressieren. Es reicht deshalb aus, die Namen der Bildfenster anzugeben, wie im nachfolgenden Beispiel:

```
'VBS142
Dim objRectangle
Set objRectangle =
HMIRuntime.Screens("ScreenWindow1.ScreenWindow2").ScreenItems("Rectangle1")
objRectangle.BackColor = RGB(255,0,0)
```

Über diese Art der Adressierung ist es möglich, Objekte in Bildfenstern mit wechselnden Bildern anzusprechen. Dies ist besonders für die Bildbausteintechnik interessant.

Eigenschaft über Rückgabewert dynamisieren

Sie können Aktionen an Eigenschaften nicht nur durch Ereignis-Trigger oder zyklische Trigger auslösen, sondern Eigenschaften auch direkt über eine Aktion dynamisieren.

Im folgenden Beispiel wird die Hintergrundfarbe eines Objektes über einen Rückgabewert dynamisiert. Der übergebene Wert kann z.B. aus einer Auswertung von Ereignissen in der Steuerung stammen und zur grafischen Darstellung eines Betriebszustandes eingesetzt werden:

```
'VBS146
Function BackColor_Trigger(ByVal Item)
BackColor_Trigger = RGB(125,0,0)
End Function
```

Hinweis

Wenn Sie mit einer VBS-Aktion eine Objekteigenschaft über den Rückgabewert eines Skriptes dynamisieren, wird der Wert der Objekteigenschaft nur dann geschrieben, wenn er sich im Vergleich zum letzten Skript-Durchlauf geändert hat. Dabei wird nicht berücksichtigt, ob der Wert von einer anderen Stelle verändert wurde.

Deshalb dürfen Eigenschaften, welche mit einer VBS-Aktion über den Rückgabewert dynamisiert wurden, nicht von anderer Stelle (z.B. anderen C- oder VBS-Scripten) verändert werden.

Wenn Sie dies nicht beachten, dann können falsche Werte angezeigt werden.

Siehe auch

VBS Referenz (Seite 113)

Beispiele in WinCC (Seite 796)

1.15.2.12 Beispiel: So starten Sie eine Aktion am Server (Logging-Objekt)

Einleitung

In Mehrplatzprojekten funktioniert das Logging-Objekt derzeit nur am Server. In dem folgenden Beispiel wird gezeigt wie man vom Client aus eine Aktion am Server startet und somit auch am Client Archivsegmente einlagern bzw. löschen kann.

Das Beispiel zeigt eine globale Aktion die mit einer Steuervariable gestartet wird. Der Inhalt der Steuervariable bestimmt ob die Methode "Restore" oder die Methode "Remove" aufgerufen wird. Am Ende der Aktion wird die Steuervariable auf "0" gesetzt.

Eine Abfrage verhindert, dass die Aktion an Client-Rechnern gestartet wird.

Pfad und Zeitraum werden durch interne Variablen übergeben.

Als Pfadangabe kann auch eine Netzwerkfreigabe angegeben werden. Die einzulagernden Archivsegmente müssen also nicht lokal am Server liegen. Es muss jedoch gewährleistet sein dass der Server korrekt auf den Pfad zugreifen kann.

Hinweis

Das Beispiel zeigt einen Lösungsvorschlag und ist bei Bedarf anzupassen.

Vorgehensweise

1. Legen Sie im WinCC Explorer die folgenden internen Variablen mit projektweiter Aktualisierung an:
   - StartLogging (Vorzeichenloser 8-Bit Wert)
   - SourcePath (Textvariable 8-Bit Zeichensatz)
   - TimeFrom (Textvariable 8-Bit Zeichensatz)
   - TimeTo (Textvariable 8-BBit Zeichensatz)
   - RetVal (Vorzeichenbehafteter 32-Bit Wert)
2. Erzeugen Sie eine globale VBS Aktion und tragen Sie als Variablentrigger die Variable 'StartLogging' mit dem Zyklus Bei Änderung ein.
3. Kopieren Sie das nachfolgende Script in die Aktion

```
'VBS180
Dim StartLogging
Dim SourcePath
Dim TimeFrom
Dim TimeTo
Dim RetVal
'Exit when running on client
```

```
If (Left(HMIRuntime.ActiveProject.Path, 1) = "\") Then
Exit Function
End If
'read parameters
StartLogging = HMIRuntime.Tags("StartLogging").Read
SourcePath = HMIRuntime.Tags("SourcePath").Read(1)
TimeFrom = HMIRuntime.Tags("TimeFrom").Read(1)
TimeTo = HMIRuntime.Tags("TimeTo").Read(1)
'restore or remove depends on the parameter
If (StartLogging = 1) Then
RetVal = HMIRuntime.Logging.Restore(SourcePath, TimeFrom, TimeTo, -1)
HMIRuntime.Tags("RetVal").Write RetVal, 1
HMIRuntime.Tags("StartLogging").Write 0,1
Elseif (StartLogging = 2) Then
RetVal = HMIRuntime.Logging.Remove(TimeFrom, TimeTo, -1)
HMIRuntime.Tags("RetVal").Write RetVal, 1
HMIRuntime.Tags("StartLogging").Write 0,1
End If
```

Die Aktion kann auf einem Client z.B. mit folgender Aktion gestartet werden. Bitte beachten Sie, dass die Parameter vor dem Setzen der Steuervariable geschrieben werden müssen.

```
'VBS181
'set parameters
HMIRuntime.Tags("SourcePath").Write "\\client_pc\temp",1
HMIRuntime.Tags("TimeFrom").Write "2004",1
HMIRuntime.Tags("TimeTo").Write "2004",1
'start action
HMIRuntime.Tags("StartLogging").Write 1,1
```

Hinweis

Die Variablen werden überwiegend im Modus "direkt" geschrieben und gelesen. Dadurch werden die Abläufe synchronisiert. Da es sich um interne Variablen handelt, kann dieser Modus bedenkenlos eingesetzt werden.

1.15.2.13 Dynamisieren der Controls

Beispiel: So rufen Sie Methoden eines ActiveX-Controls auf

Einleitung

Die folgenden Beispiele zeigen Ihnen, wie Sie Methoden und Eigenschaften eines ActiveX-Controls, das in ein WinCC-Bild eingebettet ist, aufrufen.

Beispiel 1: WinCC FunctionTrendControl

Dieses Beispiel füllt die Kurve "Trend 1" des FuntionTrendControls "Control1" mit Werten, die eine Parabel beschreiben.

Um eine Kurve mit VBS zu dynamisieren, stellen Sie im Konfigurationsdialog des Controls in der Registerkarte "Datenanbindung" unter "Datenversorgung" "0 - keine" ein.

```
'VBS300
Dim lngFactor
Dim dblAxisX
Dim dblAxisY
Dim objTrendControl
Dim objTrend

Set objTrendControl = ScreenItems("Control1")
Set objTrend = objTrendControl.GetTrend("Trend 1")

For lngFactor = -100 To 100
dblAxisX = CDbl(lngFactor * 0.02)
dblAxisY = CDbl(dblAxisX * dblAxisX + 2 * dblAxisX + 1)
objTrend.InsertData dblAxisX, dblAxisY
Next
```

Beispiel 2: WinCC FunctionTrendControl mit Werteversorgung über Array

In diesem Beispiel wird die Kurve "Trend 1" des FunctionTrendControls "Control1" mit Werten, die in Arrays abgelegt sind, versorgt.

Um eine Kurve mit VBS zu dynamisieren, stellen Sie im Konfigurationsdialog des Controls in der Registerkarte "Datenanbindung" unter "Datenversorgung" "0 - keine" ein.

```
'VBS301
Dim lngIndex
Dim dblAxisX(100)
Dim dblAxisY(100)
Dim objTrendControl
Dim objTrend
Set objTrendControl = ScreenItems("Control1")
Set objTrend = objTrendControl.GetTrend("Trend 1")
For lngIndex = 0 To 100
dblAxisX(lngIndex) = CDbl(lngIndex * 0.8)
dblAxisY(lngIndex) = CDbl(lngIndex)
Next
objTrend.InsertData dblAxisX, dblAxisY
```

Beispiel 3: WinCC FunctionTrendControl (vor WinCC V7)

Dieses Beispiel füllt das FunctionTrendControl namens "Control1" mit Werten, die eine Parabel beschreiben.

```
'VBS111
Dim lngFactor
Dim dblAxisX
Dim dblAxisY
Dim objTrendControl
Set objTrendControl = ScreenItems("Control1")
For lngFactor = -100 To 100
dblAxisX = CDbl(lngFactor * 0.02)
dblAxisY = CDbl(dblAxisX * dblAxisX + 2 * dblAxisX + 1)
objTrendControl.DataX = dblAxisX
objTrendControl.DataY = dblAxisY
objTrendControl.InsertData = True
Next
```

Beispiel 4: WinCC FunctionTrendControl mit Werteversorgung über Array (vor WinCC V7)

In diesem Beispiel wird ein FunctionTrendControl namens "Control1" mit 100 Wertepaaren versorgt. Damit die Wertepaare korrekt übergeben werden, darf die Übergabe z.B. in "dblAxisXY" nicht direkt erfolgen, sondern über eine Zwischenvariable z.B. "varTemp".

```
'VBS152
Dim lngIndex
Dim dblXY(1)
Dim dblAxisXY(100)
Dim varTemp
Dim objTrendControl
Set objTrendControl = ScreenItems("Control1")
For lngIndex = 0 To 100
 dblXY(0) = CDbl(lngIndex * 0.8)
 dblXY(1) = CDbl(lngIndex)
 dblAxisXY(lngIndex) = dblXY
Next
varTemp = (dblAxisXY)
objTrendControl.DataXY = varTemp
objTrendControl.InsertData = True
```

Beispiel 5: Microsoft Web Browser

Dieses Beispiel steuert den MS Web Browser.

```
'VBS112
Dim objWebBrowser
Set objWebBrowser = ScreenItems("WebControl")
objWebBrowser.Navigate "http://www.siemens.de"
```

```
...
objWebBrowser.GoBack
...
objWebBrowser.GoForward
...
objWebBrowser.Refresh
...
objWebBrowser.GoHome
...
objWebBrowser.GoSearch
...
objWebBrowser.Stop
...
```

Hinweis

Setzen Sie die durch Punkte getrennten Anweisungen in eigene Prozeduren. Deklaration und Zuweisungen müssen immer davor stehen.

Siehe auch

Allgemeine Beispiele zu VBScript (Seite 823)

Beispiel: So projektieren Sie einen benutzerdefinierten Toolbar-Button mit einem selbsterstellten Selektionsdialog

Einleitung

Im folgenden Beispiel erzeugen Sie einen benutzerdefinierten Toolbar-Button eines OnlineTrendControls. Sie projektieren an diesen Toolbar-Button einen selbsterstellten Selektionsdialog, mit dem Sie wahlweise einen von zwei verschiedenen Zeitbereichen des OnlineTrendControls einstellen können.

Voraussetzung

- Der Graphics Designer ist geöffnet.
- Im Tag Logging-Editor ist ein Archiv angelegt.

WinCC OnlineTrendControl einfügen und konfigurieren

1. Legen Sie im Graphics Designer ein neues Prozessbild an.
2. Speichern Sie das Prozessbild unter "OnlineTrend.pdl".
3. Fügen Sie ein Control "WinCC OnlineTrendControl" in das Prozessbild ein.
4. Wählen Sie im Kontextmenü des Controls den Befehl "Konfiurationsdialog...". Der Dialog "Eigenschaften von WinCC OnlineTrendControl" wird geöffnet.

5. Verbinden Sie in der Registerkarte "Kurve" unter "Datenanbindung" die Kurve mit einer Archivvariablen.
6. Erzeugen Sie in der Registerkarte "Symbolleiste" unter "Tastenfunktionen" einen neuen benutzerdefinierten Toolbar-Button mit der Objekt-ID "1001" für das OnlineTrendControl.
7. Klicken Sie auf "Übernehmen", um die Änderungen zu speichern.
8. Klicken Sie auf "OK", um den Dialog zu schließen.
9. Wählen Sie im Kontextmenü des Controls den Befehl "Eigenschaften".
   Der Dialog "Objekteigenschaften" wird geöffnet.
10. Geben Sie als Objektnamen für das Control "Control1" ein.
11. Wählen Sie in den Objekteigenschaften des Control "Control1" die Registerkarte "Ereignis".
12. Projektieren Sie an das Objektereignis "OnToolbarButtonClicked" das VB-Skript "VBS Aktion am Ereignis "OnToolbarButtonClicked" des benutzerdefinierten Toolbar-Button erzeugen (VBS302)".
13. Schließen Sie den Dialog "Objekteigenschaften".

Prozessbild für Selektionsdialog erstellen

1. Legen Sie im Graphics Designer ein neues Prozessbild an.
2. Speichern Sie das Prozessbild unter "Selektionsdialog.pdl".
3. Klicken Sie im Kontextmenü des Prozessbildes auf "Eigenschaften".
   Der Dialog "Objekteigenschaften" wird geöffnet.
4. Stellen Sie unter "Geometrie" für die Attribute "Bildbreite" und "Bildhöhe" den Wert "200" ein.
5. Schließen Sie den Dialog "Objekteigenschaften".
6. Fügen Sie in das Prozessbild zwei Objekte "Button" ein.
7. Geben Sie als Text für die Button "Morning" bzw. "Afternoon" ein.

Button für Selektionsdialog dynamisieren

1. Wählen Sie in den Objekteigenschaften des Button "Morning" die Registerkarte "Ereignis".
2. Projektieren Sie an das Ereignis "Mausklick" das VB-Skript "VBS-Aktion am Ereignis "Mausklick" des Button "Morning" erzeugen (VBS303)".
3. Schließen Sie den Dialog "Objekteigenschaften".
4. Wählen Sie in den Objekteigenschaften des Button "Afternoon" die Registerkarte "Ereignis".
5. Projektieren Sie an das Ereignis "Mausklick" das VB-Skript "VBS-Aktion am Ereignis "Mausklick" des Button "Afternoon" erzeugen (VBS304)".
6. Schließen Sie den Dialog "Objekteigenschaften".

Bildfenster einfügen und konfigurieren

1. Fügen Sie in das Prozessbild "OnlineTrend.pdl" ein Objekt "Bildfenster" ein.
2. Wählen Sie im Kontextmenü des Bildfensters den Befehl "Eigenschaften". Der Dialog "Objekteigenschaften" wird geöffnet.
3. Geben Sie als Objektnamen für das Bildfenster "PictureWindow1" ein.
4. Stellen Sie unter "Sonstiges" das Attribut "Anzeige" auf "nein".
5. Wählen Sie unter "Sonstiges" für das Attribut "Bildname" das Prozessbild "Selektionsdialog.pdl" aus.
6. Schließen Sie den Dialog "Objekteigenschaften".

VBS Aktion am Ereignis "OnToolbarButtonClicked" des benutzerdefinierten Toolbar-Button erzeugen (VBS302)

```
'VBS302
'Open selection window if Toolbarbutton with ID 1001 is pressed
If lId = 1001 Then
ScreenItems("PictureWindow1").Visible = True
End If
```

VBS-Aktion am Ereignis "Mausklick" des Button "Morning" erzeugen (VBS303)

```
'VBS303
Dim obj
Set obj = Parent.Parent.ScreenItems("Control1")

'choose time axis, stop update, set begin time and time range
obj.TimeAxisName = "Time axis 1"
obj.TimeAxisActualize = False
obj.TimeAxisBeginTime = CStr(Date & "4:00:00")
obj.TimeAxisTimeRangeBase = 3600000
obj.TimeAxisTimeRangeFactor = 8

'close the selection window
Parent.Visible = False
```

VBS-Aktion am Ereignis "Mausklick" des Button "Afternoon" erzeugen (VBS304)

```
'VBS304
Dim obj
Set obj = Parent.Parent.ScreenItems("Control1")

'choose time axis, stop update, set begin time and time range
obj.TimeAxisName = "Time axis 1"
```

```
obj.TimeAxisActualize = False
obj.TimeAxisBeginTime = CStr(Date & "12:00:00")
obj.TimeAxisTimeRangeBase = 3600000
obj.TimeAxisTimeRangeFactor = 8

'close the selection window
Parent.Visible = False
```

Beispiel: So fügen Sie Elemente zu einem leeren WinCC OnlineTrendControl hinzu

Einleitung

Im folgenden Beispiel fügen Sie die Elemente Kurvenfenster, Wertachse, Zeitachse und Kurven in ein leeres WinCC OnlineTrendControl ein.

Voraussetzung

- Der Graphics Designer ist geöffnet.
- Im Tag Logging-Editor ist ein Archiv mit drei Archivvariablen angelegt.

WinCC OnlineTrendControl einfügen und konfigurieren

1. Legen Sie im Graphics Designer ein neues Prozessbild an.
2. Fügen Sie ein Control "WinCC OnlineTrendControl" in das Prozessbild ein.
3. Wählen Sie im Kontextmenü des Controls den Befehl "Konfigurationdialog...". Der Dialog "Eigenschaften von WinCC OnlineTrendControl" wird geöffnet.
4. Löschen Sie in der Registerkarte "Kurven" im Bereich "Kurven" das Standard-Kurvenfenster "Kurve 1".
5. Klicken Sie auf "Übernehmen", um die Änderungen zu speichern.
6. Klicken Sie auf "OK", um den Dialog zu schließen.
7. Wählen Sie im Kontextmenü des Controls den Befehl "Eigenschaften". Der Dialog "Objekteigenschaften" wird geöffnet.
8. Geben Sie als Objektnamen für das Control "Control1" ein.
9. Schließen Sie den Dialog "Objekteigenschaften".

Button einfügen und konfigurieren

1. Fügen Sie ein Objekt "Button" in das Prozessbild ein.
2. Geben Sie als Text für den Button "Elemente einfügen" ein.
3. Wählen Sie im Kontextmenü des Button den Befehl "Eigenschaften". Der Dialog "Objekteigenschaften" wird geöffnet.
4. Wählen Sie in den Objekteigenschaften des Button die Registerkarte "Ereignis".

5. Projektieren Sie an das Ereignis "Mausklick" das VB-Skript "VBS Aktion am Ereignis "Mausklick" des Button erzeugen (VBS305)".
6. Schließen Sie den Dialog "Objekteigenschaften".

VBS Aktion am Ereignis "Mausklick" des Button erzeugen (VBS305)

```
'VBS305
Dim objTrendControl
Dim objTrendWindow
Dim objTimeAxis
Dim objValueAxis
Dim objTrend

'create reference to TrendControl
Set objTrendControl = ScreenItems("Control1")

'create reference to new window, time and value axis
Set objTrendWindow =
objTrendControl.GetTrendWindowCollection.AddItem("myWindow")
Set objTimeAxis =
objTrendControl.GetTimeAxisCollection.AddItem("myTimeAxis")
Set objValueAxis =
objTrendControl.GetValueAxisCollection.AddItem("myValueAxis")

'assign time and value axis to the window
objTimeAxis.TrendWindow = objTrendWindow.Name
objValueAxis.TrendWindow = objTrendWindow.Name

'add new trend and assign propertys
Set objTrend = objTrendControl.GetTrendCollection.AddItem("myTrend1")
objTrend.Provider = 1
objTrend.TagName = "TestArchive\ArchivTag1"
objTrend.Color = RGB(255,0,0)
objTrend.TrendWindow = objTrendWindow.Name
objTrend.TimeAxis = objTimeAxis.Name
objTrend.ValueAxis = objValueAxis.Name

'add new trend and assign propertys
Set objTrend = objTrendControl.GetTrendCollection.AddItem("myTrend2")
objTrend.Provider = 1
objTrend.TagName = "TestArchive\ArchivTag2"
objTrend.Color = RGB(0,255,0)
objTrend.TrendWindow = objTrendWindow.Name
objTrend.TimeAxis = objTimeAxis.Name
objTrend.ValueAxis = objValueAxis.Name

'add new trend and assign propertys
Set objTrend = objTrendControl.GetTrendCollection.AddItem("myTrend3")
objTrend.Provider = 1
objTrend.TagName = "TestArchive\ArchivTag3"
objTrend.Color = RGB(0,0,255)
objTrend.TrendWindow = objTrendWindow.Name
objTrend.TimeAxis = objTimeAxis.Name
```

```
objTrend.ValueAxis = objValueAxis.Name
```

Hinweis

Ersetzen Sie in dem VB-Skript das verwendete Archiv und die Archivvariablen "Archive \ArchiveTagX" durch die Namen des angelegten Archivs und der Archivvariablen.

Beispiel: So fügen Sie einem leeren OnlineTrendControl eine Kurve und eine Sollkurve hinzu

Einleitung

Im folgenden Beispiel fügen Sie in ein leeres WinCC OnlineTrendControl eine Kurve und eine Sollkurve ein. Für die Kurven werden in einem Kurvenfenster Zeitachse und Wertachse hinzugefügt.

Voraussetzung

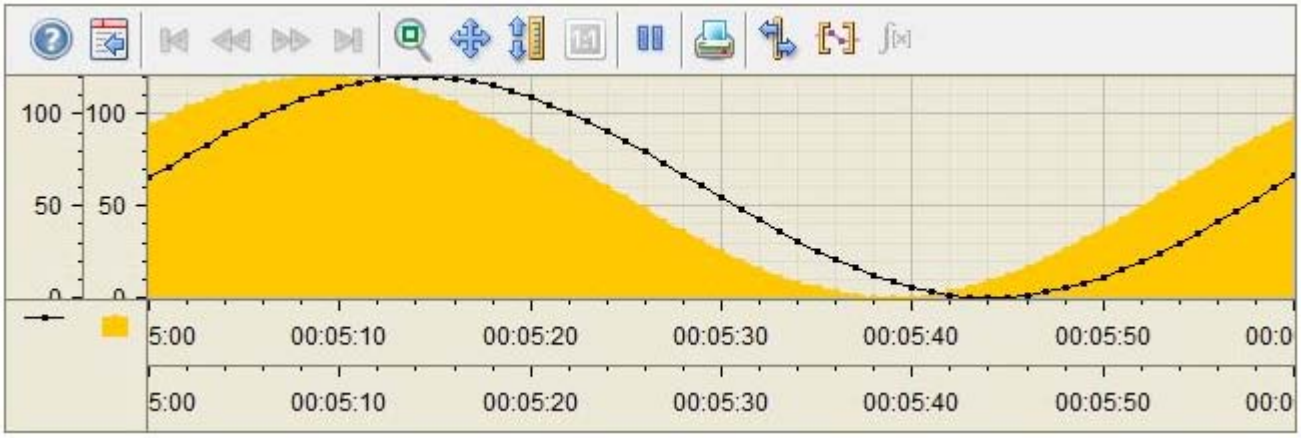
- Im Graphics Designer ist ein "WinCC OnlineTrendControl" mit dem Namen "Control2" in das Prozessbild eingefügt.
- Im Graphics Designer ist ein Button eingefügt. Sie haben für den Button z.B. das Ereignis "Mausklick" projektiert, mit einer VBS-Aktion und folgendem Skript.

Beispiel

```
'VBS352
Dim objTrendControl
Dim objTrendWindow
Dim objTimeAxis
Dim objValueAxis
Dim objTrend
'tags used to generate trend data
Dim dtCurrent
Dim dblCurrent
Dim lIndex
Dim vValues(360)
Dim vTimeStamps(360)

'create reference to TrendControl
Set objTrendControl = ScreenItems("Control2")

'---- reference trend ----
'create reference to new window, time and value axis
Set objTrendWindow = objTrendControl.GetTrendWindowCollection.AddItem("myWindow")
Set objTimeAxis = objTrendControl.GetTimeAxisCollection.AddItem("myRefTimeAxis")
Set objValueAxis = objTrendControl.GetValueAxisCollection.AddItem("myRefValueAxis")

'assign time and value axis to the window
```

```
objTimeAxis.TrendWindow = objTrendWindow.Name
objTimeAxis.ShowDate = False
objValueAxis.TrendWindow = objTrendWindow.Name

'add trend and assign propertys
Set objTrend = objTrendControl.GetTrendCollection.AddItem("myRefTrend")
objTrend.Provider = 0
objTrend.Color = RGB(0,0,0)
objTrend.TrendWindow = objTrendWindow.Name
objTrend.TimeAxis = objTimeAxis.Name
objTrend.ValueAxis = objValueAxis.Name

'generate values for reference trend
dtCurrent = CDate("23.11.2006 00:00:00")
For lIndex = 0 To 360
 vValues(lIndex) = ( Sin(dblCurrent) * 60 ) + 60
 vTimeStamps(lIndex) = dtCurrent
 dblCurrent = dblCurrent + 0.105
 dtCurrent = dtCurrent + CDate ("00:00:01")
Next

'insert data to the reference trend
objTrend.RemoveData
objTrend.InsertData vValues, vTimeStamps

'---- data trend ----
'add time and value axis to the existing window
Set objTimeAxis = objTrendControl.GetTimeAxisCollection.AddItem("myTimeAxis")
Set objValueAxis = objTrendControl.GetValueAxisCollection.AddItem("myValueAxis")

'assign time and value axis to the window
objTimeAxis.TrendWindow = objTrendWindow.Name
objTimeAxis.ShowDate = False
objValueAxis.TrendWindow = objTrendWindow.Name

'add new trend and assign propertys
Set objTrend = objTrendControl.GetTrendCollection.AddItem("myTrend")
objTrend.Provider = 0
objTrend.Color = RGB(255,200,0)
objTrend.Fill = True
objTrend.TrendWindow = objTrendWindow.Name
objTrend.TimeAxis = objTimeAxis.Name
objTrend.ValueAxis = objValueAxis.Name

'generate values for data trend
dtCurrent = CDate("23.11.2006 00:00:00")
For lIndex = 0 To 360
 vValues(lIndex) = ( Sin(dblCurrent) * 60 ) + 60
 vTimeStamps(lIndex) = dtCurrent
 dblCurrent = dblCurrent + 0.106
 dtCurrent = dtCurrent + CDate ("00:00:01")
Next

'insert values to the data trend
objTrend.RemoveData
objTrend.InsertData vValues, vTimeStamps
```

Ergebnis



Beispiel: So fügen Sie Elemente zu einem WinCC OnlineTableControl hinzu

Einleitung

Im folgenden Beispiel fügen Sie Wertspalten mit Eigenschaften in ein leeres WinCC OnlineTableControl ein und verbinden die Spalten mit Archivvariablen.

Voraussetzung

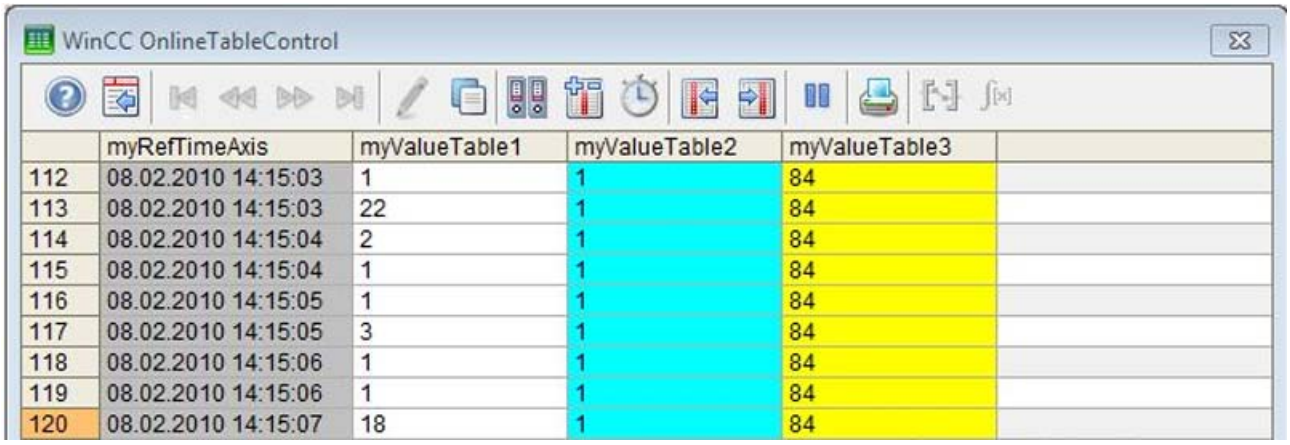
- Im Editor "Tag Logging" ist ein Archiv mit drei Archivvariablen angelegt.
- Im Graphics Designer ist ein "WinCC OnlineTableControl" mit dem Namen "Control2" in das Prozessbild eingefügt.
- Im Graphics Designer ist ein Button eingefügt. Sie haben für den Button z.B. das Ereignis "Mausklick" projektiert, mit einer VBS-Aktion und folgendem Skript.

Beispiel

```
'VBS351
Dim objTableControl
Dim objTimeColumn
Dim objValueColumn
Dim objTrend

'create reference to TableControl and enable BackColor
Set objTableControl = ScreenItems("Control2")
objTableControl.UseColumnBackColor = True

'create reference to new TimeColumn and assign column length
Set objTimeColumn = objTableControl.GetTimeColumnCollection.AddItem("myRefTimeAxis")
objTimeColumn.Length = 20
```

```
'add new ValueColumn and assign propertys
Set objValueColumn = objTableControl.GetValueColumnCollection.AddItem("myValueTable1")
objValueColumn.Provider = 1
objValueColumn.TagName = "Process value archive\PDL_ZT_1"
objValueColumn.BackColor = RGB(255,255,255)
objValueColumn.TimeColumn = objTimeColumn.Name

'add new ValueColumn and assign propertys
Set objValueColumn = objTableControl.GetValueColumnCollection.AddItem("myValueTable2")
objValueColumn.Provider = 1
objValueColumn.TagName = "Process value archive\PDL_ZT_2"
objValueColumn.BackColor = RGB(0,255,255)
objValueColumn.TimeColumn = objTimeColumn.Name

'add new ValueColumn and assign propertys
Set objValueColumn = objTableControl.GetValueColumnCollection.AddItem("myValueTable3")
objValueColumn.Provider = 1
objValueColumn.TagName = "Process value archive\PDL_ZT_3"
objValueColumn.BackColor = RGB(255,255,0)
objValueColumn.TimeColumn = objTimeColumn.Name
```

Ergebnis



	myRefTimeAxis	myValueTable1	myValueTable2	myValueTable3
112	08.02.2010 14:15:03	1	1	84
113	08.02.2010 14:15:03	22	1	84
114	08.02.2010 14:15:04	2	1	84
115	08.02.2010 14:15:04	1	1	84
116	08.02.2010 14:15:05	1	1	84
117	08.02.2010 14:15:05	3	1	84
118	08.02.2010 14:15:06	1	1	84
119	08.02.2010 14:15:06	1	1	84
120	08.02.2010 14:15:07	18	1	84

Beispiel: Skripte zum WinCC AlarmControl

Einleitung

Die folgenden Beispiele zeigen die Verwendung von Skripten für das WinCC AlarmControl.

Voraussetzung

- Im Editor "Alarm Logging" haben Sie bereits Meldungen projektiert.

Beispiel1: Filter setzen

Ein Filter mit der Meldungsnummer "2" wird gesetzt bzw. zurückgesetzt, wenn der Filter bereits gesetzt ist. Zusätzlich wird der Zustand im Diagnosefenster ausgegeben.

```
'VBS353
Dim objAlarmControl
'create reference to AlarmControl
Set objAlarmControl = ScreenItems("Control1")
'set / reset the filter and create a trace
If (objAlarmControl.MsgFilterSQL = "") Then
 objAlarmControl.MsgFilterSQL = "MSGNR = 2"
 HMIRuntime.Trace "MsgFilterSQL set to MSGNR = 2" & vbNewLine
Else
 objAlarmControl.MsgFilterSQL = ""
 HMIRuntime.Trace "no filter" & vbNewLine
End If
```

Beispiel2: Spalte zum WinCC AlarmContol hinzufügen

Die Spalte "Meldetext" wird hinzugefügt bzw. entfernt, wenn die Spalte bereits vorhanden ist. Zusätzlich wird der Zustand im Diagnosefenster ausgegeben. Der Meldeblock der Spalte "Meldetext" hat den Objektnamen "Text1".

```
'VBS354
'add this function to the declaration section
Function IsExistingMsgColumn( objAlarmControl, strName )
'this function checks if the MessageColumn exists
on error resume next
objAlarmControl.GetMessageColumn( strName )
If err.number = 0 Then
 IsExistingMsgColumn = True
else
 err.Clear
 IsExistingMsgColumn = False
end if
End Function

'example code
Dim objAlarmControl
Dim colMsgColumn
'create reference to the alarm control
Set objAlarmControl = ScreenItems("Control1")
Set colMsgColumn = objAlarmControl.GetMessageColumnCollection
'add or remove the MsgColumn
If ( IsExistingMsgColumn(objAlarmControl, "Text1") ) Then
 HMIRuntime.Trace "Remove MsgColumn" & vbNewLine
```

```
 colMsgColumn.RemoveItem("Text1")
Else
 HMIRuntime.Trace "Add MsgColumn" & vbNewLine
 colMsgColumn.AddItem("Text1")
End If
```

Beispiel3: Inhalt des Meldefensters im Diagnosefenster ausgeben

```
'VBS355
Dim objAlarmControl
Dim lIndex
Dim lCellIndex
'create reference to the alarm control
Set objAlarmControl = ScreenItems("Control1")
'enumerate and trace out row numbers
For lIndex = 1 To objAlarmControl.GetRowCollection.Count
 HMIRuntime.trace "Row: " & (objAlarmControl.GetRow(lIndex).RowNumber) & "  "
 'enumerate and trace out column titles and cell texts
 For lCellIndex = 1 To objAlarmControl.GetRow(lIndex).CellCount
  HMIRuntime.trace objAlarmControl.GetMessageColumn(lCellIndex -1).Name & " "
  HMIRuntime.trace objAlarmControl.GetRow(lIndex).CellText(lCellIndex) & "  "
 Next
 HMIRuntime.trace vbNewLine
Next
```

1.15.3 Beispiele allgemein

1.15.3.1 Allgemeine Beispiele zu VBScript

Einleitung

Sie finden hier Beispiele zur allgemeinen Verwendung von VBScript zu folgenden Themen:

- Datenanbindung mit VBS programmieren
- Methoden aufrufen
- MS-Automation Schnittstelle nutzen
- Fremdapplikation starten

Hinweis

Sie können alle Objekte, die mit dem Windows Script Host (WSH) von Microsoft ausgeliefert werden, in ihre Umgebung über die Standard-VBS-Methode "CreateObject" einbinden. Sie haben jedoch mit VBS aus WinCC heraus keinen direkten Zugriff auf das WSH-Objekt selbst.

Beispiel 1: "FileSystemObject"-Objekt zum Arbeiten mit dem Dateisystem

```
Dim fso, MyFile
Set fso = CreateObject("Scripting.FileSystemObject")
Set MyFile = fso.CreateTextFile("c:\testfile.txt", True)
MyFile.WriteLine("This is a test.")
MyFile.Close
```

Beispiel 2: "WScript.Shell"-Objekt zum Arbeiten mit der Windows-Umgebung

Siehe auch

Beispiel: So projektieren Sie eine Datenbankanbindung mit VBS (Seite 824)

Beispiel: So starten Sie eine Fremdapplikation (Seite 828)

Beispiel: So nutzen Sie die MS Automation Schnittstelle (Seite 826)

Beispiel: So rufen Sie Methoden eines ActiveX-Controls auf (Seite 810)

1.15.3.2 Beispiel: So projektieren Sie eine Datenbankanbindung mit VBS

Einleitung

Die folgenden Beispiele beschreiben die Projektierung einer Access-Datenbankanbindung über einen ODBC-Treiber.

- Beispiel 1 schreibt einen Variablenwert aus WinCC in eine Access-Datenbank.
- Beispiel 2 liest einen Wert aus der Datenbank und schreibt diesen Wert in eine WinCC Variable.

Die Beispiele enthalten keine Fehlerbehandlung.

Vorgehensweise Beispiel 1

1. Access-Datenbank mit Tabelle WINCC_DATA und Spalten (ID, TagValue) mit der ID als Autowert erstellen.
2. ODBC-Datenquelle mit Namen "SampleDSN" einrichten, Verweis auf obige AccessDatenbank.
3. Programmierung.

Beispiel 1

```
'VBS108
Dim objConnection
Dim strConnectionString
Dim lngValue
Dim strSQL
Dim objCommand
strConnectionString = "Provider=MSDASQL;DSN=SampleDSN;UID=;PWD=;"
lngValue = HMIRuntime.Tags("Tag1").Read
strSQL = "INSERT INTO WINCC_DATA (TagValue) VALUES (" & lngValue & ");"
Set objConnection = CreateObject("ADODB.Connection")
objConnection.ConnectionString = strConnectionString
objConnection.Open
Set objCommand = CreateObject("ADODB.Command")
With objCommand
    .ActiveConnection = objConnection
    .CommandText = strSQL
End With
objCommand.Execute
Set objCommand = Nothing
objConnection.Close
Set objConnection = Nothing
```

Vorgehensweise Beispiel 2

1. WinCC Variable mit dem Namen dbValue anlegen.
2. Access-Datenbank mit Tabelle WINCC_DATA und den Spalten: ID, TagValue erstellen (ID als Autowert).
3. ODBC-Datenquelle mit Namen "SampleDSN" einrichten, Verweis auf obige AccessDatenbank.
4. Programmierung.

Beispiel 2

```
'VBS108a
Dim objConnection
Dim objCommand
Dim objRecordset
Dim strConnectionString
Dim strSQL
Dim lngValue
Dim lngCount
strConnectionString = "Provider=MSDASQL;DSN=SampleDSN;UID=;PWD=;"
strSQL = "select TagValue from WINCC_DATA where ID = 1"
Set objConnection = CreateObject("ADODB.Connection")
objConnection.ConnectionString = strConnectionString
objConnection.Open
Set objRecordset = CreateObject("ADODB.Recordset")
```

```
Set objCommand = CreateObject("ADODB.Command")
objCommand.ActiveConnection = objConnection
objCommand.CommandText = strSQL
Set objRecordset = objCommand.Execute
lngCount = objRecordset.Fields.Count
If (lngCount>0) Then
objRecordset.movefirst
lngValue = objRecordset.Fields(0).Value
HMIRuntime.Tags("dbValue").Write lngValue
Else
HMIRuntime.Trace "Selection returned no fields" & vbNewLine
End If
Set objCommand = Nothing
objConnection.Close
Set objRecordset = Nothing
Set objConnection = Nothing
```

Es gibt unterschiedliche Möglichkeiten, den ConnectionString für die Verbindung zu definieren, abhängig vom verwendeten Provider:

Microsoft OLE DB provider for ODBC

Ermöglicht Verbindungen zu einer beliebigen ODBC Datenquelle. Die entsprechende Syntax ist:

```
"[Provider=MSDASQL;]{DSN=name|FileDSN=filename};
[DATABASE=database;]UID=user; PWD=password"
```

Andere Microsoft OLE DB Provider (z.B. MS Jet, MS SQL Server)

Sie können ohne DSN arbeiten. Die entsprechende Syntax ist:

```
"[Provider=provider;]DRIVER=driver; SERVER=server;
DATABASE=database; UID=user; PWD=password"
```

Siehe auch

Allgemeine Beispiele zu VBScript (Seite 823)

1.15.3.3 Beispiel: So nutzen Sie die MS Automation Schnittstelle

Einleitung

Die folgenden drei Beipiele zeigen Ihnen, wie Sie die MS Automation Schnittstelle nutzen.

Beipiel 1: MS Excel

Dieses Beispiel schreibt den Ausgabewert eines Eingabefeldes in eine Excel-Tabelle.

```
'VBS113
Dim objExcelApp
Set objExcelApp = CreateObject("Excel.Application")
objExcelApp.Visible = True
'
'ExcelExample.xls is to create before executing this procedure.
'Replace <path> with the real path of the file ExcelExample.xls.
objExcelApp.Workbooks.Open "<path>\ExcelExample.xls"
objExcelApp.Cells(4, 3).Value = ScreenItems("IOField1").OutputValue
objExcelApp.ActiveWorkbook.Save
objExcelApp.Workbooks.Close
objExcelApp.Quit
Set objExcelApp = Nothing
```

Beipiel 2: MS Access

Dieses Beispiel öffnet einen Report von MS Access.

```
'VBS114
Dim objAccessApp
Set objAccessApp = CreateObject("Access.Application")
objAccessApp.Visible = True
'
'DbSample.mdb and RPT_WINCC_DATA have to create before executing
'this procedure.
'Replace <path> with the real path of the database DbSample.mdb.
objAccessApp.OpenCurrentDatabase "<path>\DbSample.mdb", False
objAccessApp.DoCmd.OpenReport "RPT_WINCC_DATA", 2
objAccessApp.CloseCurrentDatabase
Set objAccessApp = Nothing
```

Beipiel 3: MS Internet Explorer

Dieses Beispiel öffnet den MS IE.

```
'VBS115
Dim objIE
Set objIE = CreateObject("InternetExplorer.Application")
objIE.Navigate "http://www.siemens.de"
Do
Loop While objIE.Busy
objIE.Resizable = True
objIE.Width = 500
objIE.Height = 500
objIE.Left = 0
```

```
objIE.Top = 0
objIE.Visible = True
```

Siehe auch

Allgemeine Beispiele zu VBScript (Seite 823)

1.15.3.4 Beispiel: So starten Sie eine Fremdapplikation

Einleitung

Die folgenden beiden Beispiele zeigen Ihnen, wie Sie eine Fremdapplikation starten.

Beispiel

```
'VBS117
Dim objWshShell
Set objWshShell = CreateObject("Wscript.Shell")
objWshShell.Run "Notepad Example.txt", 1
```

Siehe auch

Allgemeine Beispiele zu VBScript (Seite 823)

1.16 Grundlagen von VBScript

1.16.1 Grundlagen von VBScript

Einleitung

Im Folgenden finden Sie die wichtigsten Themen der Microsoft-VBScript Referenz:

- Das VBScript Sprachverzeichnis
- Das VBScript-Tutorium mit den wichtigsten Grundlagen
- Die Scripting Laufzeitreferenz

Wenn Sie eine vollständige Version der VBScript Referenz benötigen, finden Sie diese unter

http://msdn2.microsoft.com/en-us/library/t0aew7h6 (http://msdn2.microsoft.com/en-us/library/t0aew7h6)

Siehe auch

Microsoft VBScript Referenz (http://msdn2.microsoft.com/en-us/library/t0aew7h6)

1.16.2 VBScript Grundlagen

ANSI-C zum Erstellen von Funktionen und Aktionen 2

2.1 Erstellen von Funktionen und Aktionen mit ANSI-C

Inhalt

Hintergrundtätigkeiten, wie z.B. der tägliche Ausdruck eines Protokolls, die Überwachung von Variablen oder die Ausführung von bildunabhängigen Berechnungen werden in Runtime durch Aktionen ausgeführt.

Der Start dieser Aktionen wird durch Trigger gesteuert.

In Aktionen können Funktionen aufgerufen werden. WinCC stellt eine Vielzahl von Funktionen bereit, die zum Teil anwenderseits verändert werden können. Darüberhinaus können eigene Funktionen entwickelt werden.

Der Editor "Global Script" dient der Erstellung und Bearbeitung von Funktionen und Aktionen.

Dieses Kapitel zeigt Ihnen,

- wie Sie den Editor "Global Script" einsetzen
- wie Sie Funktionen erstellen und bearbeiten.
- wie Sie Aktionen erstellen und bearbeiten
- wie Sie Diagnosewerkzeuge einsetzen, um Laufzeitprobleme zu analysieren.

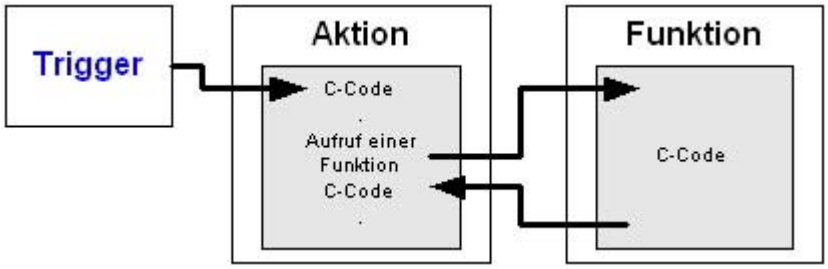
2.2 Erstellen von Funktionen und Aktionen

Einleitung

WinCC ermöglicht den Einsatz von Funktionen und Aktionen zur Dynamisierung der Abläufe in Ihrem WinCC Projekt. Diese Funktionen und Aktionen sind in der Sprache ANSI-C geschrieben.

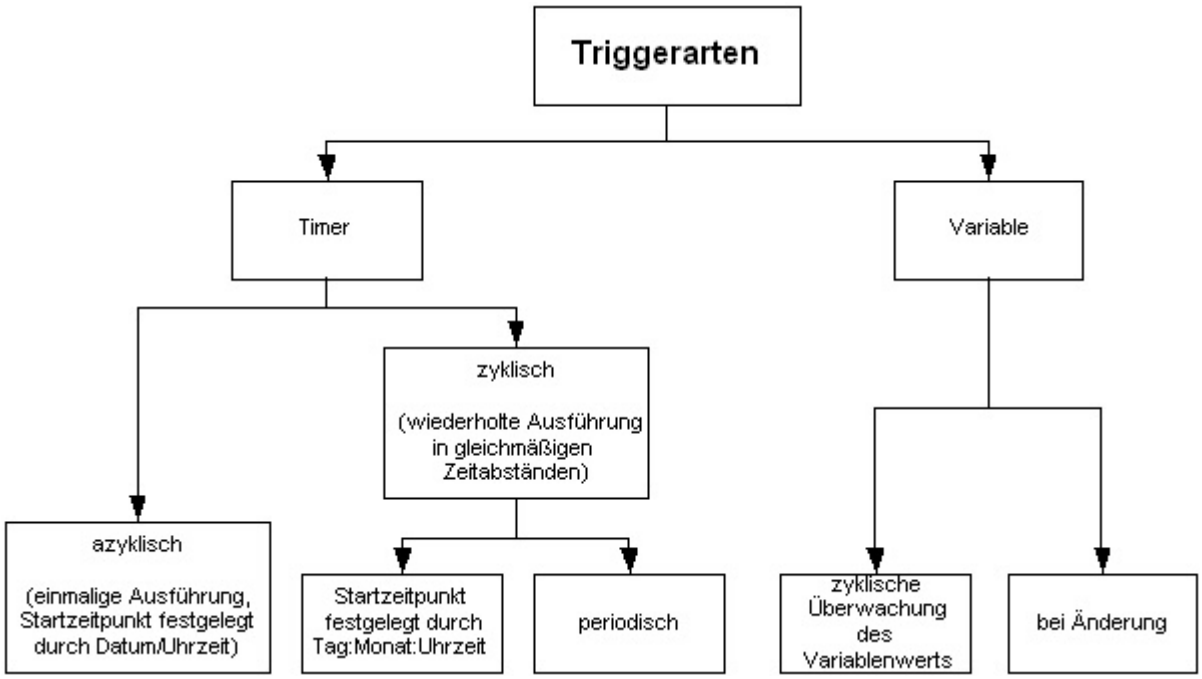
Abgrenzung Funktion - Aktion

Aktionen werden durch einen Trigger, also durch ein auslösendes Ereignis zum Ablauf gebracht. Funktionen besitzen keinen solchen Trigger und werden als Bestandteil von Aktionen sowie in Dynamik Dialogen, im Tag Logging und im Alarm Logging verwendet.



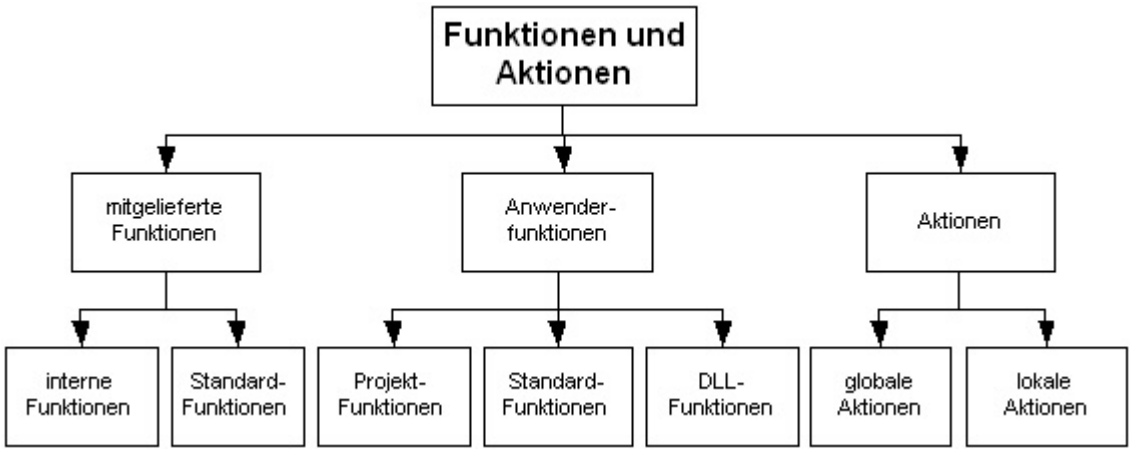
Triggerarten

Folgende Triggerarten stehen zur Verfügung:



Gliederung der Funktionen und Aktionen

Das Diagramm gibt eine Übersicht über die Funktionen- und Aktionenlandschaft:



Aktionen werden für bildunabhängige Hintergrundtätigkeiten eingesetzt, z.B. der tägliche Ausdruck eines Protokolls, die Überwachung von Variablen oder die Ausführung von Berechnungen.

Funktionen sind Codeteile, die an mehreren Stellen verwendet werden können, aber nur an einer Stelle definiert werden. WinCC stellt eine Vielzahl von Funktionen zur Verfügung. Darüber hinaus können Sie eigene Funktionen und Aktionen schreiben.

Mitgelieferte Standard-Funktionen können durch den Anwender verändert werden. Geänderte Standard-Funktionen werden bei Neuinstallation oder Hochrüstung von WinCC gelöscht bzw. durch die mitgelieferten Standardfunktionen ersetzt. Deshalb sollten sie vorher gesichert werden.

Entwurfswerkzeug

Für Entwurf und Bearbeitung von Funktionen und Aktionen stellt WinCC den Editor Global Script zur Verfügung. Gobal Script starten Sie aus dem Navigationsfenster des WinCC Explorer.

Siehe auch

Laufzeitverhalten von Aktionen (Seite 899)
Aktionen erstellen und bearbeiten (Seite 876)
Funktionen erstellen und bearbeiten (Seite 862)
Der Editor Global Script (Seite 847)
Verwendung von DLLs in Funktionen und Aktionen (Seite 845)
Verwendung globaler C-Variablen (Seite 843)
So nehmen Sie Global Script Runtime in die Anlaufliste des Projekts auf (Seite 841)
So generieren Sie den Header neu (Seite 857)
Merkmale von globalen Aktionen (Seite 840)
Merkmale von lokalen Aktionen (Seite 839)
Merkmale von internen Funktionen (Seite 838)
Merkmale von Standard-Funktionen (Seite 836)
Merkmale von Projekt-Funktionen (Seite 835)

2.3 Merkmale von Projekt-Funktionen

Merkmale von Projekt-Funktionen

Projekt-Funktionen ...

- werden von Ihnen selbst erstellt
- können von Ihnen geändert werden
- können gegen Änderungen und Einsicht durch ein Passwort geschützt werden
- besitzen keinen Trigger
- sind nur innerhalb des Projekts bekannt
- besitzen den Dateinamen "*.fct"

Projekt-Funktionen werden im Unterverzeichnis "\library" des WinCC Projektes abgelegt.

Verwendung von Projekt-Funktionen

Projekt-Funktionen können verwendet werden ...

- in anderen Projekt-Funktionen
- in Global Script Aktionen
- im Graphics Designer in C-Aktionen und innerhalb des Dynamik-Dialogs
- im Alarm Logging innerhalb der Loop in Alarm Funktionalität
- im Tag Logging beim Starten und Freigeben von Archiven und beim Auslagern von Umlaufarchiven

Siehe auch

So schützen Sie eine Funktion gegen Änderungen und Einsicht (Seite 871)

Funktionen erstellen und bearbeiten (Seite 862)

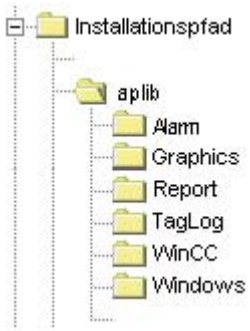
2.4 Merkmale von Standard-Funktionen

Merkmale von Standard-Funktionen

Standard-Funktionen ...

- werden von WinCC zur Verfügung gestellt
- können von Ihnen auch selbst erstellt werden
- können von Ihnen geändert werden
- können gegen Änderungen und Einsicht durch ein Passwort geschützt werden
- besitzen keinen Trigger
- sind projektübergreifend bekannt
- besitzen den Dateinamen "*.fct"

Standard-Funktionen werden in den Unterverzeichnissen von "\aplib" im Installationsverzeichnisses von WinCC abgelegt.



Verwendung von Standard-Funktionen

Standard-Funktionen können verwendet werden ...

- in Projekt-Funktionen
- in anderen Standard-Funktionen
- in Global Script Aktionen
- im Graphics Designer in C-Aktionen und innerhalb des Dynamik-Dialogs
- im Alarm Logging innerhalb der Loop in Alarm Funktionalität
- im Tag Logging beim Starten und Freigeben von Archiven und beim Auslagern von Umlaufarchiven

Hinweis

Mitgelieferte Standardfunktionen können durch den Anwender verändert werden. Geänderte Standard-Funktionen werden bei Neuinstallation oder Hochrüstung von WinCC gelöscht bzw. durch die mitgelieferten Standardfunktionen ersetzt. Deshalb sollten sie vorher gesichert werden.

Siehe auch

So verwenden Sie Standard- und Projekt-Funktionen (Seite 868)

Funktionen erstellen und bearbeiten (Seite 862)

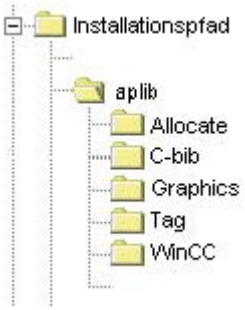
2.5 Merkmale von internen Funktionen

Merkmale von internen Funktionen

Interne Funktionen ...

- werden von WinCC zur Verfügung gestellt
- können von Ihnen **nicht** selbst erstellt werden
- können von Ihnen **nicht** geändert werden
- können **nicht** umbenannt werden
- besitzen keinen Trigger
- sind projektübergreifend bekannt
- besitzen den Dateinamen "*.icf"

Interne Funktionen werden in den Unterverzeichnissen von "\aplib" im Installationsverzeichnisses von WinCC abgelegt.



Verwendung von internen Funktionen

Interne Funktionen können verwendet werden ...

- in Projekt-Funktionen
- in Standard-Funktionen
- in Aktionen
- im Graphics Designer in C-Aktionen und innerhalb des Dynamik-Dialogs

2.6 Merkmale von lokalen Aktionen

Merkmale von lokalen Aktionen

Lokale Aktionen ...

- werden von Ihnen selbst erstellt
- können von Ihnen geändert werden
- können gegen Änderungen und Einsicht durch ein Passwort geschützt werden
- besitzen mindestens einen Trigger
- werden nur auf dem zugeordneten Rechner ausgeführt
- besitzen den Dateinamen "*.pas"

Lokale Aktionen werden im Unterverzeichnis "\<Rechnername>\Pas" des Projektverzeichnisses abgelegt.

Verwendung von lokalen Aktionen

Aktionen werden für bildunabhängige Hintergrundtätigkeiten eingesetzt, wie z.B. der tägliche Ausdruck eines Protokolls, die Überwachung von Variablen oder die Ausführung von Berechnungen. Die Ausführung der Aktion wird durch den projektierten Trigger gestartet. Damit die Aktion ausgeführt werden kann, ist es notwendig, Global Script Runtime in die Anlaufliste aufzunehmen.

Lokale Aktionen können im Gegensatz zu globalen Aktionen einem Rechner zugeordnet werden. Damit wird erreicht, dass beispielsweise ein Protokoll nur auf dem Server ausgedruckt wird.

Siehe auch

So schützen Sie eine Aktion gegen Änderungen und Einsicht (Seite 883)

Trigger (Seite 886)

Aktionen erstellen und bearbeiten (Seite 876)

So nehmen Sie Global Script Runtime in die Anlaufliste des Projekts auf (Seite 841)

2.7 Merkmale von globalen Aktionen

Merkmale von globalen Aktionen

Globale Aktionen ...

- werden von Ihnen selbst erstellt
- können von Ihnen geändert werden
- können gegen Änderungen und Einsicht durch ein Passwort geschützt werden
- besitzen mindestens einen Trigger, der die Ausführung startet
- werden in einem Client-Server Projekt auf allen Rechnern des Projekts ausgeführt
- besitzen den Dateinamen "*.pas"

Globale Aktionen werden im Unterverzeichnis "\Pas" des WinCC Projektes abgelegt.

Verwendung von globalen Aktionen

Aktionen werden für Hintergrundtätigkeiten eingesetzt, wie z.B. der tägliche Ausdruck eines Protokolls, die Überwachung von Variablen oder die Ausführung von Berechnungen. Die Ausführung der Aktion wird durch den projektierten Trigger gestartet. Damit die Aktion ausgeführt werden kann, ist es notwendig, Global Script Runtime in die Anlaufliste aufzunehmen.

Globale Aktionen werden im Gegensatz zu lokalen Aktionen in einem Client-Server Projekt auf allen Rechnern des Projekts ausgeführt. Bei einem Einzelplatzprojekt gibt es keinen Unterschied zwischen globalen und lokalen Aktionen.

Siehe auch

So schützen Sie eine Aktion gegen Änderungen und Einsicht (Seite 883)

Trigger (Seite 886)

Aktionen erstellen und bearbeiten (Seite 876)

So nehmen Sie Global Script Runtime in die Anlaufliste des Projekts auf (Seite 841)

2.8 So nehmen Sie Global Script Runtime in die Anlaufliste des Projekts auf

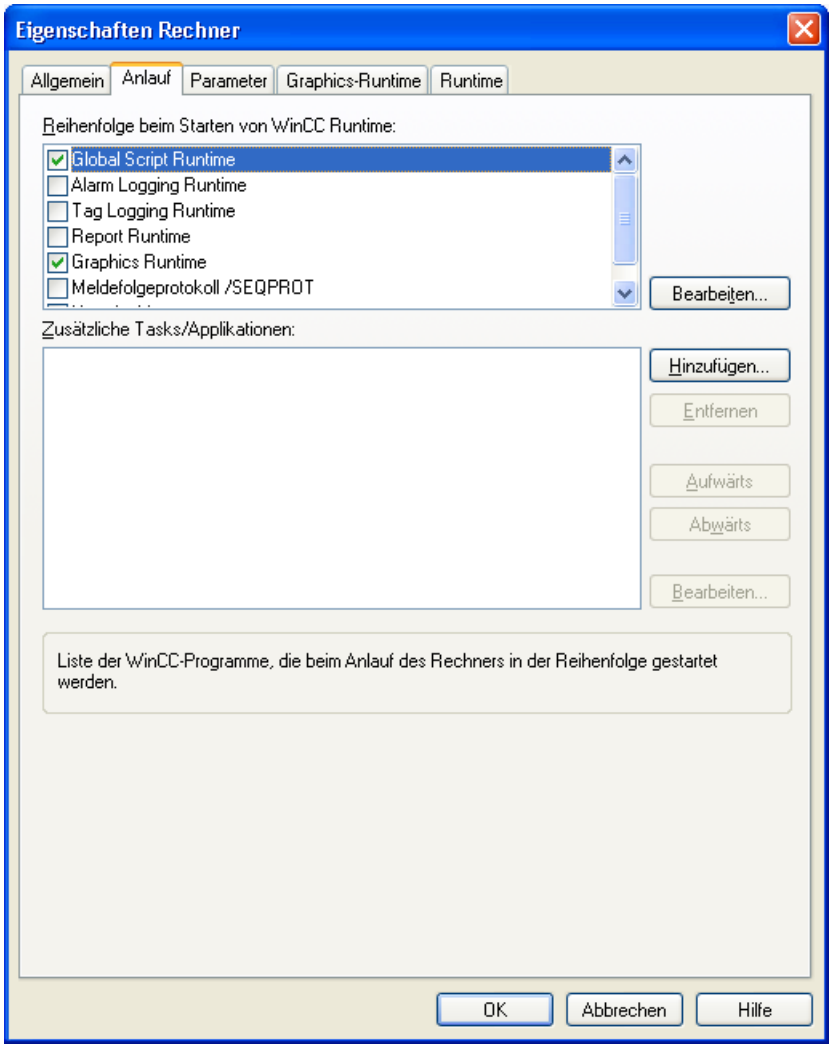
Einleitung

Damit Global Script Aktionen in Runtime ausgeführt werden, ist Global Script Runtime in die Anlaufliste des Projekts aufzunehmen. Auf die Ablauffähigkeit von Funktionen hat das keinen Einfluss.

Vorgehensweise

1. Wählen Sie im Kontextmenü des Rechners im WinCC Explorer den Eintrag "Eigenschaften". Der Dialog "Eigenschaften Rechnerliste" öffnet sich.
2. Klicken Sie auf die Schaltfläche "Eigenschaften". Der Dialog "Eigenschaften Rechner" öffnet sich.
3. Wählen Sie die Registerkarte "Anlauf".

4. Aktivieren Sie Global Script Runtime.



5. Schließen Sie das Dialogfenster mit "OK"

2.9 Verwendung globaler C-Variablen

Definition globaler C-Variablen

Eine globale C-Variable definieren Sie, indem Sie die Definitionszeile vor den Funktionsnamen einer Funktion einfügen:

```
int a; //Die Variable a wird als integer definiert
void dummy() //Funktionsname
{
. //Funktionscode
}
```

Gültigkeitsbereich

Jede so definierte Variable ist im Runtime in allen Funktionen und Aktionen bekannt. Sie wird bereits beim Start von Runtime angelegt, auch wenn die Funktion selbst nicht aufgerufen wurde.

Hinweis

Wenn Sie den WinCC ServiceMode betreiben, besteht kein gemeinsamer Datenbereich für C-Scripting. Daher können keine globalen C-Variablen zwischen "Global Script" und dem "Graphics Designer" ausgetauscht werden.

Verwendung globaler C-Variablen

Globale C-Variablen verwenden Sie in Funktionen oder Aktionen, indem Sie sie innerhalb der Funktion oder Aktion als extern deklarieren:

```
void dummy() //Funktionsname
{
extern int a; //externe Deklaration der Variablen a
. //Funktionscode
}
```

Dem Compiler wird damit gesagt, dass er die Variable a nicht selber anlegen muss, sondern dass diese in Runtime an anderer Stelle angelegt wird.

Wird nun der Wert der Variablen a verändert, dann kann diese Änderung in allen Funktionen und Aktionen ausgelesen werden.

Jede C-Variable darf nur an einer Stelle definiert werden. Aus Gründen der Übersichtlichkeit und zur Vermeidung von Doppeldefinitionen empfiehlt es sich deshalb, globale C-Variablen nur an einer Stelle zu definieren.

Hinweis

Einer Funktion und den mit ihr definierten globalen C-Variablen stehen maximal 64 KByte Speicherplatz zur Verfügung.

2.10 Verwendung von DLLs in Funktionen und Aktionen

DLLs anpassen

WinCC bietet die Möglichkeit, eigene DLLs (Dynamic Link Libraries) zu verwenden.

Der Funktionsumfang vorhandener DLLs kann für Funktionen und Aktionen nutzbar gemacht werden, indem die betreffende Funktion oder Aktion ergänzt wird.

An den Anfang der Funktion oder Aktion ist folgender Code einzufügen:

```
#pragma code("<Name>.dll")
<Typ des Returnwerts> <Funktionsname 1>(...);
<Typ des Returnwerts> <Funktionsname 2>(...);
  .
  .
  .
<Typ des Returnwerts> <Funktionsname n>(...);
#pragma code()
```

Damit sind die Funktionen <Funktionsname 1> ... <Funktionsname n> aus <Name.dll> deklariert und können in der betreffenden Funktion oder Aktion aufgerufen werden.

Beispiel:

```
#pragma code("kernel32.dll")
VOID GetLocalTime(LPSYSTEMTIME lpSystemTime);
#pragma code()

SYSTEMTIME st;

GetLocalTime(&st);
```

Alternativ zum dargestellten Verfahren, kann auch die Header-Datei Apdefap.h ergänzt werden.

Beim Einsatz eigener DLLs in WinCC ist die Release Version zu verwenden. WinCC wird als Release Version ausgeliefert, benutzt somit auch die Release Version der System-DLLs. Generiert man nun eine selbsterstellte DLL in der Debug Version, so kann es vorkommen, dass DLLs sowohl als Release- als auch als Debug Version geladen werden. Dies hat erhöhten Speicherbedarf zur Folge.

Strukturen der DLL müssen mit 1-Byte-Alignment angelegt werden

Hinweis

Die DLL muss entweder im bin-Verzeichnis oder in einem Pfad liegen, der in der Systemvariablen PATH definiert ist. Diese Variable wird über die Systemeigenschaften des Betriebssystems festgelegt.

2.11 Der Editor Global Script

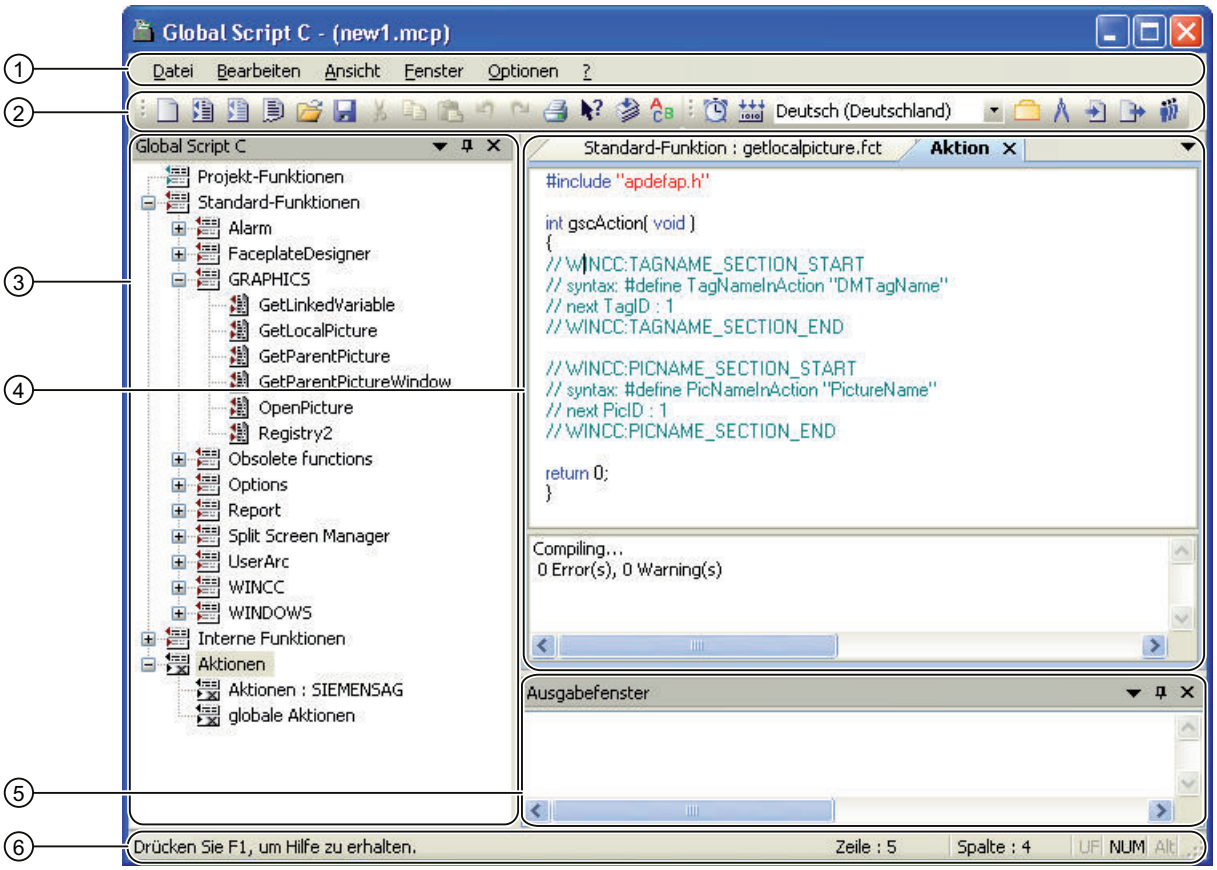
2.11.1 Der Editor Global Script

Einleitung

Für Erstellung und Bearbeitung von Funktionen und Aktionen stellt WinCC den Editor Global Script zur Verfügung. Global Script starten Sie aus dem Projektfenster des WinCC Explorer.

Aufbau des Editors Global Script

Der Editor Global Script ist nach Windows Standards aufgebaut. Er verfügt über Symbolleisten, Menüleiste und eine Statuszeile. Er enthält mehrere Fenster, die Pull-Down Menüs besitzen.



Die Menüleiste (1)
Der Inhalt der Menüleiste ist situationsabhängig. Sie ist immer sichtbar.

Die Symbolleisten (2)
Global Script enthält zwei Symbolleisten. Sie können bei Bedarf sichtbar geschaltet und mit

der Maus an jede beliebige Stelle des Bildschirms gesetzt werden.
Sie Symbolleisten sind über den Menübefehl "Ansicht" > "Symbolleisten" ein-und ausblendbar und können an jede beliebige Position innerhalb des Editors verschoben werden.

Das Navigationsfenster (3)
Das Navigationsfenster dient der Auswahl von Funktionen und Aktionen, um sie zu bearbeiten oder in ein Editierfenster an der Position der Schreibmarke einzufügen.Die Funktionen und Aktionen sind in hierarchisch geordneten Gruppen zusammengefasst.Funktionen werden mit ihrem Funktionsnamen, Aktionen mit ihrem Dateinamen angezeigt.

Das Editierfenster (4)
Im Editierfenster werden Funktionen und Aktionen geschrieben und bearbeitet. Es ist nur sichtbar, wenn eine Funktion oder Aktion zur Bearbeitung geöffnet ist. Jede Funktion oder Aktion wird in einem eigenen Editierfenster geöffnet. Es können mehrere Editierregister gleichzeitig geöffnet sein.

Das Ausgabefenster (5)
Im Ausgabefenster wird das Ergebnis der Funktionen "Suche in Dateien" oder "Alle Funktionen übersetzen" angezeigt. Es ist standardmäßig sichtbar und kann bei Bedarf unsichtbar geschaltet werden.

- Suche in Dateien:
  Das Ergebnis der Suche enthält pro gefundenem Suchbegriff eine Zeile im Ausgabefenster bestehend aus Zeilennummer, Pfad und Dateinamen sowie Text der Zeile mit der angegebenen Zeilennummer, in der der Suchbegriff gefunden wurde.Mit einem Doppelklick auf eine Anzeige im Ausgabefenster, können Sie die zugehörige Datei direkt öffnen. Die Schreibmarke wird in der Zeile platziert, in der sich der gesuchte Begriff befindet.
- Alle Funktionen übersetzen:
  Zu jeder übersetzten Funktion werden, falls vorhanden, Warnungen und Fehlermeldungen des Compilers ausgegeben. In der darauf folgenden Zeile wird der Pfad und der Dateiname der übersetzten Funktion sowie die zusammenfassende Meldung des Compilers angezeigt.

Die Statuszeile (6)
Die Statuszeile befindet sich am unteren Rand des Global Script-Fensters und lässt sich ein- und ausblenden. Sie enthält Informationen über die Position der Schreibmarke im Editierfenster und die Einstellung der Tastatur. Darüber hinaus wird eine Kurzinformation zur momentan gewählten Global Script-Funktionalität angezeigt oder ein Tipp gegeben.

Fenster-Docking

Durch Fenster-Docking haben Sie die Möglichkeit die einzelnen Fenster flexibel anzuordnen. Es steht Ihnen frei die Fenster umpositionieren, freistehend als alleinstehendes Fenster abzulegen oder Fenster in Tabgruppen zu gruppieren. Platzieren Sie beispielsweise Ihre Aktionen nebeneinander, untereinander oder als Tabs. Sie können Fenster automatisch verstecken und bei Bedarf wieder einblenden
Weitere Informationen erhalten Sie im Kapitel "Prozessbilder erstellen".

Siehe auch

Funktionen und Aktionen drucken (Seite 860)
So suchen Sie in Dateien (Seite 859)
So übersetzen Sie alle Funktionen (Seite 858)
So generieren Sie den Header neu (Seite 857)
So löschen Sie Aktionen oder Projekt- und Standard-Funktionen (Seite 857)
So verwenden Sie "Speichern unter..." (Seite 856)
So stellen Sie den Schriftstil ein (Seite 855)
So stellen Sie verschiedene Ansichten ein (Seite 855)
Arbeiten mit den Symbolleisten (Seite 852)
Arbeiten im Editierfenster (Seite 849)

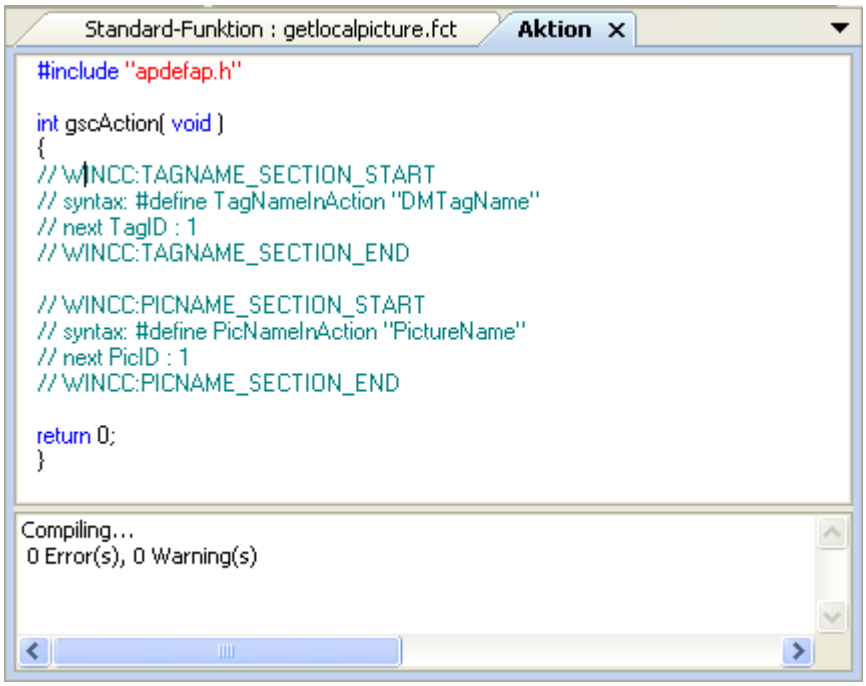
2.11.2 Arbeiten im Editierfenster

2.11.2.1 Arbeiten im Editierfenster

Einleitung

Das Editierfenster stellt Editierfunktionen zur Verfügung, die entweder mit der Tastatur oder mit der Maus ausgeführt werden.

Im Editierfenster werden Funktionen und Aktionen bearbeitet.



Inhalt

Das Fenster ist teilbar. Im oberen Teil des Editierfensters wird der Code der Funktion oder Aktion angezeigt. Im unteren Teil sehen Sie die Meldungen, die der Compiler während der Übersetzung der Funktion oder Aktion ausgibt.

Eigenschaften

Wird das Fenster zum ersten Mal geöffnet, dann ist der untere Teil des Fensters minimiert. Beim Start eines Übersetzungslaufs wird dieser Teil vergrößert, sodass die Meldungen des Compilers angezeigt werden können. Die Teilung des Fensters kann mit der Maus eingestellt werden. Mit Doppelklick auf eine Fehlermeldung wird die entsprechende Zeile im Code angesprungen.

Farbcodierung

Die Darstellung des C-Code ist wie folgt farblich gekennzeichnet:

Farbe	Bedeutung	Beispiel
blau	Schlüsselworte	define, double, if
grün	Kommentare	// das ist ein Kommentar
rot	strings	"Rectangle3"
schwarz	sonstiger C-Code	level=100*newvalue/255;

Hinweis

Eine Funktion oder Aktion kann nicht mehr als 32767 Zeichen enthalten, Leerzeichen inbegriffen.

Siehe auch

Editierfunktionen mit der Maus (Seite 852)

Editierfunktionen mit der Tastatur (Seite 851)

2.11.2.2 Editierfunktionen mit der Tastatur

Mit der Tastatur können Sie folgende Editierfunktionen ausführen:

Editierfunktion	Tastenbedienung
Schreibmodus Einfügen/Überschreiben umschalten	<Einfg>
neue Zeile einfügen	<Return>
ein Zeichen löschen rechts	<Entf>
ein Zeichen löschen links	<Backspace>
markierten Text löschen	<Entf> oder <Backspace>
Sprung an den Anfang der Zeile	<Pos1>
Sprung an das Ende der Zeile	<Ende>
Sprung an den Textanfang	<Strg+Pos1>
Sprung an das Textende	<Strg+Ende>
Schreibmarke bewegen	<Cursortasten>
Schreibmarke um einen Fensterinhalt zum Textanfang bewegen	<Bild Auf>
Schreibmarke um einen Fensterinhalt zum Textende bewegen	<Bild Ab>
Schreibmarke in die erste Zeile im Fenster setzen	<Strg+Bild Auf>
Schreibmarke in die letzte Zeile im Fenster setzen	<Strg+Bild Ab>
zur nächsten Tabulatorposition links springen	<Tab>
markierten Text ausschneiden und in die Zwischenablage kopieren	<Strg+X>
markierten Text in die Zwischenablage kopieren	<Strg+C>
Text aus der Zwischenablage einfügen	<Strg+V>

2.11.2.3 Editierfunktionen mit der Maus

Mit der Maus können Sie folgende Editierfunktionen ausführen:

Editierfunktion	Mausaktion (linke Maustaste)
Text markieren	den Text mit gedrückter Maustaste überstreichen
ein Wort markieren	Doppelklick auf das Wort
eine Zeile markieren	Dreifachklick auf die Zeile
erweitertes Markieren	<Shift> + Klick
Schreibmarke setzen	Klick
markierten Text verschieben	mit gedrückter Maustaste ziehen
markierten Text duplizieren	<Strg> + gedrückter Maustaste ziehen

Sonstige Editierfunktionen:

- Mit Doppelklick auf eine Fehlermeldung des Compilers springt die entsprechende Zeile im Code an.
- Klick mit der rechten Maustaste öffnet das Kontextmenü

Bei folgenden Aktionen wird markierter Text durch das Ergebnis der Aktion ersetzt:

- Zeichen mit der Tastatur eingeben
- Inhalt der Zwischenablage einfügen
- Aufruf einer Funktion über die Parameterversorgung einfügen

2.11.3 Arbeiten mit den Symbolleisten

Zweck

Die Symbolleisten befinden sich in ihrer Standardposition unterhalb der Menüleiste, am oberen Rand des Global Script-Fensters. Die auf den Symbolleisten angeordneten Tasten erlauben einen schnellen und bequemen Zugriff auf die von Global Script angebotene Funktionalität.

Es stehen folgende zwei Symbolleisten zur Verfügung:

Symbolleiste "Standard"

Symbolleiste "Bearbeiten"

Inhalt

Die Symbolleiste "Standard" enthält Schaltflächen mit folgenden Funktionen:

Schaltfläche	Funktion	Tastenkombination
	Legt eine neue Aktion an.	<Alt+A> oder <Strg +N>
	Legt eine neue Standard-Funktion an.	<Alt+S> oder <Strg +N>
	Legt eine neue Projekt-Funktion an.	<Alt+P> oder <Strg +N>
	Legt eine neue Header Datei an.	<Strg+H>
	Öffnet eine vorhandene Aktion oder Funktion.	<Strg+O>
	Speichert den Inhalt des aktiven Editierfensters. Die Funktion ist nur dann verfügbar, wenn ein Editierfenster geöffnet ist.	<Strg+S>
	Schneidet den markierten Text aus und kopiert ihn in die Zwischenablage. Die Funktion ist nur dann verfügbar, wenn Text markiert ist.	<Strg+X>
	Kopiert den markierten Text in die Zwischenablage. Die Funktion ist nur dann verfügbar, wenn Text markiert ist.	<Strg+C>
	Fügt den Inhalt der Zwischenablage an die Position der Schreibmarke ein. Die Funktion ist nur dann verfügbar, wenn die Zwischenablage nicht leer ist.	<Strg+V>
	Macht die letzte von maximal 30 Editieraktionen rückgängig. Die Funktion ist nur dann verfügbar, wenn eine Editieraktion ausgeführt wurde.	<Strg+Z>
	Stellt die letzte rückgängig gemachte Editieraktionen wieder her. Die Funktion ist nur dann verfügbar, wenn eine Editieraktion rückgängig gemacht worden ist.	<Strg+A>
	Druckt den Inhalt des aktiven Editierfensters in Form einer Projektdokumentation aus. Die Funktion ist nur dann verfügbar, wenn ein Editierfenster geöffnet ist.	<Strg+P>
	Aktiviert die Direkthilfe.	<Shift+F1>
	Generiert den Header neu.	<Strg+G>
	Ermöglicht, den Schriftstil einzustellen.	<Strg+F>

Hinweis

Die Tastenkombination <Strg+N> ist nur dann verfügbar, wenn mindestens ein Editierfenster geöffnet ist. Ist der Inhalt des aktiven Editierfensters eine Funktion, dann wird mit <Strg+N> eine neue Projekt-Funktion angelegt. Ist der Inhalt des aktiven Editierfensters eine Aktion, dann wird eine neue globale Aktion angelegt.

Die Symbolleiste "Bearbeiten" enthält Schaltflächen mit folgenden Funktionen:

Schaltfläche	Funktion	Tastenkombination
	Ermöglicht das Einfügen von Informationen zur Funktion und bei Aktionen zusätzlich das Einrichten eines Triggers. Die Funktion ist nur dann verfügbar, wenn ein Editierfenster geöffnet ist.	<Strg+I>
	Übersetzt den Code im aktiven Editierfenster. Die Funktion ist nur dann verfügbar, wenn ein Editierfenster geöffnet ist.	<Shift+F8>
	Stellt die passende Codepage ein. Beachten Sie, dass die Codepageauswahl mit dem Quelltext übereinstimmt. Im Quelltext kann jeweis nur eine Sprache verwendet werden.	-
	Öffnet den Variablendialog. Die Funktion ist nur dann verfügbar, wenn ein Editierfenster geöffnet ist.	<Strg+R>
	Öffnet einen Auswahldialog für Bilder. Der Name des gewählten Bildes wird im Editierfenster an der Stelle in den Code eingefügt, an der Sich die Schreibmarke befindet. Die Funktion ist nur dann verfügbar, wenn ein Editierfenster geöffnet ist.	<Strg+W>
	Importiert eine Aktion. Die Funktion ist nur dann verfügbar, wenn das aktive Fenster eine Aktion enthält.	<Strg+M>
	Exportiert die Aktion im aktiven Editierfenster. Die Funktion ist nur dann verfügbar, wenn das aktive Fenster eine Aktion enthält.	<Strg+T>
	Stellt die Bedienberechtigung für die Aktion ein. Die Funktion ist nur dann verfügbar, wenn das aktive Fenster eine Aktion enthält.	<Strg+E>

Eigenschaften

Beide Symbolleisten lassen sich ein- und ausblenden.

Sie können unterhalb der Menüleiste angedockt werden.

Im ausgedockten Zustand können sie mit der Maus an jede Stelle des Bilschirms positioniert werden.

Siehe auch

So stellen Sie verschiedene Ansichten ein (Seite 855)

2.11.4 So stellen Sie verschiedene Ansichten ein

Einleitung

Ansichten sind in diesem Zusammenhang die unterschiedlichen Kombinationen der sichtbar geschalteten Elemente Ausgabefenster, Statuszeile und Symbolleisten im Global Script Editor. Die genannten Elemente können individuell ein- und ausgeblendet werden.

In der Standardeinstellung sind alle Elemente sichtbar.

Vorgehensweise

1. Öffnen Sie das Menü "Ansicht" in der Menüleiste von Global Script
2. Aktivieren oder deaktivieren Sie die Anzeige des gewünschten Elements z.B. der Symbolleiste. Wenn die Anzeige aktiviert ist, so wird dies durch ein Häkchen vor dem Namen gekennzeichnet.

Hinweis

Wird Global Script neu gestartet, dann gilt wieder die Standardeinstellung und es werden alle Elemente angezeigt.

2.11.5 So stellen Sie den Schriftstil ein

Einleitung

Der Schriftstil wird aus den Einstellungen "Schriftart" "Schriftschnitt" und "Grad" gebildet.

Der eingestellte Schriftstil ist in allen Editierfenstern wirksam.

Vorgehensweise

1. Betätigen Sie die Schaltfläche  in der Symbolleiste "Standard" und öffnen Sie den Dialog zum Einstellen des Schriftstils.
2. Nehmen Sie dort die gewünschten Einstellungen vor
3. Übernehmen Sie die Einstellungen indem Sie mit "OK" bestätigen

Alternative Bedienung

Den Dialog zum Einstellen des Schriftstils können Sie auch folgendermaßen öffnen:

Öffnen Sie das Menü "Optionen" in der Menüleiste von Global Script und wählen Sie den Eintrag "Font" oder verwenden Sie die entsprechende Tastenkombination.

Hinweis

Die getroffenen Einstellungen werden automatisch gespeichert und stehen auch nach einem Neustart von WinCC wieder zur Verfügung.

2.11.6 So verwenden Sie "Speichern unter..."

Einleitung

Wird eine Funktion oder eine Aktion neu erstellt, dann richtet Global Script eine entsprechende Datei in einem vorgegebenen Pfad mit einem Standard-Dateinamen ein, z.B. "neue_Funktion_1.fct" für Funktionen und "gsccs1.pas" für Aktionen. Da dieser Standard-Dateiname wenig aussagekräftig ist, kann mit "Speichern unter..." die Funktion oder die Aktion unter einem anderen Dateinamen gespeichert werden.

Mit "Speichern unter..." wird nur der Dateiname geändert, der Funktions- oder Aktionsname bleibt dabei unverändert.

Global Script erwartet, dass das Verzeichnis für die zu speichernde Funktion oder Aktion einem Projektverzeichnis entspricht. Trifft dies nicht zu, dann wird ein Hinweis ausgegeben, die Datei aber dennoch gespeichert.

Voraussetzung

"Speichern unter.." ist nur verfügbar, wenn mindestens ein Editierfenster geöffnet ist, wobei der Inhalt des aktiven Editierfensters gespeichert wird.

Vorgehensweise

1. Öffnen Sie das Menü "Datei" in der Menüleiste von Global Script
2. Wählen Sie den Eintrag "Speichern unter..."
3. Geben Sie den neuen Dateinamen ein
4. Schließen Sie das Dialogfenster mit der Schaltfläche "Speichern"

Siehe auch

So löschen Sie Aktionen oder Projekt- und Standard-Funktionen (Seite 857)

2.11.7 So löschen Sie Aktionen oder Projekt- und Standard-Funktionen

Einleitung

Aktionen oder Projekt- und Standard-Funktionen können im Verlauf der Projektierung oder in Runtime gelöscht werden. Global Script löscht den Eintrag im Navigationsfenster und die zugehörige Datei.

Wird eine gelöschte Funktion von einer Aktion aufgerufen, dann wird die Aktion am Aufruf der Funktion abgebrochen.

Ist zu diesem Zeitpunkt das Global Script Diagnosefenster geöffnet, so erfolgt eine Meldung. Zusätzlich wird der Abbruch der Aktion in der Diagnosedatei WinCC_Sys_xx.log (xx = lfd. Nummer) protokolliert. Diese Diagnosedatei befindet sich im Unterverzeichnis "Diagnose" des Installationsverzeichnisses von WinCC.

Vorgehensweise

1. Öffnen Sie das Kontextmenü der zu löschenden Funktion oder Aktion im Navigationsfenster von Global Script
2. Wählen Sie den Eintrag "Löschen"
3. Bestätigen Sie die Sicherheitsabfrage mit "Ja"

Alternative Bedienung

Statt das Kontextmenü zu benutzen, können Sie auch mit der Taste <Entf> die markierte Funktion oder Aktion löschen.

Hinweis

Wird eine Funktion gelöscht, dann wird zusätzlich auch der Eintrag in der betreffenden Header-Datei gelöscht.

2.11.8 So generieren Sie den Header neu

Einleitung

In folgenden Fällen muss der Header neu generiert werden:

- Sie haben Projekt-Funktionen aus einem fremden Projekt in Ihren Projektpfad in das Verzeichnis "library" kopiert.
- Sie haben Standard-Funktionen von einem anderen Rechner in das Verzeichnis "aplib" oder Unterverzeichnisse kopiert.

Durch das Neugenerieren des Headers werden die kopierten Funktionen in die entsprechenden Header-Dateien eingetragen. Danach können Sie die Funktionen in Ihrem Projekt verwenden.

Vorgehensweise

1. Klicken Sie auf die Schaltfläche  in der Symbolleiste "Standard".

Alternative Bedienung

Alternativ können Sie den Generiervorgang auch folgendermaßen anstoßen:

Öffnen Sie das Menü "Optionen" und wählen Sie den Eintrag "Header neu generieren" oder verwenden Sie die entsprechende Tastenkombination.

Hinweis

Der Inhalt des Navigationsfensters wird nach abgeschlossener Generierung aktualisiert.

Wenn WinCC in Runtime ist, wird das Runtime-System durch das Neugenerieren des Headers nicht beeinflusst.

2.11.9 So übersetzen Sie alle Funktionen

Einleitung

Wenn Sie manuell die Header-Dateien geändert haben, müssen Sie alle Funktionen neu übersetzen. Mit dem Menübefehl "Alle Funktionen übersetzen" werden automatisch alle Projektfunktionen, Standardfunktionen und interne Funktionen übersetzt.

Wenn Funktionen in anderen Funktionen aufgerufen werden, sind Fehlermeldungen möglich. Grund dafür ist, dass die aufgerufenen Funktionen zu diesem Zeitpunkt noch nicht übersetzt wurden. Diese Funktionen müssen Sie anschließend einzeln übersetzen.

Voraussetzung

Die Funktion ist nur verfügbar, wenn alle Editierfenster geschlossen sind.

Vorgehensweise

1. Öffnen Sie das Menü "Optionen"
2. Wählen Sie den Eintrag "Alle Funktionen übersetzen"

Alternative Bedienung

Sie können das Übersetzen aller Funktionen auch mit der Tastenkombination <Alt+U> starten.

Ergebnis

Die Ergebnisse der einzelnen Übersetzungsläufe werden im Ausgabefenster angezeigt, z.B. Warnungen und Fehlermeldungen des Compilers. Desweiteren werden der Pfad und

Dateiname der übersetzten Funktion sowie die zusammenfassende Meldung des Compilers angezeigt.

Hinweis

In einem Mehrplatz-Projekt ist die Funktion "Alle Funktionen übersetzen" nicht verfügbar. Die Zuordnung der Funktionen ist bei diesen Projekten nicht mehr möglich.

Auf einem WinCC-Rechner werden die so übersetzten Funktionen erst beim nächsten Start von Runtime wirksam.

2.11.10 So suchen Sie in Dateien

Einleitung

Es werden alle Dateien der im Navigationsfenster gewählten Gruppe nach dem angegebenen Suchstring durchsucht.

Das Ergebnis der Suche wird im Ausgabefenster wie folgt angezeigt:

Für jeden gefundenen Suchbegriff wird eine Zeile im Ausgabefenster angelegt. Sie enthält die Zeilennummer der Codezeile, in der der Suchbegriff gefunden wurde, Pfad und Dateinamen sowie die Codezeile selbst.

Standard- und Projekt-Funktionen sowie Aktionen können mit Doppelklick auf das Suchergebnis geöffnet werden. Die Schreibmarke ist an den Anfang der Zeile positioniert, die den Suchbegriff enthält. Bei internen Funktionen wird die Funktion, die den Suchbegriff enthält, im Navigationsfenster vorgelegt und markiert.

Vorgehensweise

1. Öffnen Sie das Kontextmenü der zu durchsuchenden Gruppe im Navigationsfenster von Global Script
2. Wählen Sie den Eintrag "Suche in Dateien"
3. Tragen Sie im Dialogfenster den zu suchenden Text ein
4. Starten Sie die Suche, indem Sie mit "Suchen" bestätigen. Das Ergebnis der Suche wird Ihnen im Ausgabefenster angezeigt.

2.11.11 Funktionen und Aktionen drucken

2.11.11.1 Funktionen und Aktionen drucken

Einleitung

Projekt-Funktionen, Standard-Funktionen und Aktionen können unter Verwendung festgelegter Systemlayouts gedruckt werden.

Voraussetzung ist, dass die zu druckende Funktion oder Aktion in einem Editerfenster angezeigt wird. Der Inhalt des aktiven Editierfensters wird gedruckt.

Der Ausdruck kann auch in der Seitenansicht auf dem Bildschirm dargestellt werden.

Zur Steuerung des Druckvorgangs stehen Druckparameter zur Verfügung.

Die verwendeten Systemlayouts sind:

- @gsc_pfc.rpl für Projekt-Funktionen
- @gsc_sfc.rpl für Standard-Funktionen
- @gsc_act.rpl für Aktionen

Siehe auch

So drucken Sie eine Projektdokumentation (Seite 861)

So öffnen Sie die Seitenansicht (Seite 861)

So stellen Sie die Druckparameter ein (Seite 860)

2.11.11.2 So stellen Sie die Druckparameter ein

Einleitung

Sie können den Ausdruck durch folgende Einstellungen beeinflussen:

- Angabe eines vom Standardlayout abweichenden Layouts
- Seitenbereich
- Wahl des Druckers
- Ausdruck in eine Datei umleiten

Voraussetzung

Mindestens ein Editierfenster muss geöffnet sein

Vorgehensweise

1. Öffnen Sie das Menü "Datei" in der Menüleiste von Global Script
2. Wählen Sie den Eintrag "Projektdokumentation einrichten..."
3. Nehmen Sie im geöffneten Dialogfenster die gewünschten Einstellungen vor
4. Übernehmen Sie die Einstellungen, indem Sie mit "OK" bestätigen

Hinweis

Die getroffenen Einstellungen werden automatisch gespeichert und stehen auch nach einem Neustart von WinCC wieder zur Verfügung.

2.11.11.3 So öffnen Sie die Seitenansicht

Einleitung

Bevor Sie den Ausdruck einer Funktion oder Aktion veranlassen, kann es vorteilhaft sein, sich den Ausdruck auf dem Bildschirm in der Seitenansicht anzeigen zu lassen.

Der Inhalt des aktiven Editierfensters wird in der Seitenansicht angezeigt.

Vorgehensweise

1. Öffnen Sie das Menü "Datei" in der Menüleiste
2. Wählen Sie den Eintrag "Projektdokumentation Ansicht"

2.11.11.4 So drucken Sie eine Projektdokumentation

Einleitung

Sie können sich den Inhalt des aktiven Editierfensters auf Drucker oder in eine Datei ausgeben lassen. Es werden dabei die eingestellten Druckparameter verwendet.

Vorgehensweise

1. Öffnen Sie das Menü "Datei" in der Menüleiste von Global Script
2. Wählen Sie den Eintrag "Projektdokumentation drucken"

2.12 Funktionen erstellen und bearbeiten

2.12.1 Funktionen erstellen und bearbeiten

Einleitung

Es wird zwischen Projekt-, Standard- und internen Funktionen unterschieden. Mit WinCC bekommen Sie bereits eine Vielzahl von fertigen Standard-Funktionen und internen Funktionen zur Verfügung gestellt. Darüber hinaus können Sie sich eigene Projekt- und Standard-Funktionen schreiben oder vorhandene Standard-Funktionen modifizieren. Vom System mitgelieferte Standard-Funktionen werden allerdings bei der nächsten Installation von WinCC überschrieben.

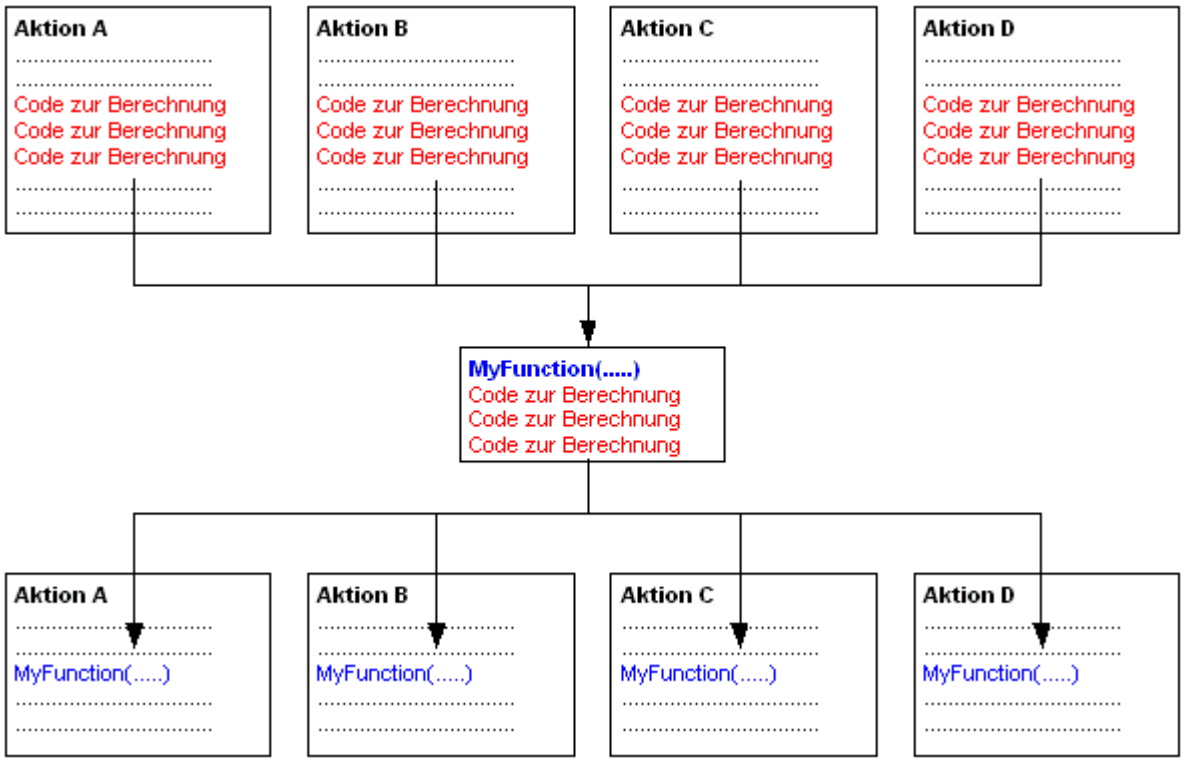
Interne Funktionen können weder erstellt noch modifiziert werden.

Verwendung von Funktionen

Wird beispielsweise ein und dieselbe Berechnung mit verschiedenen Ausgangswerten in mehreren Aktionen gebraucht, dann ist es vorteilhaft, diese Berechnung in einer Funktion zu programmieren. In den Aktionen wird dann nur noch die Funktion mit den aktuellen Parametern aufgerufen.

Damit sind folgende Vorteile verbunden:

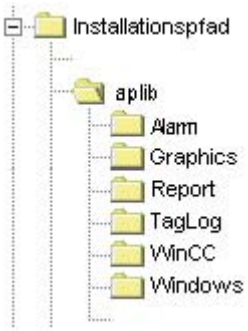
- Die Berechnung wird nur einmal programmiert
- Änderungen werden nur an einer Stelle gemacht, nämlich in der Funktion und nicht in jeder Aktion
- Der Aktionscode wird kürzer und bleibt übersichtlicher



Auffinden von Funktionen

Der Zugriff auf bestehende Funktionen oder die Erstellung neuer Funktionen erfolgt alternativ über das Navigationsfenster von Global Script, das Menü "Datei" oder mit der entsprechenden Schaltfläche auf der Symbolleiste.

Funktionen werden im Dateisystem wie folgt abgelegt:



Bearbeiten und Übersetzen von Funktionen

Eine Funktion wird in einem eigenen Editerfenster bearbeitet und übersetzt. Das Editierfenster enthält nach dem Übersetzungslauf Meldungen des Compilers. Das können Warnungen und/ oder Fehlermeldungen sein. In jedem Fall wird zusammenfassend die Anzahl der Warnungen und Fehlermeldungen ausgegeben.

Was geschieht, wenn ich den Funktionsnamen ändere?

Im Navigationsfenster werden Funktionen immer mit ihrem Funktionsnamen angezeigt, nicht mit dem Windows-Dateinamen. Wenn Sie im Editierfenster den Funktionsnamen ändern und einen Übersetzungslauf durchführen, dann stimmt der Name im Navigationsfenster nicht mehr mit dem Funktionsnamen überein. Das wird im Navigationsfenster durch das Zeichen "*" vor dem Namen gekennzeichnet. Sobald Sie die Funktion speichern, wird der aktuelle Funktionsname in das Navigationsfenster übernommen.

Dies ist beim Speichern von Funktionen zu beachten

Speichern Sie eine fehlerhaft übersetzte oder gar nicht übersetzte Funktion, dann erhält sie im Navigationsfenster das Symbol .

Speichern Sie eine fehlerfrei übersetzte Funktion, dann erhält sie im Navigationsfenster das Symbol ..

Siehe auch

Arbeiten im Editierfenster (Seite 849)
So verwenden Sie Funktionen aus anderen Quellen (Seite 875)
So benennen Sie eine Funktion um (Seite 874)
So übersetzen und speichern Sie eine Funktion (Seite 872)
So schützen Sie eine Funktion gegen Änderungen und Einsicht (Seite 871)
So fügen Sie funktionsbegleitende Informationen hinzu (Seite 869)
So verwenden Sie Standard- und Projekt-Funktionen (Seite 868)
So verwenden Sie interne Funktionen (Seite 867)
So schreiben Sie Funktionscode (Seite 866)
So legen Sie eine neue Funktion an (Seite 865)
Merkmale von Standard-Funktionen (Seite 836)
Merkmale von Projekt-Funktionen (Seite 835)

2.12.2 So legen Sie eine neue Funktion an

Einleitung

Die Vorgehensweise für Projekt- und Standard-Funktionen ist identisch. Im Navigationsfenster legen Sie den Typ (Projekt- oder Standard-Funktion) und, bei Standard-Funktionen, die Gruppe, z.B. "graphics" fest. Damit ist auch der Ablageort der Datei festgelegt.

Global Script schlägt für die neue Funktion einen Standardnamen, z.B. "neue_Funktion_3", vor. Das ist auch gleichzeitig der Dateiname. Damit die Funktionsnamen eindeutig sind enthält der vorgeschlagene Name eine laufende Nummer.

Den Standardnamen wird man in der Regel durch einen aussagekräftigen Funktionsnamen ersetzen. Beim erstmaligen Speichern der umbenannten Funktion kann der Dateiname geändert werden.

Global Script versieht die Funktion beim Anlegen mit den funktionsbegleitenden Informationen Erstellungsdatum, Änderungsdatum und Versionsangabe. Diese Informationen können im Dialog "Eigenschaften" angezeigt werden. Darüber hinaus kann in diesem Dialog die Funktion durch ein Passwort gegen Änderungen und Einsicht geschützt werden. Der Dialog wird mit der Schaltfläche geöffnet.

Hinweis

Als Funktionsnamen werden die Zeichen unterstützt, die auch von ANSI-C unterstützt werden:

- Buchstaben, mit Ausnahme der nationalen Sonderzeichen
- Ziffern
- Unterstrich

Vorgehensweise

1. Öffnen Sie im Navigationsfenster das Kontextmenü der gewünschten Gruppe
2. Wählen Sie "Neu"
   Ist eine neue Funktion angelegt worden, so enthält das zugehörige Editierfenster als erste Codezeile den Typ des Rückgabewerts und den Standardnamen der neuen Funktion. In der nachfolgenden Klammer können gegebenenfalls Übergabeparameter angegeben werden.
   Zwischen die geschweiften Klammern wird der Funktionscode eingetragen.

Alternative Bedienung

Alternativ können Sie eine neue Funktion auch anlegen, indem Sie das Symbol in der Symbolleiste, das Menü "Datei" oder die entsprechende Tastenkombination benutzen.

Siehe auch

So fügen Sie funktionsbegleitende Informationen hinzu (Seite 869)

So verwenden Sie "Speichern unter..." (Seite 856)

2.12.3 So schreiben Sie Funktionscode

Einleitung

Der Funktionscode wird im Editierfenster der Funktion geschrieben. Die Programmiersprache ist ANSI-C.

Jede Projekt- oder Standard-Funktion kann in ihrem Code andere Funktionen aufrufen. Das können Projekt-Funktionen, Standard-Funktionen, interne Funktionen oder DLL-Funktionen sein. Damit die aufgerufene Funktion in der aufrufenden Funktion bekannt ist, ist die Zeile #include "apdefap.h" als erste Codezeile in den Code der aufrufenden Funktion einzufügen.

Im Navigationsfenster unter "Interne Funktionen", "c_bib" steht Ihnen die C-Funktionsbibliothek zur Verfügung.

Die erste Codezeile enthält den Typ des Rückgabewerts und den Standardnamen der neuen Funktion. In der nachfolgenden Klammer können Übergabeparameter angegeben werden.

Zwischen die geschweiften Klammern wird der Funktionscode eingetragen.

Vorgehensweise

1. Führen Sie im Navigationsfenster einen Doppelklick auf die betreffende Funktion aus, um sie im Editierfenster zu öffnen.
2. Setzen Sie die Schreibmarke an die Stelle, an der Sie schreiben möchten.
3. Tragen Sie den gewünschten Code ein.

Alternative Bedienung

Sie haben auch folgende alternative Möglichkeiten, eine Funktion zu öffnen:

Öffnen Sie im Navigationsfenster das Kontextmenü der gewünschten Aktion und wählen Sie "Öffnen" oder verwenden Sie das Menü "Datei\Öffnen..." Sie können auch auf die Schaltfläche  auf der Symbolleiste "Standard" klicken oder die entsprechende Tastenkombination verwenden.

Hinweis

Der verfügbare Speicherplatz für lokale Variablen (Variablen, die innerhalb der geschweiften Klammern der Funktion definiert werden) beträgt maximal 32 KByte.

Siehe auch

So verwenden Sie Standard- und Projekt-Funktionen (Seite 868)

So verwenden Sie interne Funktionen (Seite 867)

Editierfunktionen mit der Maus (Seite 852)

Editierfunktionen mit der Tastatur (Seite 851)

Arbeiten im Editierfenster (Seite 849)

2.12.4 So verwenden Sie interne Funktionen

Einleitung

Als Teil des Funktionscodes können Sie alle internen Funktionen verwenden. Die internen Funktionen finden Sie im Navigationsfenster in der Gruppe "Interne Funktionen".

Wenn Sie eine Funktion mit Hilfe des Parametrierungsdialogs eingefügt haben, gibt der Kommentar zur Funktion den Typ des Rückgabewerts an.

Vorgehensweise

1. Setzen Sie die Schreibmarke an die Stelle, an der die interne Funktion eingefügt werden soll
2. Öffnen Sie im Navigationsfenster das Kontextmenü der einzufügenden internen Funktion
3. Wählen Sie "Parameterversorgung". Der Parametrierungsdialog wird geöffnet
   Der Parametrierungsdialog enthält für jeden Parameter eine Zeile. In der Spalte "Wert" wird der betreffende aktuelle Parameter eingetragen.

4. Tragen Sie in der Spalte "Wert" für jeden erforderlichen Parameter den aktuellen Wert ein. Dies kann entweder durch einen direkten Eintrag mit der Tastatur geschehen, oder Sie öffnen das Menü in der Spalte "Wert" (Einfachklick, dann Klick auf die angezeigte Schaltfläche). Aus dem Menü kann der Auswahldialog für Variablen, Bilder oder Grafik-Objekte geöffnet werden.
5. Bestätigen Sie Ihre Eingaben mit "OK". Die parametrierte Funktion wird im Editierfenster an der Stelle der Einfügemarke eingefügt.

Alternative Bedienung

Alternativ können Sie den Parametrierungsdialog der internen Funktion auch öffnen, indem Sie einen Doppelklick auf die einzufügende Funktion ausführen.

Hinweis

Wenn Sie den Parametrierungsdialog mit "OK" schließen, ohne die aktuellen Parameterwerte eingetragen zu haben, dann wird die interne Funktion mit ihren formalen Parametern eingefügt. Die Parametrierung können Sie dann zu einem späteren Zeitpunkt im Editierfenster nachholen.

Statt den Parametrierungsdialog zu verwenden, können Sie die Funktion auch mit der Tastatur eintragen.

2.12.5 So verwenden Sie Standard- und Projekt-Funktionen

Einleitung

Als Teil des Funktionscodes können Sie alle Projekt- und Standard-Funktionen verwenden, wenn Sie zuvor den Header mit #include "apdefap.h" eingefügt haben. Die Projekt-Funktionen finden Sie im Navigationsfenster in der Gruppe "Projekt-Funktionen". Die Standard-Funktionen finden Sie im Navigationsfenster in der Gruppe "Standard-Funktionen".

Projekt-Funktionen werden in die Header-Datei Ap_pbib.h, Standard-Funktionen in die Header-Datei Ap_glob.h eingetragen. Diese Einträge werden vom System vorgenommen. Die Header-Datei Ap_glob.h ist in die Header-Datei Ap_pbib.h eingebunden. Die Header-Datei Ap_pbib.h ihrerseits ist in die Header-Datei Apdefap.h eingebunden. Damit sind alle Projekt- und Standard-Funktionen in der Header-Datei Apdefap.h deklariert.

Damit der Compiler die eingefügten Projekt- und Standard-Funktionen kennt, ist vor die erste Zeile des Funktionscodes die Zeile #include "apdefap.h" einzufügen.

Wenn Sie die Funktion mit Hilfe des Parametrierungsdialogs eingefügt haben, gibt der Kommentar zur Funktion den Typ des Rückgabewerts an.

Vorgehensweise

1. Setzen Sie die Schreibmarke an die Stelle, an der die Projekt- oder Standard-Funktion eingefügt werden soll
2. Öffnen Sie im Navigationsfenster das Kontextmenü der einzufügenden Funktion
3. Wählen Sie "Parameterversorgung". Der Parametrierungsdialog wird geöffnet
   Der Parametrierungsdialog enthält für jeden Parameter eine Zeile. In der Spalte "Wert" wird der betreffende aktuelle Parameter eingetragen.
4. Tragen Sie in der Spalte "Wert" für jeden erforderlichen Parameter den aktuellen Wert ein. Dies kann entweder durch einen direkten Eintrag mit der Tastatur geschehen, oder Sie öffnen das Menü in der Spalte "Wert" (Einfachklick, dann Klick auf die angezeigte Schaltfläche). Aus dem Menü kann der Auswahldialog für Variablen, Bilder oder Grafik-Objekte geöffnet werden.
5. Bestätigen Sie Ihre Eingaben mit "OK"

Hinweis

Benötigt eine Funktion keine Parameter, dann wird sie sofort in den Funktionscode eingefügt, ohne den Parametrierungsdialog zu öffnen.

Wenn Sie den Parametrierungsdialog mit "OK" schließen, ohne die aktuellen Parameterwerte eingetragen zu haben, dann wird die Funktion mit ihren formalen Parametern eingefügt. Die Parametrierung können Sie dann zu einem späteren Zeitpunkt im Editierfenster nachholen.

2.12.6 So fügen Sie funktionsbegleitende Informationen hinzu

Einleitung

Jede Funktion kann mit zusätzlichen Informationen versehen werden.

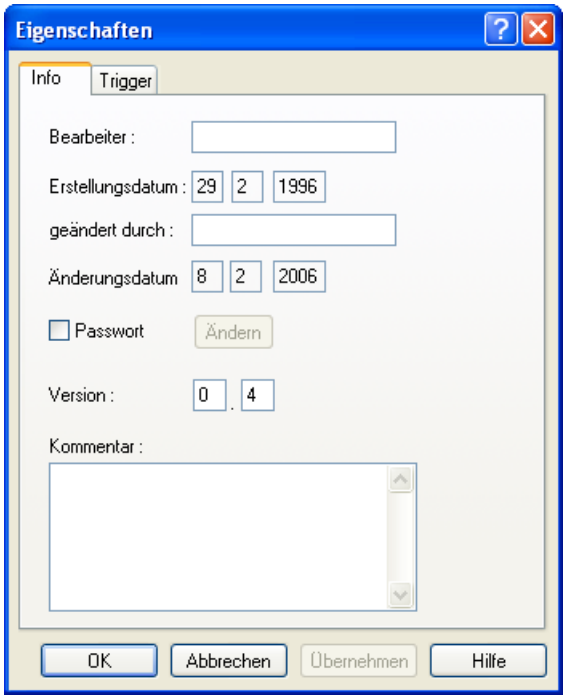
Wenn Sie eine neue Funktion anlegen, wird automatisch und nicht änderbar das Erstellungsdatum in den funktionsbegleitenden Informationen eingetragen. Zusätzlich wird der Funktion die Versionsnummer 1.0 vergeben. Die Versionsnummer können Sie beim bearbeiten einer Funktion individuell vergeben. Wenn Sie eine Funktion ändern und speichern, wird automatisch und nicht änderbar das aktuelle Änderungsdatum eingetragen. In diesem Dialog kann die Funktion durch ein Passwort gegen Änderungen und Einsicht geschützt werden.

Voraussetzung

Die Funktion, auf die sich die Informationen beziehen, muss im Editierfenster geöffnet sein.

Vorgehensweise

1. Betätigen Sie die Schaltfläche in der Symbolleiste "Bearbeiten". Es öffnet sich der Dialog "Eigenschaften".
2. Nehmen Sie hier die gewünschten Einträge vor



3. Bestätigen Sie Ihre Eingaben mit "OK"

Alternative Bedienung

Alternativ können Sie den Dialog "Eigenschaften" auch folgendermaßen öffnen:

Wählen Sie im Menü "Bearbeiten" den Eintrag "Info" oder verwenden Sie die entsprechende Tastenkombination.

Siehe auch

So schützen Sie eine Funktion gegen Änderungen und Einsicht (Seite 871)

So stellen Sie verschiedene Ansichten ein (Seite 855)

Arbeiten mit den Symbolleisten (Seite 852)

2.12.7 So schützen Sie eine Funktion gegen Änderungen und Einsicht

Einleitung

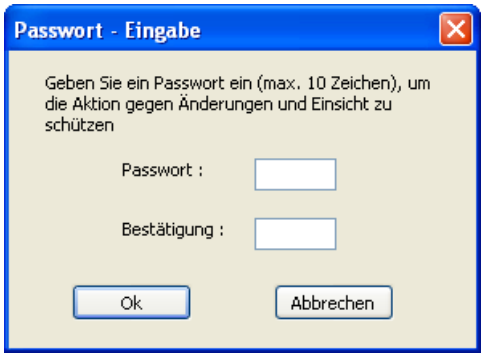
Sie können Funktionen durch ein Passwort gegen Änderungen und Einsicht schützen. Das Passwort ist Bestandteil der funktionsbegleitenden Informationen.

Voraussetzung

Die zu schützende Funktion muss im Editierfenster geöffnet sein.

Vorgehensweise

1. Betätigen Sie die Schaltfläche in der Symbolleiste "Bearbeiten". Es öffnet sich der Dialog "Eigenschaften".
2. Aktivieren Sie das Kontrollkästchen "Passwort"
3. Betätigen Sie die nunmehr aktive Schaltfläche "Ändern"



4. Geben Sie in die Zeile "Passwort" das Passwort ein
5. Geben Sie in die Zeile "Bestätigung" das Passwort ein zweites mal ein
6. Bestätigen Sie Ihre Eingaben mit "OK"
7. Schließen Sie das Dialogfenster mit "OK"

Alternative Bedienung

Alternativ können Sie den Dialog "Eigenschaften" auch folgendermaßen öffnen:

Wählen Sie im Menü "Bearbeiten" den Eintrag "Info" oder verwenden Sie die entsprechende Tastenkombination.

Hinweis

Eine mit Passwort geschützte Funktion lässt sich im Editierfenster nur mit Angabe des Passworts öffnen.

Möchten Sie den Passwortschutz aufheben, dann deaktivieren Sie das Kontrollkästchen "Passwort".

Siehe auch

So stellen Sie verschiedene Ansichten ein (Seite 855)

Arbeiten mit den Symbolleisten (Seite 852)

2.12.8 So übersetzen und speichern Sie eine Funktion

Einleitung

Um eine Funktion verwenden zu können, muss sie zuerst übersetzt werden. Es wird nur die Funktion im aktiven Editierfenster übersetzt.

Fehler, die der Compiler meldet, werden im unteren Teil des Editierfensters angezeigt. Jede Meldung steht in einer eigenen Zeile. Die Zeile enthält die Zeilennummer der Codezeile, in der der Fehler auftrat, einen hexadezimal verschlüsselten Fehlercode und eine verbale Darstellung des aufgetretenen Fehlers.

Ein Doppelklick auf eine Fehlermeldung markiert die Codezeile, in der der Fehler auftrat.

Es ist sinnvoll, die zuerst aufgelistete Fehlermeldung zu prüfen, denn die weiteren Fehlermeldungen können Folgefehler des ersten Fehlers sein. Ist der erste Fehler behoben, dann verschwinden nach der nächsten Übersetzung auch die Folgefehler.

Damit vorgenommene Änderungen an einer Funktion dauerhaft erhalten bleiben, muss sie gespeichert werden.

Speichern Sie eine fehlerhaft übersetzte oder gar nicht übersetzte Funktion, dann erhält sie im Navigationsfenster das Symbol  .

Speichern Sie eine fehlerfrei übersetzte Funktion, dann erhält sie im Navigationsfenster das Symbol  .

Voraussetzung

Die zu compilierende Funktion muss im Editierfenster geöffnet sein.

Vorgehensweise

1. Stellen Sie in der Symbolleiste die Sprache ein, in der die C-Funktion kompiliert wird.
2. Betätigen Sie die Schaltfläche  in der Symbolleiste "Bearbeiten"
3. Prüfen Sie die Meldungen des Compilers im unteren Teil des Editierfensters
4. Hat der Compiler Fehler gemeldet, dann muss der Funktionscode korrigiert werden. Ist das geschehen, beginnen Sie wieder mit Schritt 1 in dieser Tabelle
5. Hat der Compiler Warnungen gemeldet, dann kann der Funktionscode korrigiert werden. Haben Sie den Funktionscode korrigiert, beginnen Sie wieder mit Schritt 1 in dieser Tabelle, ansonsten gehen Sie zu Schritt 6
6. Betätigen Sie die Schaltfläche  in der Symbolleiste "Standard"

Alternative Bedienung

Alternativ können Sie den Übersetzungsvorgang auch folgendermaßen anstoßen:

Öffnen Sie das Menü "Bearbeiten" und wählen Sie den Eintrag "Übersetzen" oder wählen Sie den Eintrag "Übersetzen" aus dem Kontextmenü des Editierfensters oder verwenden Sie die entsprechende Tastenkombination.

Alternativ können Sie auch folgendermaßen speichern:

Öffnen Sie das Menü "Datei" und wählen Sie den Eintrag "Speichern" oder verwenden Sie die entsprechende Tastenkombination.

Hinweis

Wenn Variablennamen in einer C-Funktion mehrfach verwendet werden, dann gibt der Compiler keine Fehlermeldung aus. Dies ist auch der Fall, wenn ein Variablenname sowohl als Übergabeparameter als auch als lokale Variablendefinition verwendet wird.

Zum Beispiel führt folgendes fehlerhaftes Script zu keiner Fehlermeldung im Compiler:

```
void neue_Funktion(DWORD dwMyVar)
{
DWORD dwMyVar = 0;
}
```

Meldung im Ausgabefenster des Compilers:

Compiling ...

0 Error(s), 0 Warning(s)

Siehe auch

Laufzeitverhalten von Aktionen (Seite 899)

2.12.9 So benennen Sie eine Funktion um

Einleitung

Es ist sinnvoll, eine Funktion umzubenennen, wenn sie neu erstellt wurde.

Dazu wird der Funktionsname im Editierfenster entsprechend geändert. Da hiermit der Code geändert wurde, muss die Funktion neu übersetzt werden. Der im Navigationsfenster angezeigte alte Funktionsname wird nach der Übersetzung mit dem vorangestellten Zeichen * gekennzeichnet.

Abschließend ist die geänderte Funktion zu speichern, wobei Pfad und Dateiname geändert werden können. Die alte Funktion sollte danach gelöscht werden, um eine Anhäufung nicht mehr aktueller Funktionen zu vermeiden.

Hinweis

Beachten Sie bitte, dass Sie in Funktionsnamen nur bestimmte Zeichen verwenden dürfen: Buchstaben, mit Ausnahme der nationalen Sonderzeichen, Ziffern und den Unterstrich.

Vorgehensweise

1. Ändern Sie den Funktionsnamen im Editierfenster
2. Betätigen Sie die Schaltfläche  in der Symbolleiste "Bearbeiten". Die Funktion wird übersetzt.
3. Betätigen Sie auf die Schaltfläche  in der Symbolleiste "Standard", um die Funktion zu speichern.
4. Legen Sie gegebenenfalls einen anderen Pfad und einen anderen Dateinamen fest
5. Bestätigen Sie Ihre Angaben mit der Schaltfläche "Speichern"

Alternative Bedienung

Alternativ können Sie den Übersetzungsvorgang auch folgendermaßen anstoßen:

Öffnen Sie das Menü "Bearbeiten" und wählen Sie den Eintrag "Übersetzen" oder wählen Sie den Eintrag "Übersetzen" aus dem Kontextmenü des Editierfensters oder verwenden Sie die entsprechende Tastenkombination.

Alternativ können Sie auch folgendermaßen speichern:

Öffnen Sie das Menü "Datei" und wählen Sie den Eintrag "Speichern" oder verwenden Sie die entsprechende Tastenkombination.

2.12.10 So verwenden Sie Funktionen aus anderen Quellen

Einleitung

Projekt-Funktionen aus anderen WinCC-Projekten und Standard-Funktionen aus anderen WinCC Systemen können auch für das aktuelle Projekt nutzbar gemacht werden. Dazu müssen sie in das aktuelle Projekt eingebracht werden.

Abgesehen vom Ablageort der Funktionen im Dateisystem besteht in der Vorgehensweise für Projekt- und Standardfunktionen kein Unterschied.

Vorgehensweise

1. Kopieren Sie die Funktionen. Projekt-Funktionen werden im Projektpfad in das Verzeichnis "library" kopiert. Standard-Funktionen werden im WinCC-Pfad in das Verzeichnis "aplib\..." kopiert. Der Inhalt des Navigationsfensters wird automatisch aktualisiert.
2. Betätigen Sie die Schaltfläche  in der Symbolleiste "Standard". Durch das Neugenerieren des Headers werden die einkopierten Funktionen bekannt gemacht und können nun im aktuellen Projekt zur Projektierung verwendet werden.

Alternative Bedienung

Alternativ können Sie den Generiervorgang auch folgendermaßen anstoßen:

Öffnen Sie das Menü "Optionen" und wählen Sie den Eintrag "Header neu generieren" oder verwenden Sie die entsprechende Tastenkombination.

Hinweis

Benutzerdefinierte oder geänderte Standard-Funktionen werden bei Neuinstallation oder Hochrüstung von WinCC gelöscht bzw. durch die Originalfunktionen ersetzt.

Befindet sich WinCC in Runtime, so wird das Runtime System durch das Neugenerieren des Headers nicht beeinflusst.

2.13 Aktionen erstellen und bearbeiten

2.13.1 Aktionen erstellen und bearbeiten

Einleitung

Es wird zwischen globalen und lokalen Aktionen unterschieden. In einem Client-Server Projekt werden globale Aktionen auf allen Rechnern des Projekts ausgeführt, während lokale Aktionen nur auf dem zugeordneten Rechner ausgeführt werden.

Eine globale Aktion kann z.B. dazu verwendet werden, um eine Berechnung auf allen Rechnern des Projekts auszuführen.

Eine lokale Aktion kann z.B. dazu verwendet werden, um ein Protokoll an einem Server auszugeben.

Die Erstellung und Bearbeitung beider Aktionstypen ist völlig identisch.

Unterschiede zwischen Aktionen und Funktionen

- Aktionen können, im Gegensatz zu Funktionen, einen Trigger besitzen. Das heißt, eine Funktion allein kann in Runtime niemals zum Ablauf gebracht werden.
- Aktionen können exportiert und importiert werden.
- Aktionen kann eine Berechtigung zugewiesen werden. Die Berechtigung bezieht sich auf die Bedienmöglichkeiten im Diagnosefenster Global Script - Runtime.
- Eine Aktion besitzt keine Parameter.

Auffinden von Aktionen

Der Zugriff auf bestehende Aktionen oder die Erstellung neuer Aktionen erfolgt über das Navigationsfenster von Global Script.

Aktionen werden im Dateisystem wie folgt abgelegt:

Bearbeiten und Übersetzen von Aktionen

Eine Aktion wird in einem eigenen Editerfenster bearbeitet und übersetzt. Das Editierfenster enthält nach dem Übersetzungslauf Meldungen des Compilers. Das können Warnungen und/ oder Fehlermeldungen sein. In jedem Fall wird zusammenfassend die Anzahl der Warnungen und Fehlermeldungen ausgegeben.

Darstellung von Aktionen

Wenn Sie eine syntaktisch fehlerhafte Aktion speichern, wird sie im Navigationsfenster von Global Script mit nebenstehendem Symbol angezeigt.

Wenn Sie eine syntaktisch korrekte Aktion ohne Trigger speichern, wird sie im Navigationsfenster von Global Script mit nebenstehendem Symbol angezeigt.

Wenn Sie eine syntaktisch korrekte Aktion mit Trigger speichern, wird sie im Navigationsfenster von Global Script mit nebenstehendem Symbol angezeigt.

Dies ist bei der Erstellung von Aktionen zu beachten

WinCC bietet mit CrossReference die Möglichkeit, Querverweislisten zu erstellen. Damit beim Aufbau der Querverweisliste die Variablen und Bilder bei Verwendung von Funktionsaufrufen innerhalb von Aktionen erkannt werden, sollte die weiter unten beschriebene Codierregel eingehalten werden.

Aktionen umbenennen

Im Navigationsfenster werden Aktionen immer mit ihrem Dateinamen angezeigt. Das Umbenennen einer Aktion ist gleich bedeutend mit dem Umbenennen der Datei, die den Aktionscode enthält.

Systemverhalten, wenn in Runtime Aktionen verändert, gelöscht und gespeichert werden

Wenn eine lokale Aktion während Runtime gespeichert wird, dann werden auf dem Rechner, zu dem die lokale Aktion gehört, alle lokalen und globalen Aktionen des Rechners zurückgesetzt.

Wird eine globale Aktion während Runtime gespeichert, dann werden alle lokalen und globalen Aktionen des gesamten Projekts und damit auf allen Rechnern zurückgesetzt.

Das Zurücksetzen hat zur Folge, dass z. B. die Variablen und Zeiten, die als Trigger in den Aktionen verwendet werden, neu initialisiert werden und dadurch die Aktion ausgelöst wird.

Statische Variablen, die in den zurückgesetzten Aktionen verwendet werden, werden neu initialisiert.

Mögliche Ursachen, wenn eine Aktion in Runtime nicht ausgeführt wird

Warum eine Aktion in Runtime nicht ausgeführt wird kann folgende Gründe haben:

- Die Aktion hat keinen Trigger
- Die Aktion wurde nicht übersetzt
- In der Startliste des Projekts ist Global Script Runtime nicht aktiviert

Hinweis

Vor dem Erstellen einer Aktion sollte geprüft werden, ob die betreffende Funktionalität nicht auch im AG realisiert werden kann.

Siehe auch

So schützen Sie eine Aktion gegen Änderungen und Einsicht (Seite 883)

So verwenden Sie projektfremde Aktionen (Seite 897)

So benennen Sie eine Aktion um (Seite 896)

So importieren Sie eine Aktion (Seite 895)

So exportieren Sie eine Aktion (Seite 894)

So weisen Sie eine Berechtigung zu (Seite 893)

Trigger (Seite 886)

So übersetzen und speichern Sie eine Aktion (Seite 884)

So fügen Sie aktionsbegleitende Informationen hinzu (Seite 882)

So bearbeiten Sie eine Aktion (Seite 881)

So legen Sie eine neue Aktion an (Seite 880)

WinCC Codierregel (Seite 879)

So nehmen Sie Global Script Runtime in die Anlaufliste des Projekts auf (Seite 841)

Merkmale von globalen Aktionen (Seite 840)

Merkmale von lokalen Aktionen (Seite 839)

2.13.2 WinCC Codierregel

Codierregel für den Einsatz von CrossReference

WinCC bietet mit CrossReference die Möglichkeit, Querverweislisten zu erstellen. Damit beim Aufbau der Querverweisliste die Variablen und Bilder bei Verwendung von Funktionsaufrufen innerhalb von Aktionen erkannt werden, muss die hier beschriebene Codierregel eingehalten werden.

Der Aktionscode beginnt mit zwei Abschnitten. Im ersten Abschnitt müssen alle benutzten Variablen deklariert werden, im zweiten Abschnitt alle benutzten Bildnamen.

Innerhalb der Abschnitte dürfen Sie keine weitere Anweisungen eintragen.

Beide Abschnitte sind in Kommentarform beim Erstellen einer neuen Aktion bereits vorgegeben:

```
// WINCC:TAGNAME_SECTION_START
// syntax: #define TagNameInAction "DMTagName"
// next TagID : 1
// WINCC:TAGNAME_SECTION_END

// WINCC:PICNAME_SECTION_START
// syntax: #define PicNameInAction "PictureName"
// next PicID : 1
// WINCC:PICNAME_SECTION_END
```

Die Abschnitte werden jetzt beispielsweise wie folgt ergänzt:

```
// WINCC:TAGNAME_SECTION_START
// syntax: #define TagNameInAction "DMTagName"
// next TagID : 1
#define ApcTagName1 "TagName1"
// WINCC:TAGNAME_SECTION_END

// WINCC:PICNAME_SECTION_START
// syntax: #define PicNameInAction "PictureName"
// next PicID : 1
#define ApcPicName1 "PicName1"
#define ApcPicName2 "PicName2"
```

```
#define ApcPicName3 "PicName3"
// WINCC:PICNAME_SECTION_END
```

Der Aufruf der Funktionen zum Lesen und Schreiben der Variablen und die Verwendung der Bildnamen muss dann mit den definierten Namen erfolgen:

```
GetTagDWord(ApcTagName1);
OpenPicture(ApcPicName1);
SetPictureName(ApcPicName2,"PictureWindow1",ApcPicName3);
```

2.13.3 So legen Sie eine neue Aktion an

Einleitung

In einem Client-Server Projekt werden globale Aktionen auf allen Rechnern des Projekts ausgeführt, während lokale Aktionen nur auf dem zugeordneten Rechner ausgeführt werden.

Die Vorgehensweise für globale und lokale Aktionen ist identisch. Durch den Ablageort im Navigationsfenster legen Sie den Typ fest (global oder lokal).

Global Script schlägt einen Standard-Dateinamen für die neue Aktion vor.

Eine neu angelegte Aktion enthält bereits die Anweisung #include "apdefap.h". Damit sind innerhalb der Aktion alle Funktionen bekannt. In der dritten Zeile steht der Aktionsname. Die ersten drei Zeilen können weder gelöscht noch verändert werden. Das bedeutet, dass in jeder Aktion, ohne besondere Vorkehrungen treffen zu müssen, alle Funktionen verwendet werden können. Außerdem ist der Rückgabewert jeder Aktion immer vom Typ int und ist bereits mit dem Wert 0 belegt.

Der Rückgabewert einer Aktion kann in Verbindung mit GSC-Runtime zu Diagnosezwecken genutzt werden.

Der Aktionscode beginnt mit einem Codegerüst in Kommentarform. Wird dieses Codegerüst nach der Codierregel ausgefüllt, dann werden Variablen und Bildnamen von Cross Reference erkannt.

Vorgehensweise

1. Öffnen Sie im Navigationsfenster das Kontextmenü des gewünschten Aktionstyps
2. Wählen Sie "Neu"

Alternative Bedienung

Eine neue globale Aktion können Sie alternativ auch anlegen, indem Sie das Symbol in der Symbolleiste, das Menü "Datei" oder die entsprechende Tastenkombination benutzen.

Siehe auch

GSC-Runtime (Seite 900)

WinCC Codierregel (Seite 879)

2.13.4 So bearbeiten Sie eine Aktion

Einleitung

Eine Aktion bearbeiten Sie im Editierfenster genauso wie eine Funktion. Lediglich die ersten 3 Zeilen sind nicht editierbar.

Die Aktion muss einen Rückgabewert besitzen. Der Rückgabewert ist vom Typ int und ist mit dem Wert 0 vorbesetzt. Der Rückgabewert kann verändert und in Verbindung mit GSC-Runtime zu Diagnosezwecken verwendet werden. Der Typ des Rückgabewerts ist nicht veränderbar.

Damit die Aktion in Runtime ausgeführt wird, muss sie mit einem Trigger versehen werden.

Vorgehensweise

1. Führen Sie im Navigationsfenster einen Doppelklick auf die betreffende Aktion aus, um sie zur Bearbeitung zu öffnen.
2. Bearbeiten Sie den Aktionscode.

Alternative Bedienung

Sie haben auch folgende alternative Möglichkeiten, eine Aktion zu öffnen:

Öffnen Sie im Navigationsfenster das Kontextmenü der gewünschten Aktion und wählen Sie "Öffnen" oder verwenden Sie das Menü "Datei\Öffnen..." Sie können auch auf die Schaltfläche auf der Symbolleiste "Standard" klicken oder die entsprechende Tastenkombination verwenden.

Hinweis

Der verfügbare Speicherplatz für lokale Variablen (Variablen, die innerhalb der geschweiften Klammern der Aktion definiert werden) beträgt maximal 32 KByte.

Siehe auch

GSC-Runtime (Seite 900)

So schreiben Sie Funktionscode (Seite 866)

2.13.5 So fügen Sie aktionsbegleitende Informationen hinzu

Einleitung

Jede Aktion kann mit zusätzlichen Informationen versehen werden.

Wenn Sie eine neue Aktion anlegen, wird automatisch und nicht änderbar das Erstellungsdatum in den aktionsbegleitenden Informationen eingetragen. Zusätzlich wird der Aktion die Versionsnummer 1.0 vergeben. Die Versionsnummer können Sie beim bearbeiten einer Aktion individuell vergeben. Wenn Sie eine Aktion ändern und speichern, wird automatisch und nicht änderbar das aktuelle Änderungsdatum eingetragen.

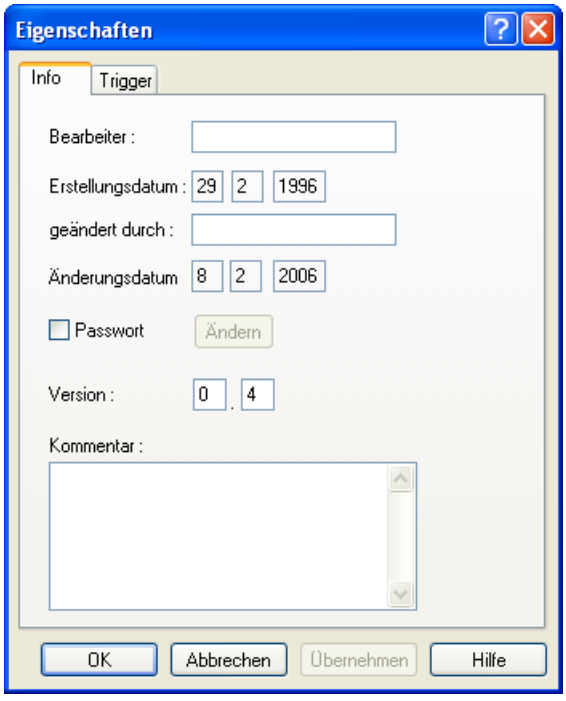
In diesem Dialog kann die Aktion durch ein Passwort gegen Änderungen und Einsicht geschützt werden.

Voraussetzung

Die Aktion, auf die sich die Informationen beziehen, muss im Editierfenster geöffnet sein.

Vorgehensweise

1. Betätigen Sie die Schaltfläche in der Symbolleiste "Bearbeiten". Es öffnet sich der Dialog "Eigenschaften".
2. Nehmen Sie hier die gewünschten Einträge vor



3. Bestätigen Sie Ihre Eingaben mit "OK"

Alternative Bedienung

Alternativ können Sie den Dialog "Eigenschaften" auch folgendermaßen öffnen:

Wählen Sie im Menü "Bearbeiten" den Eintrag "Info" oder verwenden Sie die entsprechende Tastenkombination.

Siehe auch

So schützen Sie eine Aktion gegen Änderungen und Einsicht (Seite 883)

So stellen Sie verschiedene Ansichten ein (Seite 855)

Arbeiten mit den Symbolleisten (Seite 852)

2.13.6 So schützen Sie eine Aktion gegen Änderungen und Einsicht

Einleitung

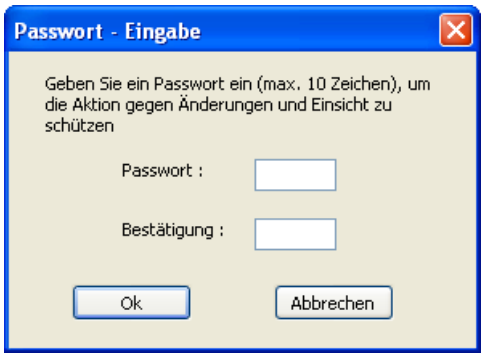
Sie können Aktionen durch ein Passwort gegen Änderungen und Einsicht schützen. Das Passwort ist Bestandteil der aktionsbegleitenden Informationen.

Voraussetzung

Die zu schützende Aktion muss im Editierfenster geöffnet sein.

Vorgehensweise

1. Betätigen Sie die Schaltfläche in der Symbolleiste "Bearbeiten". Es öffnet sich der Dialog "Eigenschaften".
2. Aktivieren Sie das Kontrollkästchen "Passwort"
3. Betätigen Sie die nunmehr aktive Schaltfläche "Ändern"



4. Geben Sie in die Zeile "Passwort" das Passwort ein
5. Geben Sie in die Zeile "Bestätigung" das Passwort ein zweites mal ein
6. Bestätigen Sie Ihre Eingaben mit "OK"
7. Schließen Sie das Dialogfenster mit "OK"

Alternative Bedienung

Alternativ können Sie den Dialog "Eigenschaften" auch folgendermaßen öffnen:

Wählen Sie im Menü "Bearbeiten" den Eintrag "Info" oder verwenden Sie die entsprechende Tastenkombination.

Hinweis

Eine mit Passwort geschützte Aktion lässt sich im Editierfenster nur mit Angabe des Passworts öffnen.

Möchten Sie den Passwortschutz aufheben, dann deaktivieren Sie das Kontrollkästchen "Passwort".

Siehe auch

So stellen Sie verschiedene Ansichten ein (Seite 855)

Arbeiten mit den Symbolleisten (Seite 852)

2.13.7 So übersetzen und speichern Sie eine Aktion

Einleitung

Um eine Aktion verwenden zu können, muss sie zuerst übersetzt werden. Es wird nur die Aktion im aktiven Editierfenster übersetzt.

Fehler, die der Compiler meldet, werden im unteren Teil des Editierfensters angezeigt. Jede Meldung steht in einer eigenen Zeile. Die Zeile enthält die Zeilennummer der Codezeile, in der der Fehler auftrat, einen hexadezimal verschlüsselten Fehlercode und eine verbale Darstellung des aufgetretenen Fehlers.

Ein Doppelklick auf solch eine Zeile markiert die Codezeile, in der der Fehler auftrat.

Es ist sinnvoll, die zuerst aufgelistete Fehlermeldung zu prüfen, denn die weiteren Fehlermeldungen können Folgefehler des ersten Fehlers sein. Ist der erste Fehler behoben, dann verschwinden nach der nächsten Übersetzung auch die Folgefehler.

Voraussetzung

Die zu compilierende Aktion muss im Editierfenster geöffnet sein.

Vorgehensweise

1. Stellen Sie in der Symbolleiste die Sprache ein, in der die C-Aktion kompliliert wird.
2. Betätigen Sie die Schaltfläche  in der Symbolleiste "Bearbeiten"
3. Prüfen Sie die Meldungen des Compilers im unteren Teil des Editierfensters

4. Hat der Compiler Fehler gemeldet, dann muss der Aktionscode korrigiert werden. Ist das geschehen, beginnen Sie wieder mit Schritt 1 in dieser Tabelle
5. Hat der Compiler Warnungen gemeldet, dann kann der Aktionscode korrigiert werden. Haben Sie den Aktionscode korrigiert, beginnen Sie wieder mit Schritt 1 in dieser Tabelle, ansonsten gehen Sie zu Schritt 6
6. Betätigen Sie die Schaltfläche in der Symbolleiste "Standard"

Alternative Bedienung

Alternativ können Sie den Übersetzungsvorgang auch folgendermaßen anstoßen:

Öffnen Sie das Menü "Bearbeiten" und wählen Sie den Eintrag "Übersetzen" oder wählen Sie den Eintrag "Übersetzen" aus dem Kontextmenü des Editierfensters oder verwenden Sie die entsprechende Tastenkombination.

Alternativ können Sie auch folgendermaßen speichern:

Öffnen Sie das Menü "Datei" und wählen Sie den Eintrag "Speichern" oder verwenden Sie die entsprechende Tastenkombination.

Darstellung von Aktionen

Wenn Sie eine syntaktisch fehlerhafte Aktion speichern, wird sie im Navigationsfenster von Global Script mit nebenstehendem Symbol angezeigt.

Wenn Sie eine syntaktisch korrekte Aktion ohne Trigger speichern, wird sie im Navigationsfenster von Global Script mit nebenstehendem Symbol angezeigt.

Wenn Sie eine syntaktisch korrekte Aktion mit Trigger speichern, wird sie im Navigationsfenster von Global Script mit nebenstehendem Symbol angezeigt.

Siehe auch

Laufzeitverhalten von Aktionen (Seite 899)

2.13.8 Trigger

2.13.8.1 Trigger

Definition und Verwendung von Triggern

Trigger werden gebraucht, um Aktionen in Runtime auszuführen. Dazu wird ein Trigger mit einer Aktion verbunden und bildet somit das auslösende Ereignis für den Aufruf der Aktion. Aktionen ohne Trigger werden nicht ausgeführt.

Triggertypen

Folgende Trigger werden angeboten:

Azyklische Trigger

Sie bestehen aus der Angabe des Datums und der Uhrzeit. Die Aktion, die mit solch einem Trigger verbunden ist, wird einmal am angegebenen Datum zur angegebenen Uhrzeit ausgeführt.

Zyklische Trigger

Sie bestehen aus der Angabe eines Zeitintervalls und dessen Beginn. Folgende zyklische Trigger stehen zur Verfügung:

- Standardzyklus. Der Beginn des ersten Zeitintervalls fällt mit dem Start von Runtime zusammen. Die Länge des Intervalls wird durch den Zyklus bestimmt.
- Stündlich. Der Beginn des Zeitintervalls wird mit Minute und Sekunde festgelegt. Die Länge des Intervalls beträgt eine Stunde.
- Täglich. Der Beginn des Zeitintervalls wird mit der Uhrzeit (Stunde, Minute und Sekunde) festgelegt. Die Länge des Intervalls beträgt ein Tag.
- Wöchentlich. Der Beginn des Zeitintervalls wird mit Wochentag (Montag, Dienstag, ...) und Uhrzeit festgelegt. Die Länge des Intervalls beträgt eine Woche.
- Monatlich. Der Beginn des Zeitintervalls wird mit Tag und Uhrzeit festgelegt. Die Länge des Intervalls beträgt einen Monat.
- Jährlich. Der Beginn des Zeitintervalls wird mit Tag, Monat und Uhrzeit festgelegt. Die Länge des Intervalls beträgt ein Jahr.

Variablentrigger

Sie bestehen aus der Angabe einer oder mehrerer Variablen. Die Aktion, die mit solch einem Trigger verbunden ist, wird jedes Mal ausgeführt, wenn eine Änderung des Werts einer dieser Variablen festgestellt wurde.
Wie die Variablenwerte abgefragt werden, ist für jede Variable individuell einstellbar. Sie können wählen zwischen einer zyklischen Abfrage mit einem einstellbaren Standardzyklus und einer Reaktion, sobald vom System eine Änderung des Variablenwerts festgestellt wurde. Je nach Wahl der Abfrage kann es vorkommen, dass sich die Variable ändert, die Änderung vom System aber nicht wahrgenommen wird. Die Aktion wird in diesem Fall nicht ausgeführt.

Wirkung von Triggern auf eine Aktion

Ist die Aktion mit nur einem Trigger verbunden, dann wird die Aktion ausgeführt, sobald das auslösende Ereignis eintrifft.

Eine Aktion kann aber auch mit mehreren Triggern verbunden werden, beispielsweise mit einem zyklischen Trigger und mit einem Variablentrigger. Die Aktion wird immer dann ausgeführt, wenn eins dieser auslösenden Ereignisse eintrifft. Treffen zwei Ereignisse gleichzeitig ein, dann wird die Aktion zweimal hintereinander ausgeführt. Ändern sich zwei im Trigger enthaltene Variablen zur selben Zeit, so wird die Aktion nur einmal durchlaufen.

Die Abarbeitung einer Aktion sollte beendet sein, bevor sie ein weiteres mal angestoßen wird, da dies sonst zu einem Überlauf der Warteschlange führen kann.

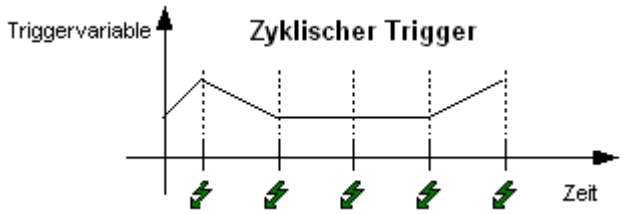
Tipp: Soll die Aktion nicht bei jedem Ereignis ausgeführt werden, dann kann in der Aktion eine Bedingung formuliert werden, die abhängig vom Ergebnis die weitere Ausführung der Aktion steuert. Soll die Aktion nicht weiter ausgeführt werden, dann kann sie mit return <wert> verlassen werden.

Regeln für die Auswahl des Triggers

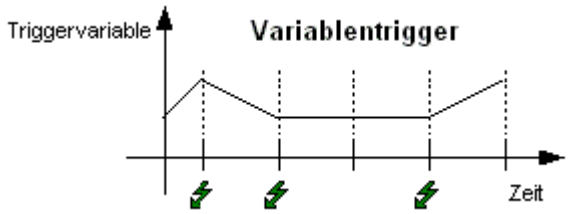
Systembedingt kann nicht garantiert werden, dass eine Aktion mit zyklischem Trigger genau zu den angegebenen Zeiten ausgeführt wird. Soll dies gewährleistet sein, dann ist die Aufgabe (z.B. Überprüfungen) im AG zu realisieren.

Bei Variablenbearbeitung sollte der Variablentrigger dem zyklischen Trigger vorgezogen werden.

- Bei zyklischem Trigger wird die Aktion immer ausgeführt, wenn das Triggerereignis eintritt z.B. alle 20 Sekunden. (= Aktion wird ausgeführt)



- Der Variablentrigger führt die Aktion nur dann aus, wenn sich der Wert der Triggervariable geändert hat. Damit wird eine Reduzierung der Systemlast erreicht (= Aktion wird ausgeführt).



Bei Variablen, die im Trigger enthalten sind, ist der Wert bereits beim Starten der Aktion bekannt. Bei einem GetTag() Aufruf wird der Wert direkt übergeben. Die Abarbeitung erfolgt daher wesentlich schneller, als bei GetTag() Anforderungen auf nicht im Trigger enthaltene Variablen.

Darstellung von Aktionen

Wenn Sie eine syntaktisch fehlerhafte Aktion speichern, wird sie im Navigationsfenster von Global Script mit nebenstehendem Symbol angezeigt.

Wenn Sie eine syntaktisch korrekte Aktion ohne Trigger speichern, wird sie im Navigationsfenster von Global Script mit nebenstehendem Symbol angezeigt.

Wenn Sie eine syntaktisch korrekte Aktion mit Trigger speichern, wird sie im Navigationsfenster von Global Script mit nebenstehendem Symbol angezeigt.

Siehe auch

So löschen Sie einen Trigger (Seite 892)

So ändern Sie einen Trigger (Seite 891)

So fügen Sie einen neuen Trigger vom Typ "Variable" hinzu (Seite 890)

So fügen Sie einen neuen Trigger vom Typ "Timer" hinzu (Seite 888)

2.13.8.2 So fügen Sie einen neuen Trigger vom Typ "Timer" hinzu

Einleitung

Trigger werden benötigt, um Aktionen in Runtime auszuführen. Dazu wird ein Trigger mit einer Aktion verbunden und bildet somit das auslösende Ergeignis für den Aufruf der Aktion. Aktionen ohne Trigger werden nicht ausgeführt.

Trigger vom Typ "Timer" sind azyklische oder zyklische Trigger.

Azyklische Trigger bestehen aus der Angabe des Datums und der Uhrzeit. Die Aktion, die mit solch einem Trigger verbunden ist, wird einmal am angegebenen Datum zur angegebenen Uhrzeit ausgeführt.

Zyklische Trigger bestehen aus der Angabe eines Zeitintervalls und dessen Beginn. Folgende zyklische Trigger stehen zur Verfügung:

- Standardzyklus. Der Beginn des ersten Zeitintervalls fällt mit dem Start des Runtime Systems zusammen. Die Länge des Intervalls wird durch den Zyklus bestimmt.
- Stündlich. Der Beginn des Zeitintervalls wird mit Minute und Sekunde festgelegt. Die Länge des Intervalls beträgt eine Stunde.
- Täglich. Der Beginn des Zeitintervalls wird mit der Uhrzeit (Stunde, Minute und Sekunde) festgelegt. Die Länge des Intervalls beträgt ein Tag.
- Wöchentlich. Der Beginn des Zeitintervalls wird mit Wochentag (Montag, Dienstag, ...) und Uhrzeit festgelegt. Die Länge des Intervalls beträgt eine Woche.

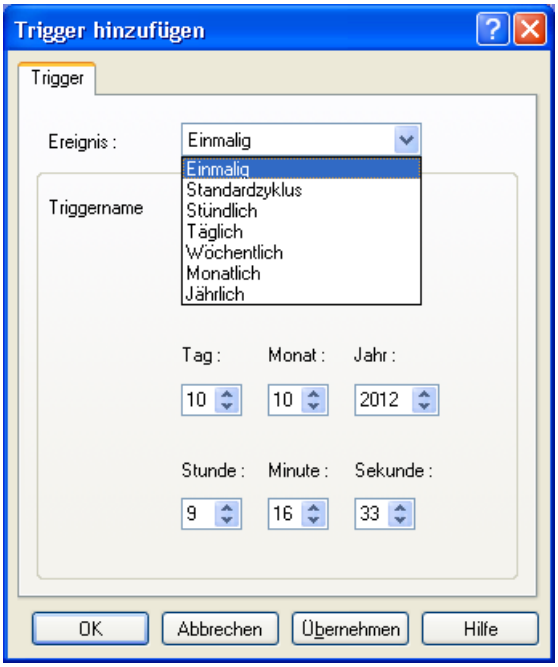
- Monatlich. Der Beginn des Zeitintervalls wird mit Tag und Uhrzeit festgelegt. Die Länge des Intervalls beträgt einen Monat.
- Jährlich. Der Beginn des Zeitintervalls wird mit Tag, Monat und Uhrzeit festgelegt. Die Länge des Intervalls beträgt ein Jahr.

Voraussetzung

Die Aktion, die mit einem Trigger verbunden werden soll, muss im Editierfenster geöffnet sein.

Vorgehensweise

1. Betätigen Sie die Schaltfläche in der Symbolleiste "Bearbeiten". Es öffnet sich der Dialog "Eigenschaften".
2. Wählen Sie die Registerkarte Trigger
3. Wählen Sie die Triggerquelle "Timer" und klicken Sie auf die Schaltfläche "Hinzufügen"
4. Wählen Sie "einmalig", um einen azyklischen Trigger hinzuzufügen oder wählen Sie einen Zyklus, um einen zyklischen Trigger hizuzufügen



5. Vervollständigen Sie die Angaben im Dialog
6. Bestätigen Sie Ihre Einstellungen mit "OK"
7. Schließen Sie den Dialog "Eigenschaften" mit "OK"

Alternative Bedienung

Alternativ können Sie den Dialog "Eigenschaften" auch folgendermaßen öffnen:

Wählen Sie im Menü "Bearbeiten" den Eintrag "Info" oder im Kontextmenü des Editierfensters den Eintrag "Info / Trigger" oder verwenden Sie die entsprechende Tastenkombination.

2.13.8.3 So fügen Sie einen neuen Trigger vom Typ "Variable" hinzu

Einleitung

Trigger werden benötigt, um Aktionen in Runtime auszuführen. Dazu wird ein Trigger mit einer Aktion verbunden und bildet somit das auslösende Ergeignis für den Aufruf der Aktion. Aktionen ohne Trigger werden nicht ausgeführt.

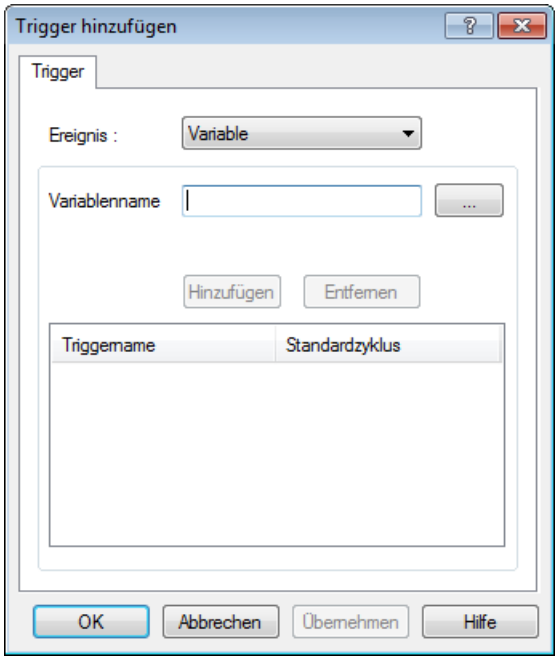
Der Variablentrigger besteht aus der Angabe einer oder mehrerer Variablen. Die Aktion, die mit solch einem Trigger verbunden ist, wird jedes Mal ausgeführt, wenn eine Änderung des Werts einer dieser Variablen festgestellt wird.
Wie die Variablenwerte abgefragt werden, ist für jede Variable individuell einstellbar. Sie können wählen zwischen einer zyklischen Abfrage mit einem einstellbaren Standardzyklus oder einer Reaktion sobald vom System eine Änderung des Variablenwerts festgestellt wurde. Je nach Wahl der Abfrage kann es vorkommen, dass sich die Variable ändert, die Änderung vom System aber nicht wahrgenommen wird. Die Aktion wird in diesem Fall nicht ausgeführt.

Voraussetzung

Die Aktion, die mit einem Trigger verbunden werden soll, muss im Editierfenster geöffnet sein.

Vorgehensweise

1. Betätigen Sie die Schaltfläche in der Symbolleiste "Bearbeiten". Es öffnet sich der Dialog "Eigenschaften".
2. Wählen Sie die Registerkarte Trigger
3. Wählen Sie die Triggerquelle "Variable" und klicken Sie auf die Schaltfläche "Hinzufügen". Es öffnet sich der Dialog "Trigger hinzufügen".



4. Öffnen Sie mit der Schaltfläche ... den Variablenauswahldialog, wählen Sie eine Variable und bestätigen Sie Ihre Auswahl mit "OK"
5. Öffnen Sie im Dialog "Trigger hinzufügen" das Kontextmenü in der Spalte "Standardzyklus" und wählen Sie den gewünschten Überwachungszyklus. "Bei Änderung" bedeutet ständige Überwachung.
6. Wiederholen Sie die Schritte 4 und 5, wenn Sie eine weitere Variable hinzufügen möchten
7. Bestätigen Sie Ihre Einstellungen mit "OK"
8. Schließen Sie den Dialog "Eigenschaften" mit "OK"

Alternative Bedienung

Alternativ können Sie den Dialog "Eigenschaften" auch folgendermaßen öffnen:

Wählen Sie im Menü "Bearbeiten" den Eintrag "Info" oder im Kontextmenü des Editierfensters den Eintrag "Info / Trigger" oder verwenden Sie die entsprechende Tastenkombination.

Im Dialog "Trigger hinzufügen" können Sie alternativ einen Variablennamen auch direkt eintragen und die Variable mit der Schaltfläche "Hinzufügen" in die Spalte "Triggername" aufnehmen. Die Existenz der Variablen wird hierbei allerdings nicht geprüft.

2.13.8.4 So ändern Sie einen Trigger

Einleitung

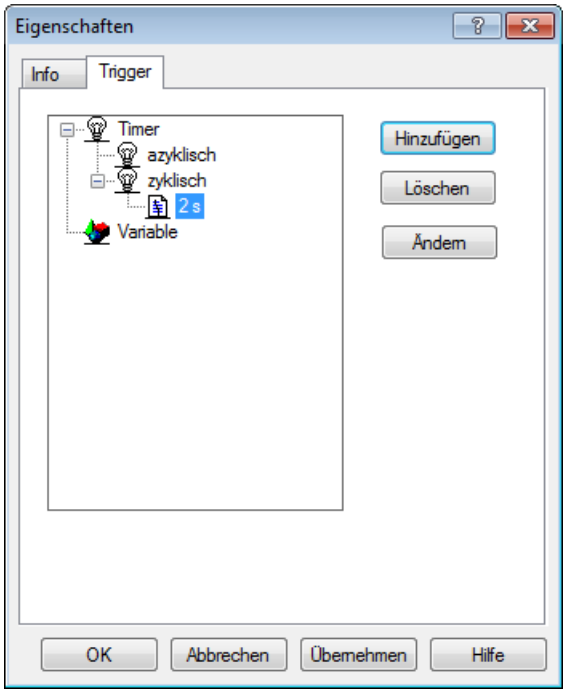
Festgelegte Trigger können jederzeit geändert werden. Dies kann auch während Runtime geschehen.

Voraussetzung

Die betroffene Aktion muss im Editierfenster geöffnet sein.

Vorgehensweise

1. Betätigen Sie die Schaltfläche in der Symbolleiste "Bearbeiten". Es öffnet sich der Dialog "Eigenschaften".
2. Wählen Sie die Registerkarte "Trigger" und markieren Sie den zu verändernden Trigger



3. Öffnen Sie den Dialog "Trigger ändern" mit der Schaltfläche "Ändern"
4. Führen Sie die Änderungen durch
5. Bestätigen Sie Ihre Änderungen mit "OK"
6. Schließen Sie den Dialog "Eigenschaften" mit "OK"

Alternative Bedienung

Alternativ können Sie den Dialog "Eigenschaften" auch folgendermaßen öffnen:

Wählen Sie im Menü "Bearbeiten" den Eintrag "Info" oder im Kontextmenü des Editierfensters den Eintrag "Info / Trigger" oder verwenden Sie die entsprechende Tastenkombination.

2.13.8.5 So löschen Sie einen Trigger

Einleitung

Festgelegte Trigger können jederzeit wieder gelöscht werden. Dies kann auch während Runtime geschehen.

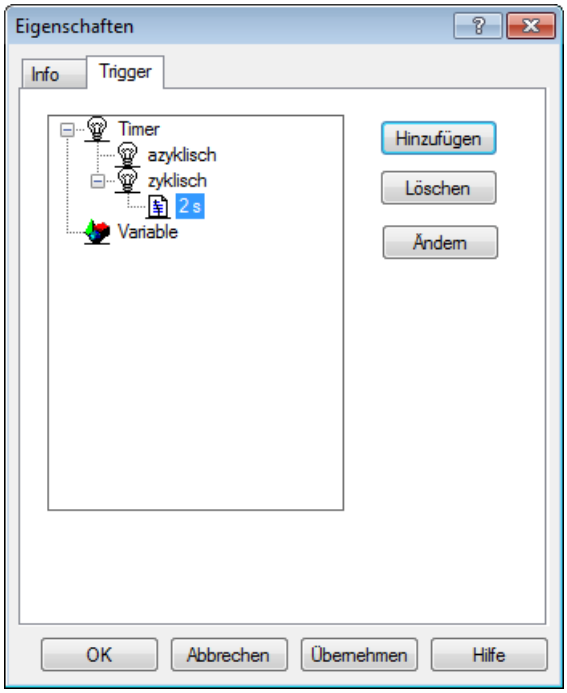
Wird in Runtime ein Trigger gelöscht, dann wirkt sich das erst nach Speichern der Aktion aus.

Voraussetzung

Die betroffene Aktion muss im Editierfenster geöffnet sein.

Vorgehensweise

1. Betätigen Sie die Schaltfläche in der Symbolleiste "Bearbeiten". Es öffnet sich der Dialog "Eigenschaften".
2. Wählen Sie die Registerkarte "Trigger" und markieren Sie den zu löschenden Trigger



3. Löschen Sie den markierten Trigger mit der Schaltfläche "Löschen"
4. Schließen Sie den Dialog "Eigenschaften" mit "OK"

Alternative Bedienung

Alternativ können Sie den Dialog "Eigenschaften" auch folgendermaßen öffnen:

Wählen Sie im Menü "Bearbeiten" den Eintrag "Info" oder im Kontextmenü des Editierfensters den Eintrag "Info / Trigger" oder verwenden Sie die entsprechende Tastenkombination.

2.13.9 So weisen Sie eine Berechtigung zu

Einleitung

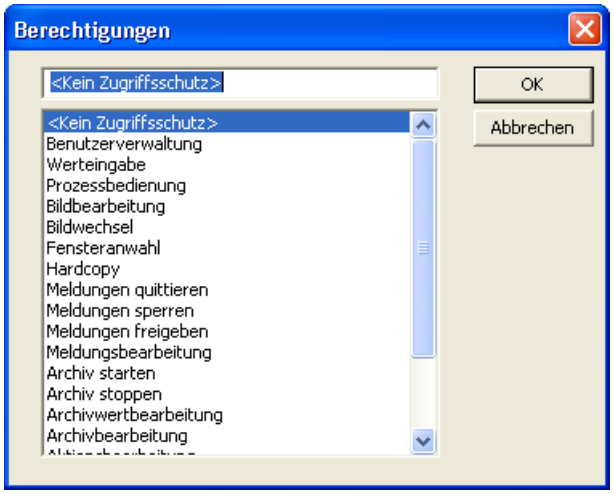
Mit dem Diagnosewerkzeug Global Script - Runtime kann in Runtime auf den Ablauf von Aktionen Einfluss genommen werden. Sie können einer Aktion eine Berechtigung zuweisen. Diese Berechtigung wirkt sich nur auf die Bedienung des Fensters Global Script - Runtime aus.

Voraussetzung

Die betroffene Aktion muss im Editierfenster geöffnet sein.

Vorgehensweise

1. Wählen Sie im Menü "Bearbeiten" den Befehl "Bedienberechtigung".
   Der Dialog "Berechtigungen" wird geöffnet.
2. Wählen Sie eine Berechtigung aus.



3. Bestätigen Sie Ihre Auswahl mit "OK".

Siehe auch

GSC-Runtime (Seite 900)

2.13.10 So exportieren Sie eine Aktion

Einleitung

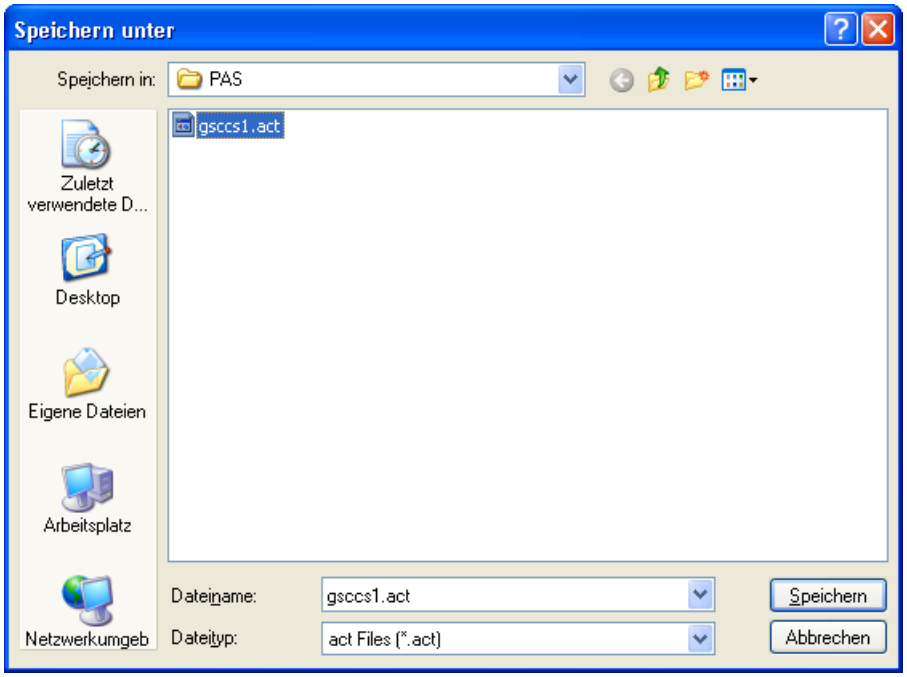
Verwenden Sie Export und Import von Aktionen, um Aktionen in ein anderes Projekt zu bringen. Die mit den Aktionen verbundenen Trigger bleiben dabei erhalten.

Voraussetzung

Die zu exportierende Aktion muss im Editierfenster geöffnet sein.

Vorgehensweise

1. Betätigen Sie die Schaltfläche in der Symbolleiste "Bearbeiten". Es öffnet sich der Dialog "Datei speichern unter".
2. Wählen Sie einen Pfad und einen Dateinamen für die Aktion, die Sie exportieren möchten.



3. Exportieren Sie die Aktion mit der Schaltfläche "Speichern"

Alternative Bedienung

Alternativ können Sie den Export auch folgendermaßen anstoßen:

Wählen Sie im Menü "Bearbeiten" den Eintrag "Exportieren" oder im Kontextmenü des Editierfensters den Eintrag "Aktion exportieren" oder verwenden Sie die entsprechende Tastenkombination.

2.13.11 So importieren Sie eine Aktion

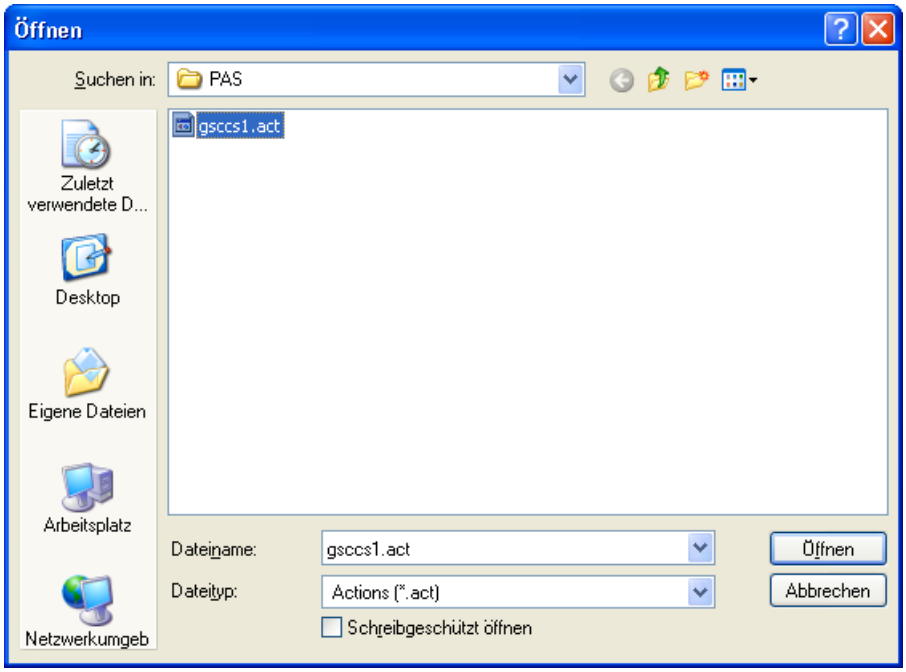
Einleitung

Verwenden Sie Export und Import von Aktionen, um Aktionen in ein anderes Projekt zu bringen. Die mit den Aktionen verbundenen Trigger bleiben dabei erhalten.

Die Aktion im aktiven Editierfenster wird durch die importierte Aktion ersetzt.

Vorgehensweise

1. Betätigen Sie die Schaltfläche in der Symbolleiste "Bearbeiten". Es öffnet sich der Dialog "Öffnen".
2. Wählen Sie Pfad und Dateiname der Aktion, die Sie importieren möchten



3. Importieren Sie die Aktion mit der Schaltfläche "Öffnen"

Alternative Bedienung

Alternativ können Sie den Import auch folgendermaßen anstoßen:

Wählen Sie im Menü "Bearbeiten" den Eintrag "Importieren" oder im Kontextmenü des Editierfensters den Eintrag "Aktion importieren" oder verwenden Sie die entsprechende Tastenkombination.

2.13.12 So benennen Sie eine Aktion um

Einleitung

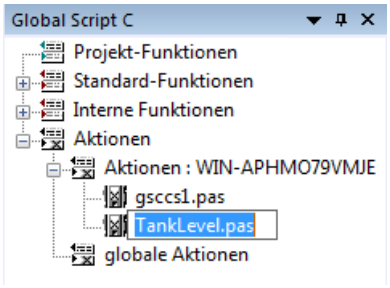
Sie können eine Aktion umbenennen. Die Aktion erhält damit einen anderen Dateinamen.

Voraussetzung

Die Aktion, die Sie umbenennen möchten, darf nicht im Editierfenster geöffnet sein.

Vorgehensweise

1. Öffnen Sie das Kontextmenü der Aktion, die Sie umbenennen möchten
2. Wählen Sie den Eintrag "Umbenennen"
3. Geben Sie den neuen Namen mit Namenserweiterung .pas ein



4. Bestätigen Sie den neuen Namen mit der Taste <Return>

Alternative Bedienung

Alternativ können Sie auch mit zwei aufeinander folgenden Einfachklicks auf den Aktionsnamen das Umbenennen einleiten.

Hinweis

Wenn Sie die Namenserweiterung .pas nicht eintragen, dann bleibt der Aktionsname unverändert.

2.13.13 So verwenden Sie projektfremde Aktionen

Einleitung

Sie haben zwei Möglichkeiten, projektfremde Aktionen in Ihrem Projekt zu verwenden:

- Import von exportierten Aktionen
- Kopieren der Datei mit der gewünschten Aktion in den entsprechenden Pfad Ihres Projekts. Der Pfad für lokale Aktionen ist im Projektpfad das Verzeichnis <Rechnername>\Pas. Der Pfad für globale Aktionen ist im Projektpfad das Verzeichnis Pas.

Damit einkopierte Aktionen im Navigationsfenster angezeigt werden, ist die Anzeige zu aktualisieren. Dies kann durch Beenden und erneutes Starten von Global Script geschehen.

Bei aktivem Runtime werden die so übernommenen Aktionen erst ausgeführt, nachdem sie im Global Script Editor geöffnet und dann gespeichert wurden.

Hinweis

Aktionen können Aufrufe von Projekt- und Standard-Funktionen enthalten. Diese können ihrerseits wieder Aufrufe von Projekt- und Standard-Funktionen enthalten usw. Bei der Übernahme von projektfremden Aktionen ist deshalb sicherzustellen, dass im aktuellen Projekt alle hierfür notwendigen Funktionen vorhanden sind.

Besondere Aufmerksamkeit ist geboten, wenn es sich um Aktionen handelt, die von einem anderen Rechner bezogen werden. Da Standard-Funktionen anwenderspezifisch verändert werden können, besteht die Möglichkeit, dass die in der Aktion aufgerufenen Standard-Funktionen auf dem Quell-Rechner eine andere Funktionalität besitzen wie die gleichnamigen Standard-Funktionen auf dem Ziel-Rechner.

Siehe auch

So importieren Sie eine Aktion (Seite 895)

So exportieren Sie eine Aktion (Seite 894)

2.14 Laufzeitverhalten von Aktionen

2.14.1 Laufzeitverhalten von Aktionen

Analyse des Laufzeitverhaltens

WinCC stellt drei Werkzeuge zur Verfügung, um das Laufzeitverhalten in Aktionen zu analysieren. Dies sind die Applikationsfenster GSC-Runtime und GSC-Diagnose sowie die Anwendung apdiag.exe.

Um die Applikationsfenster GSC-Runtime und GSC-Diagnose zu verwenden, werden sie in ein Prozessbild eingefügt. Dies kann ein eigens zu Diagnosezwecken entworfenes Prozessbild sein. Es wird in Runtime aufgerufen.

Mit diesen Applikationsfenstern werden folgende unterschiedliche Strategien verfolgt:

- GSC-Runtime gibt Auskunft über das dynamische Verhalten aller (Global Script-) Aktionen, ermöglicht den individuellen Start sowie das An- und Abmelden jeder einzelnen Aktion und bietet den Einsprung in den Global Script Editor, während Runtime aktiv ist
- GSC-Diagnose gibt die in den Aktionen enthaltenen printf-Anweisungen in der zeitlichen Reihenfolge ihres Aufrufs aus. Das gilt auch für printf-Anweisungen in Funktionen, die in Aktionen aufgerufen werden. Durch gezielten Einsatz von printf-Anweisungen, beispielsweise zur Ausgabe von Variablenwerten, lässt sich so der Ablauf von Aktionen und den darin aufgerufenen Funktionen verfolgen. Auch Fehlersituationen, die den Aufruf der Funktion OnErrorExecute zur Folge haben, werden im Diagnosefenster angezeigt.

Hinweis

Beachten Sie bei C-Skript-Dynamisierungen mit Zugriff auf Bildobjekte, dass ein in Abarbeitung befindliches Skript durch die Abwahl des Bildes nicht automatisch beendet wird.

Dies kann dazu führen, dass der Zugriff auf im Skript angesprochene Objekte fehlschlägt. Z. B. wenn die Eigenschaften vom Typ "Text" gelesen werden und die zurückgelieferte Werte in nachfolgenden String-Operationen verändert oder weiterverarbeitet werden.

Siehe auch

GSC-Diagnose (Seite 905)

GSC-Runtime (Seite 900)

2.14.2 GSC-Runtime

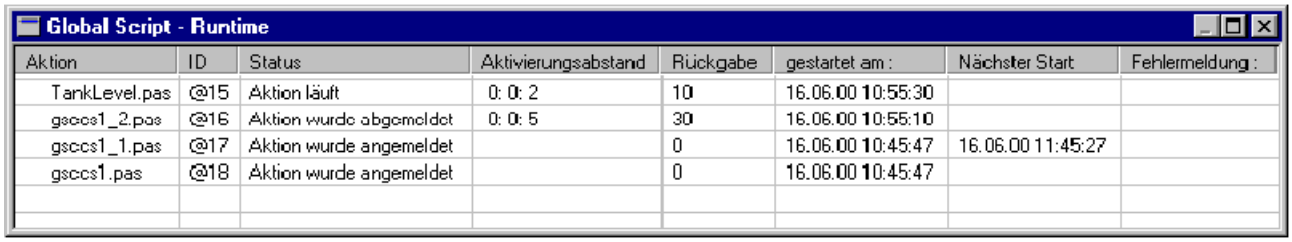
2.14.2.1 GSC-Runtime

Fenster GSC-Runtime

GSC-Runtime ist ein Fenster, das das dynamische Verhalten aller (Global Script-) Aktionen in Runtime anzeigt. Darüber hinaus ermöglicht GSC-Runtime, Einfluss auf die Ausführung jeder einzelnen Aktion zu nehmen und bietet den Einsprung in den Global Script Editor, während Runtime aktiv ist.

Folgende Informationen werden ausgegeben:

- Aktionsname: Der Name der Aktion
- ID: Die Aktions ID. Sie wird systemintern verwendet und beispielsweise beim Autreten eines Fehlers in einer Aktion durch die Funktion OnErrorExecute zusammen mit der Fehlerbeschreibung ausgegeben. GSC-Runtime liefert zur Aktions ID den zugehörigen Aktionsnamen. Die Verbindung zwischen ID und Aktionsname ist nur solange gültig, bis Runtime beendet wird oder bei aktivem Runtime eine Aktion gespeichert wird.
- Status: Gibt Auskunft über den momentanen Status der Aktion. Die möglichen Stati entnehmen Sie bitte unten stehender Tabelle
- Aktivierungsabstand: Die Zeit in der Form Stunde:Minute:Sekunde, die zwischen zwei Aufrufen der Aktion verstrichen ist
- Rückgabe: Der Rückgabewert der Aktion
- gestartet am: Datum und Uhrzeit des aktuellen Starts der Aktion
- Nächster Start: Datum und Uhrzeit des nächsten Starts der Aktion
- Fehlermeldung: enthält den Fehlertext im Falle eines Fehlers



Aktion	ID	Status	Aktivierungsabstand	Rückgabe	gestartet am :	Nächster Start	Fehlermeldung :
TankLevel.pas	@15	Aktion läuft	0: 0: 2	10	16.06.00 10:55:30		
gsccs1_2.pas	@16	Aktion wurde abgemeldet	0: 0: 5	30	16.06.00 10:55:10		
gsccs1_1.pas	@17	Aktion wurde angemeldet		0	16.06.00 10:45:47	16.06.00 11:45:27	
gsccs1.pas	@18	Aktion wurde angemeldet		0	16.06.00 10:45:47		

Zustände von Aktionen

Die möglichen Stati einer Aktion sind:

- Aktion wurde angemeldet
- Aktion wurde abgemeldet
- Aktion wurde gestoppt
- Aktion läuft

- Fehler beim Anmelden der Aktion!
- Fehler beim Ausführen der Aktion!

Fehlermeldungen

Folgende Fehlermeldungen sind möglich:

- Kein Fehler aufgetreten
- Applikation ist mit der Aktionssteuerung schon verbunden. Kein weiterer Verbindungsaufbau möglich.
- Es besteht keine Verbindung mit der Aktionssteuerung. Möglicherweise hat kein Verbindungsaufbau stattgefunden.
- Bei der Interprozesskommunikation ist ein Fehler aufgetreten. Die Fehlerursache ist unbekannt.
- Nicht näher beschriebener Fehler.
- Die Parameterversorgung ist fehlerhaft. Eventuell wurden benötigte Parameter nicht versorgt.
- Aktionssteuerung ist nicht gestartet. Überprüfen Sie Bitte ob WinCC gestartet wurde.
- Es hat ein Zeitüberlauf stattgefunden. Sie sollten die Verbindung überprüfen oder die Zeitüberwachungszeit heraufsetzen.
- Aktionssteuerung wurde beendet.
- Servicekanal konnte nicht installiert werden.
- Für den Auftrag EndAct wurde eine nicht bekannte Auftragsnummer verwendet.
- Die Aktion konnte nicht fehlerfrei ausgeführt werden. Die Rueckgabeergebnisse sind ungültig.
- Es ist ein Fehler innerhalb der Serverapplikation aufgetreten.
- Die maximale Anzahl von Verbindungen zur Aktionssteuerung ist erreicht.
- Transaktion ist nicht bekannt. Es wurde versucht eine Transaktion zu beenden, die vorher nicht angemeldet wurde.
- Eine precompiled Headerdatei kann nicht aus einer precompiled Headerdatei generiert werden.
- Auf die Aktion kann nicht zugegriffen werden. Das Modul wird momentan benützt.
- Das Programm ist ungültig.
- Die Aktion ist ungültig.
- Datei konnte von der Aktionssteuerung nicht angelegt werden.
- Der Aktionsinterpreter hat nicht mehr genügend Speicher zu Verfügung.
- Dateiformat ist für die Aktionssteuerung ungültig.
- Datei konnte von der Aktionssteuerung nicht geöffnet werden.
- Programm ist momentan vom der Aktionssteuerung gesperrrt. Kein weiterer Zugriff möglich.
- Aktion wurde bereits der Aktionssteuerung zu Bearbeitung gegeben.

- In der Aktion ist ein Konflikt mit einer anderen Aktion aufgetreten.
- Aktion wurde von der Aktionssteuerung nicht gefunden.
- Funktion wurde von der Aktionssteuerung nicht gefunden.
- Angegebene Zeileninformation ist ungültig.
- Angegebenes Symbol ist außerhalb des Gültigkeitsbereichs.
- Der übergebene Speicher ist für den Aktionsinterpreter zu klein.
- Der angegebene Typ ist dem Aktionsinterpreter unbekannt.
- Das angegebene Symbol wurde nicht gefunden.
- Lade Projektfunktionen.
- Im Aktionsinterpreter ist ein Stapelüberlauf während der Ausführung aufgetreten. Weitere Ausführung der Aktion wird abgebrochen.
- Innerhalb der Ausführung einer Aktion ist ein Division durch 0 aufgetreten. Aktion wird abgebrochen.
- Innerhalb der Aktion wurde während der Ausführung auf ein nicht vorhandenes Symbol referenziert.
- Innerhalb der Aktion wurde während der Ausführung auf nicht definierten Speicher zugegriffen.
- Der Aktionsinterpreter lief auf einen Breakpoint.
- Der Aktionsinterpreter wurd im Debugger um einen Bearbeitungsschritt fortgesetzt.
- Aktion beinhaltet keinen Interpretercode.
- Aktion besitzt falsches Datenformat.
- Der Rückgabewert der Aktion kann nicht als Variant dargestellt werden.
- Der Applikation steht nicht mehr Speicher zu Verfügung, um diese Operation auszuführen.
- Innerhalb der Transaktion ist ein Fehler aufgetreten. Nähere Fehlerbeschreibung bei den AP_ACT_KEYs.
- Bei den Ausührung der Aktion ist ein Fehler aufgetreten. Nähere Beschreibung in den AP_ACT_KEYs.
- Bei den Ausührung der Aktion ist ein Fehler aufgetreten. Nähere Beschreibung in den AP_ACT_KEYs.
- Für das bestehende Datenformat gibt es keine Updatemöglichkeit. Aktion kann nicht gelesen werden.

Kontextmenü zu Aktionen

Im Kontextmenü zu jeder Aktion finden Sie folgende Funktionen:

- abmelden: Die betreffende Aktion wird nach Ende der aktuellen Ausführung nicht mehr ausgeführt
- anmelden: Die betreffende Aktion wird mit dem nächsten Trigger wieder ausgeführt

- starten: Die betreffende Aktion wird einmal ausgeführt
- bearbeiten: Die betreffende Aktion wird im Global Script Editor zur Bearbeitung geöffnet. Runtime bleibt dabei aktiv. Wird die bearbeitete Aktion übersetzt (falls notwendig) und gespeichert, dann werden die Änderungen sofort vom Runtime System übernommen

Die Möglichkeit, das Kontextmenü zu öffnen, können Sie für jede Aktion durch die Vergabe einer Berechtigung steuern.

Damit Sie GSC-Runtime verwenden können, ist ein Applikationsfenster vom Typ GSC-Runtime in ein Prozessbild einzufügen. Mit den Attributen von GSC-Runtime können Sie das Erscheinungsbild des GSC-Runtime-Fensters steuern.

Hinweis

Die Aktualisierung des GSC-Runtime-Fensters erhöht die Systembelastung. Die Systembelastung ist abhängig davon, wie viele Aktionen im Fenster sichtbar sind. Die Systembelastung kann reduziert werden, wenn das Fenster in seiner Höhe verkleinert wird, sodass weniger Zeilen sichtbar sind.

Siehe auch

So bearbeiten Sie eine Aktion (Seite 905)

Die Attribute von GSC-Runtime (Seite 904)

So bringen Sie GSC-Runtime in ein Prozessbild (Seite 903)

So weisen Sie eine Berechtigung zu (Seite 893)

2.14.2.2 So bringen Sie GSC-Runtime in ein Prozessbild

Einleitung

Damit Sie GSC-Runtime verwenden können, ist GSC-Runtime in ein Prozessbild einzufügen. Dieses Prozessbild kann ein vorhandenes Bild oder auch ein Bild sein, das eigens Diagnosezwecken dient. GSC-Runtime lässt sich nicht direkt in das Prozessbild einfügen, sondern wird als Anwendung in ein Applikationsfenster eingefügt. Das Applikationsfenster selbst ist dabei Bestandteil des Prozessbilds. Die beschriebenen Maßnahmen sind im Graphics Designer vorzunehmen.

Voraussetzung

Der Graphics Designer ist gestartet und das Prozessbild ist geöffnet.

Vorgehensweise

1. Wählen Sie aus der Objektpalette "Smart-Objekte\Applikationsfenster"
2. Ziehen Sie in der Zeichenfläche das Applikationsfenster auf
3. Wählen Sie aus dem Dialog "Fensterinhalt" den Eintrag "Global Script"
4. Bestätigen Sie Ihre Wahl mit "OK"
5. Wählen Sie aus dem Dialog "Vorlage" den Eintrag "GSC-Runtime"
6. Bestätigen Sie Ihre Auswahl mit "OK"

Siehe auch

Die Attribute von GSC-Runtime (Seite 904)

2.14.2.3 Die Attribute von GSC-Runtime

Erscheinungsbild des GSC-Runtime-Fensters

GSC-Runtime besitzt Attribute, die das Erscheinungsbild des GSC-Runtime-Fensters in Runtime beeinflussen. Dies sind die Geometrieattribute und insbesondere folgende Attribute:

- Anzeige: Mit diesem Attribut legen Sie fest, ob das Fenster sichtbar oder unsichtbar ist. Das Attribut ist mit dem Namen Visible dynamisierbar
- Größe veränderbar: Mit diesem Attribut bestimmen Sie, ob die Größe des Fensters in Runtime verändert werden kann
- Verschiebbar: Mit diesem Attribut legen Sie fest, ob das Fenster in Runtime verschiebbar ist
- Rahmen: Mit diesem Attribut bestimmen Sie, ob das Fenster einen Rahmen erhält. Hat das Fenster einen Rahmen, dann ist es in Höhe und Breite in Runtime veränderbar
- Titel: Hiermit legen Sie fest, ob das Fenster eine Titelleiste besitzt
- Maximierbar: Mit diesem Attribut bestimmen Sie, ob die Titelleiste die Schaltfläche zum Maximieren des Fensters enthält
- Schließbar: Mit diesem Attribut bestimmen Sie, ob die Titelleiste die Schaltfläche zum Schließen des Fensters enthält
- Vordergrund: Hiermit legen Sie fest, ob das Fenster immer im Vordergrund ist

Die Attribute werden im Graphics Designer angezeigt und sind dort einstellbar.

2.14.2.4 So bearbeiten Sie eine Aktion

Einleitung

Alle in Ihrem Projekt enthaltenen Aktionen werden in je einer Zeile des GSC-Runtime-Fensters angezeigt. Sie können eine Aktion aus dem GSC-Runtime-Fenster heraus öffnen und mit dem Global Script Editor bearbeiten. Nach dem Speichern wird die bearbeitete Aktion in Runtime verwendet.

Vorgehensweise

1. Öffnen Sie das Kontextmenü zur betreffenden Aktion
2. Wählen Sie den Eintrag "bearbeiten"

Siehe auch

Aktionen erstellen und bearbeiten (Seite 876)

2.14.3 GSC-Diagnose

2.14.3.1 GSC-Diagnose

Funktionsbeschreibung

GSC-Diagnose gibt die in den Aktionen enthaltenen printf-Anweisungen in der zeitlichen Reihenfolge ihres Aufrufs im Diagnosefenster aus. Das gilt auch für printf-Anweisungen in Funktionen, die in Aktionen aufgerufen werden. Durch gezielten Einsatz von printf-Anweisungen, beispielsweise zur Ausgabe von Variablenwerten, lässt sich so der Ablauf von

Aktionen und den darin aufgerufenen Funktionen verfolgen. Auch Fehlersituationen, die den Aufruf der Funktion OnErrorExecute zur Folge haben, werden im Diagnosefenster angezeigt.

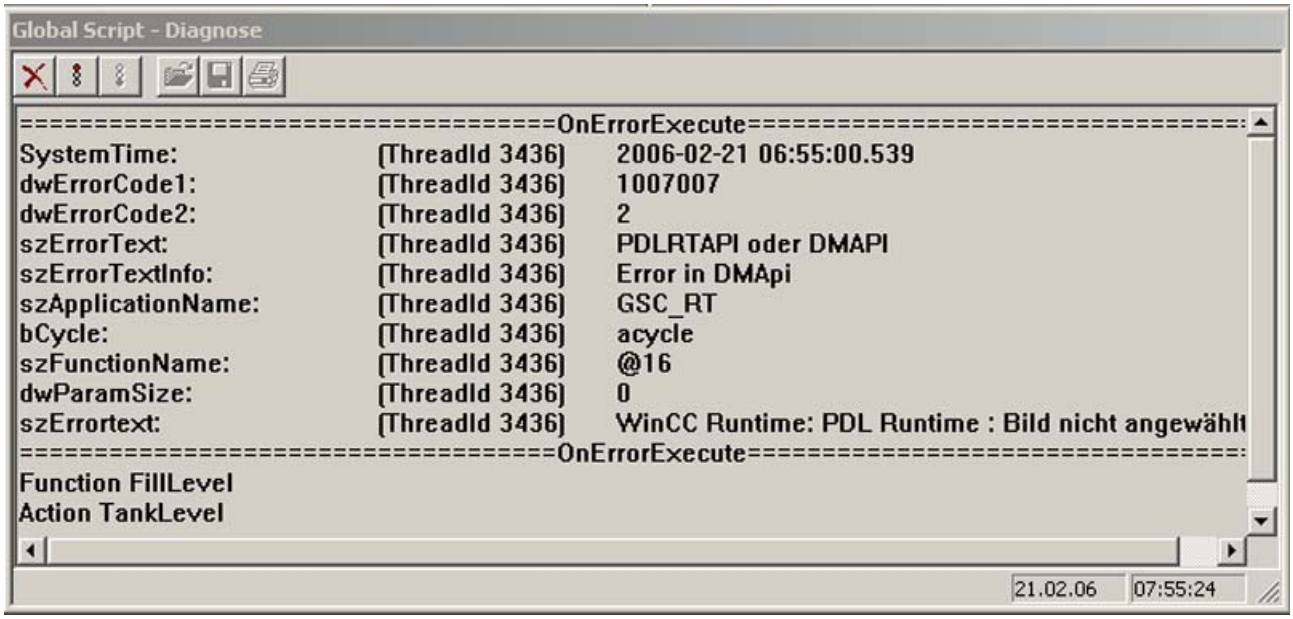


Bild 2-1 Das Diagnosefenster von GSC-Diagnose

Damit Sie GSC-Diagnose verwenden können, ist ein Applikationsfenster vom Typ GSC-Diagnose in ein Prozessbild einzufügen. Mit den Attributen von GSC-Diagnose können Sie das Erscheinungsbild des GSC-Diagnosefensters steuern.

Bei einem Bildwechsel wird der Inhalt des GSC-Diagnosefensters gelöscht.

Hinweis

Ein printf() kann maximal 360 Zeichen enthalten.

Siehe auch

Die Symbolleiste von GSC-Diagnose (Seite 908)

Die Attribute von GSC-Runtime (Seite 904)

So bringen Sie GSC-Diagnose in ein Prozessbild (Seite 906)

2.14.3.2 So bringen Sie GSC-Diagnose in ein Prozessbild

Einleitung

Damit Sie GSC-Diagnose verwenden können, ist GSC-Diagnose in ein Prozessbild einzufügen. Dieses Prozessbild kann ein vorhandenes Bild oder auch ein Bild sein, das eigens Diagnosezwecken dient. GSC-Diagnose lässt sich als Applikation nicht direkt in das Prozessbild einfügen, sondern wird als Anwendung in ein Applikationsfenster eingefügt. Das

Applikationsfenster selbst ist dabei Bestandteil des Prozessbilds. Die beschriebenen Maßnahmen sind im Graphics Designer vorzunehmen.

Voraussetzung

Der Graphics Designer ist gestartet und das Prozessbild ist geöffnet.

Vorgehensweise

1. Wählen Sie aus der Objektpalette "Smart-Objekte\Applikationsfenster"
2. Ziehen Sie in der Zeichenfläche das Applikationsfenster auf
3. Wählen Sie aus dem Dialog "Fensterinhalt" den Eintrag "Global Script"
4. Bestätigen Sie Ihre Wahl mit "OK"
5. Wählen Sie aus dem Dialog "Vorlage" den Eintrag "GSC-Diagnose"
6. Bestätigen Sie Ihre Auswahl mit "OK"

Siehe auch

Die Attribute von GSC-Runtime (Seite 904)

Die Symbolleiste von GSC-Diagnose (Seite 908)

2.14.3.3 Die Attribute von GSC-Diagnose

Attribute der GSC-Diagnose

GSC-Diagnose besitzt Attribute, die das Erscheinungsbild des GSC-Diagnosefensters in Runtime beeinflussen. Dies sind die Geometrieattribute und insbesondere folgende Attribute:

- Anzeige: Mit diesem Attribut legen Sie fest, ob das Fenster sichtbar oder unsichtbar ist. Das Attribut ist mit dem Namen Visible dynamisierbar
- Größe veränderbar: Mit diesem Attribut bestimmen Sie, ob die Größe des Fensters in Runtime verändert werden kann
- Verschiebbar: Mit diesem Attribut legen Sie fest, ob das Fenster in Runtime verschiebbar ist
- Rahmen: Mit diesem Attribut bestimmen Sie, ob das Fenster einen Rahmen erhält. Hat das Fenster einen Rahmen, dann ist es in Höhe und Breite in Runtime veränderbar
- Titel: Hiermit legen Sie fest, ob das Fenster eine Titelleiste besitzt
- Maximierbar: Mit diesem Attribut bestimmen Sie, ob die Titelleiste die Schaltfläche zum Maximieren des Fensters enthält
- Schließbar: Mit diesem Attribut bestimmen Sie, ob die Titelleiste die Schaltfläche zum Schließen des Fensters enthält
- Vordergrund: Hiermit legen Sie fest, ob das Fenster immer im Vordergrund ist

Die Attribute werden im Graphics Designer angezeigt und sind dort einstellbar.

2.14.3.4 Die Symbolleiste von GSC-Diagnose

Funktionen der Symbolleiste

Die Symbolleiste von GSC-Diagnose erlaubt die Steuerung der Ausgabe im Diagnosefenster sowie das Speichern, Drucken und Öffnen des Fensterinhalts.

Die Symbolleiste enthält Schaltflächen mit folgenden Funktionen:

Schaltfläche	Funktion
	löscht den Inhalt des Fensters
	hält die Aktualisierung des Fensters an
	nimmt die Aktualisierung des Fensters wieder auf
	öffnet eine Textdatei im Fenster
	speichert den Fensterinhalt in einer Textdatei
	druckt den Inhalt des Fensters

2.15 ANSI-C Funktionsbeschreibungen

2.15.1 lpszPictureName

Überblick

"lpszPictureName" ist der Name des Bildes.

Wenn Sie in WinCC eine Aktion an einer Eigenschaft oder am Ereignis "Mausklick" projektieren, wird in der Aktion als "lpszPictureName" der Name des Bildes geliefert. Der Bildname hat dabei den folgenden Aufbau:

<GRUNDBILDNAME>.<BILDFENSTERNAME>:<BILDNAME>. ... .<Bildfenstername>:<Bildname>.

Dabei werden der "GRUNDBILDNAME" und der "BILDNAME" ohne die Dateierweiterung ".PDL" geliefert.

Damit können Sie feststellen, welchen Bildpfad das Objekt hat. Sie können auch gezielt bestimmte Bildfenster ansprechen, wenn z.B. ein Prozessbild mehr als einmal geöffnet ist.

Hinweis

Den Text in "lpszPictureName" dürfen Sie nicht verändern, auch nicht über die Funktion "strcat".

2.15.2 Standard Funktionen

2.15.2.1 Standard-Funktionen - Kurzbeschreibung

Das System stellt Standard-Funktionen zur Verfügung. Diese Funktionen können Sie ändern und so Ihren Bedürfnissen anpassen. Darüber hinaus können Sie Standard-Funktionen selbst erstellen.

Das Basissystem stellt Ihnen Standard-Funktionen zur Verfügung. Sie sind in folgende Funktionsgruppen eingeteilt:

- Alarm
- Graphics
- Report
- TagLog
- WinCC
- Windows

Im Verzeichnis "Obsolete functions" befinden sich Standard-Funktionen, die für die Steuerung der Controls vor WinCC V7 verwendet wurden.

Falls die entsprechenden Optionen installiert sind, stehen zusätzlich folgende Funktionsgruppen zur Verfügung:

- Options
- Split Screen Manager
- userarc (Anwenderarchive)

2.15.2.2 Alarm

AcknowledgeMessage

Funktion

Quittiert im Meldesystem die Meldung mit der Nummer, die als Parameter übergeben wird.

Syntax

void AcknowledgeMessage (DWORD MsgNr)

Parameter

MsgNr

Zu quittierende Meldung

Hinweis

Stellen Sie sicher, dass zu der übergebenen Meldungsnummer eine projektierte Meldung existiert.
Wenn Sie die Funktion auf einem Client mit eigenem Projekt nutzen wollen, muss auf dem Client ein Standardserver für Alarme projektiert sein.

Siehe auch

Beispiel AcknowledgeMessage

AXC_SetFilter

Funktion

Externe Meldefensterbedienung

Diese Funktion setzt einen Filter für das WinCC Alarm Control, um eine Teilmenge der vorhandenen Meldungen entsprechend dem Filterkriterium anzuzeigen.

Syntax

BOOL AXC_SetFilter(char* lpszPictureName, char* lpszObjectName,
LPMSG_FILTER_STRUCT lpMsgFilter, LPCMN_ERROR, lpError)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Namen WinCC Alarm Control

lpMsgFilter

Zeiger auf die Struktur mit dem Filterkriterium

lpError

Zeiger auf die Struktur der Fehlerbeschreibung

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Strukturdefinition CMN_ERROR

Beispiel ResetFilter

Beispiel AXC_SetFilter

Strukturdefinition MSG_FILTER_STRUCT

GCreateMyOperationMsg

Funktion

Die Standardfunktion "GCreateMyOperationMsg" ermöglicht Ihnen, eine eigene Bedienmeldung im Meldesystem auszulösen. Die Meldung mit der Meldenummer "dwMsgNum" müssen Sie bereits als Bedienmeldung projektiert haben.

Syntax

int GCreateMyOperationMsg(DWORD dwFlags, DWORD dwMsgNum, char* lpszPictureName, char* lpszObjectName, DWORD dwMyTextID, double doValueOld, double doValueNew, char* pszComment)

Parameter

dwFlags

Die Ausprägung der Meldung können Sie mit dem Parameter "dwFlags" auswählen.

Name	Wert	Bedeutung
FLAG_COMMENT_PARAMETER	0x00000001	Der Text wird direkt als Kommentar in die Meldung in Runtime eingetragen, ohne eigenen Kommentardialog. Der Pointer auf Kommentar muss ungleich "NULL" sein.
FLAG_COMMENT_DIALOG	0x00000003	Ein Kommentar-Dialog wird aufgeblendet. Der dort eingegebene Kommentar wird in die Meldung übernommen.
FLAG_TEXTID_PARAMETER	0x00000100	Die Text-ID eines Texts aus der TextLibrary wird als Prozessbegleitwert der Meldung mitgegeben.

dwMsgNum

WinCC-Meldenummer einer selbst angelegten Bedienmeldung.

lpszPictureName

Zeiger auf den Bildnamen des Bildes, aus dem die Funktion aufgerufen wird.

lpszObjectName

Zeiger auf den WinCC-Variablennamen, zu dem die Altwerte und Neuwerte gehören.

Der Name wird als Instanzname der Bedienmeldung weitergereicht und in den Prozessbegleitwert "1" eingetragen.

dwMyTextID

Text-ID eines Texts aus der TextLibrary.

Bei gesetztem "FLAG_TEXTID_PARAMETER" wird die Text-ID als numerischer Prozessbegleitwert "8" der Meldung mitgegeben und als Zahl im Prozesswertblock 8

angezeigt. Damit in der Meldung der sprachabhängige Text aus der TextLibrary angezeigt wird, müssen Sie im Meldetextblock die Formatanweisung "@8%s@" eingeben.

doValueOld

Numerischer Altwert der WinCC-Variablen mit dem in "lpszObjectName" angegebenen Namen. "doValueOld" wird in den Prozessbegleitwert "2" der Meldung eingetragen.
Die Funktion selbst hat keine Möglichkeit, einen Variablenwert vor der Aktion auszulesen. Verwenden Sie dazu die verfügbaren "GetTag..."-Funktionen.

doValueNew

Numerischer Neuwert der WinCC-Variablen mit dem in "lpszObjectName" angegebenen Namen. "doValueNew" wird in den Prozessbegleitwert "3" der Meldung eingetragen.
Die Funktion selbst hat keine Möglichkeit, einen Variablenwert nach der Aktion auszulesen. Verwenden Sie dazu die verfügbaren "GetTag..."-Funktionen.

pszComment

Kommentartext oder leerer String.

Bei gesetztem "FLAG_COMMENT_PARAMETER" wird der Text direkt als Kommentar in die Meldung in Runtime eingetragen. Dazu braucht die Meldung keinen eigenen Kommentardialog.

Rückgabewert

Wert	Bedeutung
0	Die Funktion wurde fehlerfrei durchlaufen.
-101	Die Meldungsbearbeitung konnte nicht gestartet werden.
-201	Beim Aufruf der Funktion "MSRTGetComment()" ist ein Fehler aufgetreten.
-301	Beim Aufruf der Funktion "MSRTCreateMsgInstanceWithComment()" ist ein Fehler aufgetreten.

Hinweis

Stellen Sie sicher, dass für die Funktion "GCreateMyOperationMsg" ausschließlich Bedienmeldungen verwendet werden. Die Verwendung von Meldungen anderer Meldeklassen ist nicht zulässig.

Beachten Sie bei der Verwendung der Funktion bei einem Client die Rolle des Standardservers. Nähere Informationen erhalten Sie im Kapitel "Client-Projektierung".

GMsgFunction

Funktion

Diese Funktion liefert die Meldungsdaten.

Sie stellt eine globale Funktion für Einzelmeldungen dar. Sie wird bei jeder Meldung aufgerufen, bei der der Parameter "Löst eine Aktion aus" gesetzt wurde.

Eine Auswertung der Meldungsdaten wird am besten in einer Projektfunktion vorgenommen, welche dann aus GMsgFunction aufgerufen wird.

Syntax

BOOL GMsgFunction(char* pszMsgData)

Parameter

pszMsgData

Zeiger auf einen String, dessen Daten mit scanf auf die Struktur MSG_RTDATA_STRUCT abgebildet werden.

Der String "MSG_RTDATA_STRUCT" enthält folgende Daten, die mit "#" voneinander getrennt sind:

1. Telegrammzeit
2. Prozesswerte
3. Instanz
4. Benutzer
5. Rechner
6. Gekommenzeit im Format "yyyy.mm.dd, hh:mm:ss.mmm"

Hinweis

Der Wert "Instanz" des Strings "MSG_RTDATA_STRUCT" wird nur versorgt, wenn eine Instanzmeldung ausgelöst wurde.

Die Werte "Benutzer" und "Rechner" des Strings "MSG_RTDATA_STRUCT" werden nur versorgt, wenn beim Erzeugen der Meldung mit dem gleichen Aufruf ein Kommentar mitgegeben wurde.

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Beachten Sie, dass veränderte Standard-Funktionen durch eine WinCC Installation überschrieben werden und somit die Änderungen verloren gehen.

Siehe auch

Strukturdefinition MSG_RTDATA_STRUCT

2.15.2.3 Graphics

Graphics - Kurzbeschreibung

Die Gruppe Graphics enthält Funktionen für die Programmierung des Graphic Systems.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetLinkedVariable

Funktion

Liefert den Namen der Variablen, die mit einer bestimmten Objekteigenschaft verknüpft ist.

Syntax

```
char* GetLinkedVariable(char* lpszPictureName, char* lpszObjectName, char*
lpszPropertyName);
```

Parameter

lpszPictureName

Zeiger auf das Bild

lpszObjectName

Zeiger auf das Objekt

lpszPropertyName

Zeiger auf die Objekteigenschaft

Rückgabewert

Zeiger auf den Namen der Variablen, die mit einer bestimmten Objekteigenschaft verknüpft ist

Siehe auch

Beispiel GetLinkedVariable

GetLocalPicture

Funktion

Liefert einen Zeiger auf den Namen des Bildes. Der Bildname ist dabei der Dateiname ohne die Erweiterung ".PDL".

Syntax

```
char* GetLocalPicture(char* lpszPictureName);
```

Parameter

lpszPictureName

Zeiger auf das Bild

Rückgabewert

Zeiger auf den Namen des Bildes.

Hinweis

Der übergebene Aufrufparameter lpszPictureName muss den Aufbau haben, wie ihn das Grafiksystem für die Bildpfade bildet:

<Grundbildname>.<Bildfenstername>:<Bildname>. ... .<Bildfenstername>:<Bildname>,

wobei <Grundbildname> und <Bildname> ohne die Dateierweiterung ".PDL" eingehen.

Beispiel:

In einem Grundbild "AAA" liegt ein Bildfenster "bbb", in dem ein Bild "CCC" aufgerufen wird, das seinerseits ein Bildfenster "ddd" enthält, in dem ein Bild "EEE" aufgerufen wird.

Dann liefert der Funktionsaufruf

```
GetLocalPicture(lpszPictureName)
```

den Zeiger auf den Bildnamen:

"EEE" zurück, wenn die Funktion im Bild "EEE" aufgerufen wird;

"CCC" zurück, wenn die Funktion im Bild "CCC" aufgerufen wird;

"AAA" zurück, wenn die Funktion im Bild "AAA" aufgerufen wird.

Siehe auch

Beispiel GetLocalPicture

GetParentPicture

Funktion

Liefert einen Zeiger auf den Namen des Bildes. Der Bildname ist dabei der Dateiname ohne die Erweiterung ".PDL".

Syntax

char* GetParentPicture(char* lpszPictureName);

Parameter

lpszPictureName

Zeiger auf das Bild

Rückgabewert

Namen des aktuellen Bildes, wenn die Funktion im Grundbild aufgerufen wird

Namenspfad des übergeordneten Bildes, wenn die Funktion in einem Bildfenster aufgerufen wird

Hinweis

Der übergebene Aufrufparameter lpszPictureName muss den Aufbau haben, wie ihn das Grafiksystem für die Bildpfade bildet:

<Grundbildname>.<Bildfenstername>:<Bildname>. ... .<Bildfenstername>:<Bildname>,

wobei <Grundbildname> und <Bildname> ohne die Dateierweiterung ".PDL" eingehen.

Siehe auch

Beispiel GetParentPicture

GetParentPictureWindow

Funktion

Liefert einen Zeiger auf den Namen des Bildfensters.

Syntax

```
char* GetParentPictureWindow(char* lpszPictureName);
```

Parameter

lpszPictureName

Zeiger auf das Bild

Rückgabewert

Zeiger auf den Namen des Bildfensters, wenn die Funktion in einem Bild aufgerufen wird, welches in einem Bildfenster eines übergeordneten Bildes angezeigt wird

Aufrufparameter lpszPictureName unverändert, wenn die Funktion im Grundbild aufgerufen wird

Hinweis

Der übergebene Aufrufparameter lpszPictureName muss den Aufbau haben, wie ihn das Grafiksystem für die Bildpfade bildet:

<Grundbildname>.<Bildfenstername>:<Bildname>. ... .<Bildfenstername>:<Bildname>,

wobei <Grundbildname> und <Bildname> ohne die Dateierweiterung ".PDL" eingehen.

Beispiel:

In einem Grundbild "Bild_1" liegt ein Bildfenster "Bildfenster_1", in dem ein Bild "Bild_2" aufgerufen wird.

Im Bild "Bild_2" liegt ein Bildfenster "Bildfenster_2", in dem ein Bild "Bild_3" aufgerufen wird.

Dann liefert der Funktionsaufruf

```
GetParentPictureWindow(lpszPictureName)
```

den Zeiger auf den Bildfensternamen:

"Bild_2" zurück, wenn die Funktion im Bild "Bild_3" aufgerufen wird;

"Bildfenster_1" zurück, wenn die Funktion im Bild "Bild_2" aufgerufen wird;

"Bild_1" zurück, wenn die Funktion im Bild "Bild_1" aufgerufen wird.

OpenPicture

Funktion

Führt ein Wechsel des angegebenen Grundbildes durch. Am Client und bei einem Bildnamen mit Serverpräfix wird ein Bildwechsel im Bildfenster durchgeführt.

Wenn sich z. B. das Bildfenster in einem anderen Bildfenster mit Serverpräfix befindet, wird der Bildwechsel nicht im Bildfenster durchgeführt, in dem die Funktion aufgerufen wurde.

Syntax

void OpenPicture(Picture Bildname)

Parameter

Bildname

Name des Bildes

Registry2

Funktion

Diese Funktion verwaltet eine Liste von Stringpaaren (String0, String1).

Sie kennt folgende Aufrufarten, die über den Parameter mode gesteuert werden:

- Registry2("set", "String0", "String1");

Nimmt das übergebene Stringpaar in die Liste auf.

- Registry2("get", "String0", NULL);

Liefert den ersten Stringpaarpartner String1 zurück, der zum übergebenen String0 gehört und löscht anschließend das Stringpaar aus der Liste.

- Registry2("reset", NULL, NULL);

Löscht alle Stringpaare aus der Liste.

- Registry2("display", NULL, NULL);

Gibt die aktuell in der Liste gespeicherten Stringpaare in einem Global Script Diagnosefenster aus.

Syntax

char* Registry2(char* mode, char* String0, char* String1);

Parameter

mode

Legt die Arbeitsweise der Funktion fest.

set	Aufnahme des Stringpaares in die Liste
get	Ermitteln des ersten Stringpaarpartners zu String0 und Löschen des Stringpaares aus der Liste

reset	Löschen aller Stringpaare
display	Ausgabe der Stringpaare in einem Global Script Diagnosefenster

String0

Die Versorgung des Parameters ist abhängig von der Arbeitsweise der Funktion.

String1

Die Versorgung des Parameters ist abhängig von der Arbeitsweise der Funktion.

Rückgabewert

Im Modus mode=get, wird ein Zeiger auf den erste Stringpaarpartner zurückgegeben.

Hinweis

Diese Funktion wird im Zusammenhang mit der Bildbausteintechnik benutzt.

Wenn Sie im Dynamic Wizard - unter der Registerkarte "Bildbausteine" - mit den Wizards "Bildbaustein als Typ erstellen " und "Instanz(en) erzeugen im Anlagenbild" arbeiten, dürfen Sie die Funktion "Registry2" nicht einsetzen.

2.15.2.4 Obsolete functions

Alarm

AXC_OnBtnAlarmHidingList

Funktion

Die Funktion zeigt in einem Meldefenster die Liste der ausgeblendeten Meldungen an.

Syntax

BOOL AXC_OnBtnAlarmHidingList(char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

AXC_OnBtnArcLong

Funktion

Die Funktion zeigt in einem Meldefenster die in einer Langzeitarchivliste gespeicherten Meldungen an.

Syntax

BOOL AXC_OnBtnArcLong (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnArcShort

Funktion

Die Funktion zeigt in einem Meldefenster die in einer Kurzzeitarchivliste gespeicherten Meldungen an.

Syntax

BOOL AXC_OnBtnArcShort (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnComment

Funktion

Externe Meldefensterbedienung

Diese Funktion zeigt den Kommentar der zuvor selektierten Meldung an.

Syntax

BOOL AXC_OnBtnComment (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnEmergAckn

Funktion

Externe Meldefensterbedienung

Diese Funktion öffnet den Quittierungsdialog (Not-Quittierung/Reset).

Syntax

BOOL AXC_OnBtnEmergAckn (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnHideDlg

Funktion

Diese Funktion öffnet den Anzeigeoptions-Dialog, in dem festgelegt wird, welche Meldungen im Meldefenster angezeigt werden. Die Optionen sind "Alle Meldungen", "Eingeblendete Meldungen" oder "Ausgeblendete Meldungen".

Syntax

BOOL AXC_OnBtnHideDlg(char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

AXC_OnBtnHideUnhideMsg

Funktion

Die Funktion blendet die ausgewählte Meldung aus oder blendet die ausgeblendete Meldung wieder ein.

Syntax

BOOL AXC_OnBtnHideUnhideMsg (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

AXC_OnBtnHit

Funktion

Die Funktion zeigt in einem Meldefenster die in der Hitliste gespeicherten Meldungen an.

Syntax

BOOL AXC_OnBtnHit (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnHornAckn

Funktion

Externe Meldefensterbedienung

Diese Funktion quittiert das Hupensignal .

Syntax

BOOL AXC_OnBtnHornAckn (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnInfo

Funktion

Externe Meldefensterbedienung

Diese Funktion zeigt den Infotext an.

Syntax

BOOL AXC_OnBtnInfo (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnLock

Funktion

Externe Meldefensterbedienung

Diese Funktion öffnet den Dialog "Sperrliste parametrieren".

Syntax

BOOL AXC_OnBtnLock (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnLockUnlock

Funktion

Die Funktion sperrt im Meldefenster die selektierte Meldung. Diese Meldung wird danach nicht mehr archiviert.

Die Funktion entsperrt in der Sperrliste die selektierte Meldung.

Syntax

BOOL AXC_OnBtnLockUnlock (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnLockWin

Funktion

Externe Meldefensterbedienung.

Diese Funktion ruft die Sperrliste auf.

Syntax

BOOL AXC_OnBtnLockWin (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnLoop

Funktion

Externe Meldefensterbedienung

Mit dieser Funktion wird die Funktion "LoopInAlarm" der selektierten Meldung ausgelöst.

Syntax

BOOL AXC_OnBtnLoop (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnMsgFirst

Funktion

Externe Meldefensterbedienung

Diese Funktion wechselt an den Anfang der Meldeliste.

Syntax

BOOL AXC_OnBtnMsgFirst (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnMsgLast

Funktion

Externe Meldefensterbedienung

Diese Funktion wechselt an das Ende der Meldeliste.

Syntax

BOOL AXC_OnBtnMsgLast (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgLast

AXC_OnBtnMsgNext

Funktion

Externe Meldefensterbedienung

Diese Funktion wechselt zur nächsten Meldung in der Meldeliste.

Syntax

BOOL AXC_OnBtnMsgNext (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnMsgPrev

Funktion

Externe Meldefensterbedienung

Diese Funktion wechselt zur vorherigen Meldung in der Meldeliste.

Syntax

BOOL AXC_OnBtnMsgPrev (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnMsgWin

Funktion

Externe Meldefensterbedienung

Diese Funktion ruft die Meldeliste auf.

Hinweis

Die Meldeliste beinhaltet die aktuell anstehenden und nicht quittierten Meldungen.

Syntax

BOOL AXC_OnBtnMsgWin (char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnPrint

Funktion

Externe Meldefensterbedienung

Es werden alle Meldungen auf dem Drucker ausgegeben, die das im Alarm Control eingestellte Selektionskriterium erfüllen.

Syntax

BOOL AXC_OnBtnPrint(char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnProtocol

Funktion

Externe Meldefensterbedienung

Es wird der Ausdruck der aktuellen Sicht des Alarm Controls gestartet. Es werden alle Meldungen auf dem Drucker ausgegeben, die das im Alarm Control eingestellte Selektionskriterium erfüllen.

Syntax

BOOL AXC_OnBtnProtocol(char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

AXC_OnBtnScroll

Funktion

Externe Meldefensterbedienung

Diese Funktion aktiviert bzw. deaktiviert die horizontale und vertikale Scrollfunktion.

Syntax

BOOL AXC_OnBtnScroll(char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnScroll

AXC_OnBtnSelect

Funktion

Externe Meldefensterbedienung

Diese Funktion öffnet den Dialog "Selektion festlegen" für die angezeigte Liste.

Syntax

BOOL AXC_OnBtnSelect(char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

AXC_OnBtnSinglAckn

Funktion

Externe Meldefensterbedienung

Diese Funktion quittiert die aktuell selektierte Meldung.

Syntax

BOOL AXC_OnBtnSinglAckn(char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnSinglAckn

AXC_OnBtnSortDlg

Funktion

Externe Bedienung des Meldefensters

Diese Funktion öffnet den Dialog zur Einstellung einer benutzerdefinierten Sortierung der angezeigten Meldungen für die angezeigte Liste.

Syntax

BOOL AXC_OnBtnSortDlg(char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

AXC_OnBtnTimeBase

Funktion

Externe Bedienung des Meldefensters

Diese Funktion öffnet den Dialog zur Einstellung der Zeitbasis für die in den Meldungen angezeigten Zeitangaben.

Syntax

BOOL AXC_OnBtnTimeBase(char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

AXC_OnBtnVisibleAckn

Funktion

Externe Meldefensterbedienung

Alle im Meldefenster sichtbaren Meldungen werden quittiert (Sammelquittierung).

Syntax

BOOL AXC_OnBtnVisibleAckn(char* lpszPictureName, char* lpszObjectName)

Parameter

lpszPictureName

Zeiger auf den Bildnamen des Bildes, in dem sich das WinCC Alarm Control befindet

lpszObjectName

Zeiger auf den Objektnamen des WinCC Alarm Controls

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC AlarmControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel AXC_OnBtnMsgFirst

Report

ReportJob

Funktion

Je nach Wert des Parameters lpMethodName wird ein Druckauftrag oder die Vorschau zu einem Druckauftrag gestartet.

Syntax

void ReportJob(LPSTR lpJobName, LPSTR lpMethodName)

Parameter

lpJobName

Zeiger auf den Namen des Druckauftrags

lpMethodName

PRINTJOB	Druckauftrag wird gestartet
PREVIEW	Vorschau zum Druckauftrag wird gestartet

Hinweis

Diese Funktion wird durch die Funktionen RPTJobPreview und RPTJobPrint ersetzt und sollte nicht mehr verwendet werden.

TagLog

TOOLBAR_BUTTONS

TlgTableWindowPressEditRecordButton

Funktion

Das Editieren des Tabellenfensters wird gesperrt oder freigegeben (Toggle-Funktion).

Wird das Editieren freigegeben, dann wird zugleich auch die Aktualisierung des Tabellenfensters angehalten.

Die Aktualisierung des Tabellenfensters bleibt danach angehalten, auch wenn das Editieren durch einen weiteren Aufruf der Funktion gesperrt wird.

Syntax

BOOL TlgTableWindowPressEditRecordButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

TlgTableWindowPressFirstButton

Funktion

Zeigt die ersten Datensätze des Darstellungsbereichs im Tabellenfenster an.

Die Anzahl der angezeigten Datensätze ist abhängig vom projektierten Zeitbereich.

Syntax

BOOL TlgTableWindowPressFirstButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTableWindowPressHelpButton

Funktion

Zeigt die Online Hilfe für das Tabellenfenster an.

Syntax

BOOL TlgTableWindowPressHelpButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTableWindowPressInsertRecordButton

Syntax

BOOL TlgTableWindowPressInsertRecordButton(char* lpszWindowName)

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

TlgTableWindowPressLastButton

Funktion

Zeigt die letzten Datensätze des Darstellungsbereichs im Tabellenfenster an.

Die Anzahl der angezeigten Datensätze ist abhängig vom projektierten Zeitbereich.

Syntax

BOOL TlgTableWindowPressLastButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTableWindowPressNextButton

Funktion

Die auf den aktuellen Darstellungsbereich folgenden Datensätze des Darstellungsbereichs werden im Tabellenfenster angezeigt.

Die Anzahl der angezeigten Datensätze ist abhängig vom projektierten Zeitbereich.

Syntax

BOOL TlgTableWindowPressNextButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTableWindowPressNextItemButton

Funktion

Die Spalten des Tabellenfensters werden um eine Spalte nach links verschoben, wobei die linke Spalte an die Position der rechten Spalte gesetzt wird.

Syntax

BOOL TlgTableWindowPressNextItemButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTableWindowPressOpenArchiveVariableSelectionDlgButton

Funktion

Öffnet den Dialog zur Verbindung von Tabellenspalten mit Archiven und Variablen.

Syntax

BOOL TlgTableWindowPressOpenArchiveVariableSelectionDlgButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTableWindowPressOpenDlgButton

Funktion

Öffnet den Dialog zur Onlineprojektierung des Tabellenfensters.

Syntax

BOOL TlgTableWindowPressOpenDlgButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressOpenDlgButton

TlgTableWindowPressOpenItemSelectDlgButton

Funktion

Öffnet den Dialog zur Auswahl der sichtbaren Spalten und der ersten Spalte des Tabellenfensters.

Syntax

BOOL TlgTableWindowPressOpenItemSelectDlgButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTableWindowPressOpenTimeSelectDlgButton

Funktion

Öffnet den Dialog zur Einstellung des in den Tabellenspalten darzustellenden Zeitbereichs.

Syntax

BOOL TlgTableWindowPressOpenTimeSelectDlgButton(char* lpszWindowNumber)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTableWindowPressPrevButton

Funktion

Die vor dem aktuellen Darstellungsbereich liegenden Datensätze des Darstellungsbereichs werden im Tabellenfenster angezeigt.

Die Anzahl der angezeigten Datensätze ist abhängig vom projektierten Zeitbereich.

Syntax

BOOL TlgTableWindowPressPrevButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTableWindowPressPrevItemButton

Funktion

Die Spalten des Tabellenfensters werden um eine Spalte nach rechts verschoben, wobei die rechte Spalte an die Position der linken Spalte gesetzt wird.

Syntax

BOOL TlgTableWindowPressPrevItemButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTableWindowPressRemoveRecordButton

Syntax

BOOL TlgTableWindowPressRemoveRecordButton(char* lpszWindowName)

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

TlgTableWindowPressStartStopButton

Funktion

Die Aktualisierung des Tabellenfensters wird ein- oder ausgeschaltet (Toggle-Funktion).

Syntax

BOOL TlgTableWindowPressStartStopButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressFirstButton

Funktion

Zeigt die ersten Datensätze des Darstellungsbereichs im Kurvenfenster an.

Die Anzahl der angezeigten Datensätze ist abhängig vom projektierten Zeitbereich.

Syntax

BOOL TlgTrendWindowPressFirstButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressHelpButton

Funktion

Zeigt die Online-Hilfe für das Kurvenfenster an.

Syntax

BOOL TlgTrendWindowPressHelpButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressLastButton

Funktion

Zeigt die letzten Datensätze des Darstellungsbereichs im Kurvenfenster an.

Die Anzahl der angezeigten Datensätze ist abhängig vom projektierten Zeitbereich.

Syntax

BOOL TlgTrendWindowPressLastButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressLinealButton

Funktion

Das Lineal des Kurvenfensters wird ein- oder ausgeblendet (Toggle-Funktion).

Das Lineal kann mit den Tasten "Cursor links" und "Cursor rechts" verschoben werden.

Syntax

BOOL TlgTrendWindowPressLinealButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressNextButton

Funktion

Die auf den aktuellen Darstellungsbereich folgenden Datensätze des Darstellungsbereichs werden im Kurvenfenster angezeigt.

Die Anzahl der angezeigten Datensätze ist abhängig vom projektierten Zeitbereich.

Syntax

BOOL TlgTrendWindowPressNextButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressNextItemButton

Funktion

Bringt alle Kurven im Kurvenfenster eine Ebene nach vorn.

Die Kurve im Vordergrund wird dabei in den Hintergrund gesetzt.

Syntax

BOOL TlgTrendWindowPressNextItemButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressOneToOneButton

Funktion

Stellt die Standardgröße (1:1) im Kurvenfenster wieder her.

Syntax

BOOL TlgTrendWindowPressOneToOneButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressOpenArchiveVariableSelectionDlgButton

Funktion

Öffnet den Dialog zur Verbindung von Kurven mit Archiven und Variablen.

Syntax

BOOL TlgTrendWindowPressOpenArchiveVariableSelectionDlgButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressOpenDlgButton

Funktion

Öffnet den Dialog zur Onlineprojektierung des Kurvenfensters.

Syntax

BOOL TlgTrendWindowPressOpenDlgButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressOpenItemSelectDlgButton

Funktion

Öffnet den Dialog zur Auswahl der sichtbaren Kurven und der Kurve, die im Vordergrund stehen soll.

Syntax

BOOL TlgTrendWindowPressOpenItemSelectDlgButton(char* lpszWindowNumber)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressOpenTimeSelectDlgButton

Funktion

Öffnet den Dialog zur Einstellung des darzustellenden Zeitbereichs.

Syntax

BOOL TlgTrendWindowPressOpenTimeSelectDlgButton(char* lpszWindowNumber)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressPrevButton

Funktion

Die vor dem aktuellen Darstellungsbereich liegenden Datensätze des Darstellungsbereichs werden im Kurvenfenster angezeigt.

Die Anzahl der angezeigten Datensätze ist abhängig vom projektierten Zeitbereich.

Syntax

BOOL TlgTrendWindowPressPrevButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressPrevItemButton

Funktion

Bringt alle Kurven im Kurvenfenster eine Ebene nach hinten.

Die Kurve im Hintergrund wird dabei in den Vordergrund gesetzt.

Syntax

BOOL TlgTrendWindowPressPrevItemButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressPrintButton

Funktion

Die aktuelle Sicht der Kurven wird entsprechend der projektierten Darstellung des WinCC Trend Control ausgegeben.

Syntax

BOOL TlgTrendWindowPressPrintButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

TlgTrendWindowPressReportSaveButton

Funktion

Die angezeigten Daten des Kurvenfensters werden in einer Textdatei gespeichert.

Syntax

BOOL TlgTrendWindowPressReportSaveButton (char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

TlgTrendWindowPressStartStopButton

Funktion

Die Aktualisierung des Kurvenfensters wird ein- oder ausgeschaltet (Toggle-Funktion).

Syntax

BOOL TlgTrendWindowPressStartStopButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressStatsResultButton

Funktion

Startet die Auswertung der Daten im gewählten Zeitbereich.
Berechnet werden die Statistikwerte Minimum, Maximum, Durchschnitt, Standardabweichung.

Syntax

BOOL TlgTrendWindowPressStatsResultButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressStatsSelectRangeButton

Funktion

Zur Wahl des Zeitbereichs für die Statistikfunktion werden die Lineale für Anfangs- und Endzeitpunkt angezeigt.

Syntax

BOOL TlgTrendWindowPressStatsSelectRangeButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressStartStopButton

TlgTrendWindowPressZoomInButton

Funktion

Die Lupe im Kurvenfenster wird eingeschaltet. Die Auswahl des Zoombereichs ist nur mit der Maus möglich.

Syntax

BOOL TlgTrendWindowPressZoomInButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressZoomInButton

TlgTrendWindowPressZoomOutButton

Funktion

Das Kurvenfenster wird in den Zustand gebracht, den es vor dem Einschalten der Lupe hatte. Die Lupe wird ausgeschaltet.

Die Auswahl des Zoombereichs ist nur mit der Maus möglich (siehe auch TlgTrendWindowPressZoomInButton).

Syntax

BOOL TlgTrendWindowPressZoomOutButton(char* lpszWindowName)

Parameter

lpszWindowName

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgTrendWindowPressZoomOutButton

Template

TlgGetNumberOfColumns

Funktion

Liefert die Anzahl der Spalten im Tabellenfenster.

Der Fenstertitel des entsprechenden WinCC Online Table Control wird mit dem Parameter lpszTemplate übergeben.

Syntax

int TlgGetNumberOfColumns(char* lpszTemplate)

Parameter

lpszTemplate

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

Anzahl der Spalten in einem Tabellenfenster

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

TlgGetNumberOfRows

Funktion

Liefert die Anzahl der Zeilen im Tabellenfenster.

Der Fenstertitel des entsprechenden WinCC Online Table Control wird mit dem Parameter lpszTemplate übergeben.

Syntax

int TlgGetNumberOfRows(char* lpszTemplate)

Parameter

lpszTemplate

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

Anzahl der Zeilen im Tabellenfenster

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgGetNumberOfRows

TlgGetNumberOfTrends

Funktion

Liefert die Anzahl der Kurven im Kurvenfenster.

Der Fenstertitel des entsprechenden WinCC Online Trend Control wird mit dem Parameter lpszTemplate übergeben.

Syntax

int TlgGetNumberOfTrends(char* lpszTemplate)

Parameter

lpszTemplate

Zeiger auf den Fenstertitel des WinCC Online Trend Control

Rückgabewert

Anzahl der Kurven im Kurvenfenster

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

TlgGetRowPosition

Funktion

Liefert die aktuelle Position des Zeilenzeigers im Tabellenfenster.

Der Fenstertitel des entsprechenden WinCC Online Table Control wird mit dem Parameter lpszTemplate übergeben.

Syntax

int TlgGetRowPosition(char* lpszTemplate)

Parameter

lpszTemplate

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

Aktuelle Position des Zeilenzeigers im Tabellenfenster

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

TlgGetRulerArchivNameTrend

Funktion

Liefert den Archivnamen der Kurve mit der Nummer nTrend im Kurvenfenster an der Position des Lineals.

Der Fenstertitel des entsprechenden WinCC Online Trend Control wird mit dem Parameter lpszTemplate übergeben.

Syntax

char* TlgGetRulerArchivNameTrend(char* lpszTemplate, int nTrend)

Parameter

lpszTemplate

Zeiger auf den Fenstertitel des WinCC Online Trend Control

nTrend

Nummer der Kurve

(0 <= nTrend <= Anzahl sichtbarer Kurven - 1)

Rückgabewert

Archivname der Kurve mit der Nummer nTrend im Kurvenfenster an der Position des Lineals

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgGetRulerVariableNameTrend

TlgGetRulerTimeTrend

Funktion

Liefert die Zeit der Kurve mit der Nummer nTrend im Kurvenfenster an der Position des Lineals.

Der Fenstertitel des entsprechenden WinCC Online Trend Control wird mit dem Parameter lpszTemplate übergeben.

Syntax

SYSTEMTIME TlgGetRulerTimeTrend(char* lpszTemplate, int nTrend)

Parameter

lpszTemplate

Zeiger auf den Fenstertitel des WinCC Online Trend Control

nTrend

Nummer der Kurve

(0 <= nTrend <= Anzahl sichtbarer Kurven - 1)

Rückgabewert

Zeit der Kurve mit der Nummer nTrend im Kurvenfenster an der Position des Lineals

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgGetRulerTimeTrend (Seite 1587)

TlgGetRulerValueTrend

Funktion

Liefert den Wert der Kurve mit der Nummer nTrend im Kurvenfenster an der Position des Lineals.

Der Fenstertitel des entsprechenden WinCC Online Trend Control wird mit dem Parameter lpszTemplate übergeben.

Syntax

double TlgGetRulerValueTrend(char* lpszTemplate, int nTrend)

Parameter

lpszTemplate

Zeiger auf den Fenstertitel des WinCC Online Trend Control

nTrend

Nummer der Kurve

(0 <= nTrend <= Anzahl sichtbarer Kurven - 1)

Rückgabewert

Wert der Kurve mit der Nummer nTrend im Kurvenfenster an der Position des Lineals

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

TlgGetRulerVariableNameTrend

Funktion

Liefert den Variablennamen der Kurve mit der Nummer nTrend im Kurvenfenster.

Der Fenstertitel des entsprechenden WinCC Online Trend Control wird mit dem Parameter lpszTemplate übergeben.

Syntax

char* TlgGetRulerVariableNameTrend(char* lpszTemplate, int nTrend)

Parameter

lpszTemplate

Zeiger auf den Fenstertitel des WinCC Online Trend Control

nTrend

Nummer der Kurve

(0 <= nTrend <= Anzahl sichtbarer Kurven - 1)

Rückgabewert

Der Variablennamen der Kurve mit der Nummer nTrend im Kurvenfenster.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgGetRulerVariableNameTrend

TlgGetTextAtPos

Funktion

Liefert für Prozesswertarchive und Anwenderarchive den Inhalt einer Zelle des Tabellenfensters als Text.

Die Zelle wird durch nColumn und nLine spezifiziert.

Der Fenstertitel des entsprechenden WinCC Online Table Control wird mit dem Parameter lpszTemplate übergeben.

Syntax

char* TlgGetTextAtPos(char* lpszTemplate, int nColumn, int nLine)

Parameter

lpszTemplate

Zeiger auf den Fenstertitel des WinCC Online Table Control

nColumn

Nummer der Spalte

nLine

Nummer der Zeile

Rückgabewert

Inhalt einer Zelle des Tabellenfensters als Text

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgGetRulerVariableNameTrend

TlgGetColumnPosition

Funktion

Liefert die aktuelle Position des Spaltenzeigers im Tabellenfenster als Spaltenindex.

Syntax

int TlgGetColumnPosition(char* lpszTemplate)

Parameter

lpszTemplate

Zeiger auf den Fenstertitel des WinCC Online Table Control

Rückgabewert

Aktuelle Position des Spaltenzeigers in einem Tabellenfenster

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTableControl ab WinCC V7.0 nicht mehr unterstützt.

Siehe auch

Beispiel TlgGetNumberOfColumns

TlgTrendWindowActivateCurve

Funktion

Aktiviert eine bestimmte Kurve im WinCC Online Trend Control über den projektierten Namen der Kurve. Diese Funktion wird unabhängig davon ausgeführt, ob die Kurve sichtbar ist bzw. im Vordergrund steht.

Syntax

BOOL TlgTrendWindowActivateCurve(char* lpszPictureName, char* lpszObjectName, char* szValue)

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Trend Control

szValue

Name der Kurve

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Die Standardfunktion wird für das neue WinCC OnlineTrendControl ab WinCC V7.0 nicht mehr unterstützt.

2.15.2.5 Report

Report - Kurzbeschreibung

Die Gruppe Report enthält Funktionen, mit denen Sie die Druckvorschau eines Druckauftrags bzw. den Druckauftrag selbst starten können.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

RPTJobPreview

Funktion

Die Vorschau zu einem Druckauftrag wird gestartet.

Syntax

BOOL RPTJobPreview(LPSTR lpJobName)

Parameter

lpJobName

Zeiger auf den Namen des Druckauftrags

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel RPTJobPreview

RPTJobPrint

Funktion

Ein Druckauftrag wird gestartet.

Syntax

BOOL RPTJobPrint(LPSTR lpJobName)

Parameter

lpJobName

Zeiger auf den Namen des Druckauftrags

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel RPTJobPrint

RptShowError

Funktion

Die Funktion liefert eine Fehlerbeschreibung zu einem gescheiterten Druckauftrag.

Der Aufruf der Funktion ist bereits Bestandteil der Standard-Funktionen RptJobPrint und RptJobPreview und bedarf daher keines eigenen Aufrufes.

Die Fehlerbeschreibung wird in einem Global Script Diagnosefenster ausgegeben.

Hinweis

Da es sich bei RptShowError um eine Standard-Funktion handelt, kann bei Bedarf Art und Form der Ausgabe geändert werden.

Bitte beachten Sie, dass veränderte Standard-Funktionen durch eine WinCC Installation überschrieben werden und somit die Änderungen verloren gehen.

Syntax

void RptShowError(LPCSTR pszFailedFunction, CMN_ERRORA* pCmnErrorA)

Parameter

pszFailedFunction

Zeiger auf den Namen der gescheiterten Funktion.

Ist dieser Zeiger NULL, dann erfolgt keine Ausgabe des Funktionsnamens.

pCmnErrorA

Zeiger auf die Fehlerstruktur der gescheiterten Funktion.

Ist dieser Zeiger NULL, dann erfolgt keine Ausgabe der Fehlerstruktur.

STRUKTUREN_TABELLEN_FEHLERSTRUKTUR

2.15.2.6 WinCC

WinCC - Kurzbeschreibung

Die Gruppe WinCC enthält Funktionen, die das gesamte WinCC System betreffen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetHWDiag

Funktion

Mit dieser Funktion realisieren Sie einen direkten Start der Diagnose zur Laufzeit, ausgelöst durch ein zu projektierendes Ereignis an einem Objekt.

Tritt das Ereignis ein, wird die Funktion "Hardware diagnostizieren" von STEP7 für die zugehörige Steuerung gestartet.

Damit Sie die Funktion nutzen können, müssen folgende Voraussetzungen erfüllt sein:

- Das WinCC Projekt mit dem Bild, aus dem der Einsprung erfolgen soll, und das STEP7 Projekt müssen auf demselben Rechner liegen.
- Das WinCC Projekt muss als Unterverzeichnis des STEP7 Projekts angelegt sein (STEP7 Projekt\wincproj\WinCC Projekt).
- Die S7 Variablen sind auf WinCC abgebildet.

Syntax

BOOL GetHWDiag(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, LPCTSTR lpProperties)

Parameter

lpszPictureName

Name des Bildes (PDL-Datei), das die Variablen enthält, die für den Einsprung in die Hardware-Diagnose verwendet werden

Da die Bezeichnung "lpszPictureName" für das aktuelle Bild steht, wird hier nur in den Fällen ein Eintrag benötigt, in denen man auf eine Variable eines Objekts in einem anderen Bild einspringen möchte.

lpszObjectName

Name des Objekts im Bild, das mit den Variablen verbunden ist, die für den Einsprung in die Hardware-Diagnose verwendet werden

Da die Bezeichnung "lpszObjectName" für das aktuelle Objekt steht, wird hier nur in den Fällen ein Eintrag benötigt, in denen man auf eine Variable eines anderen Objekts einspringen möchte.

lpProperties

Name des Attributs, das mit der Variablen verbunden ist, die für den Einsprung in die Hardware-Diagnose verwendet wird

Bei Angabe mehrerer Attribute müssen diese durch Semikolon (";") getrennt werden.

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

GetHWDiagLevel

Funktion

Führt eine Berechtigungsprüfung für den angemeldeten Benutzer anhand der unter dwLevel angeführten User Administrator Funktionsnummer durch.

Anschließend wird ein direkter Start der Diagnose zur Laufzeit durchgeführt – ausgelöst durch ein zu projektierendes Ereignis an einem Objekt.

Tritt das Ereignis ein, wird die Funktion "Hardware diagnostizieren" von STEP7 für die zugehörige Steuerung gestartet.

Damit Sie die Funktion nutzen können, müssen folgende Voraussetzungen erfüllt sein:

- Das WinCC Projekt mit dem Bild, aus dem der Einsprung erfolgen soll, und das STEP7 Projekt müssen auf demselben Rechner liegen.
- Das WinCC Projekt muss als Unterverzeichnis des STEP7 Projekts angelegt sein (STEP7 Projekt\wincproj\WinCC Projekt).
- Die S7 Variablen sind auf WinCC abgebildet.
- Damit der in WinCC angemeldete Benutzer den Dialog zur Hardware-Diagnose editieren kann, muss der Benutzer die WinCC Benutzerberechtigung mit der Nummer besitzen, die im Funktionsaufruf im Parameter "dwLevel" übergeben wird.

Syntax

BOOL GetHWDiagLevel(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, LPCTSTR lpProperties, DWORD dwLevel)

Parameter

lpszPictureName

Name des Bildes (PDL-Datei), das die Variablen enthält, die für den Einsprung in die Hardware-Diagnose verwendet werden

Da die Bezeichnung "lpszPictureName" für das aktuelle Bild steht, wird hier nur in den Fällen ein Eintrag benötigt, in denen man auf eine Variable eines Objekts in einem anderen Bild einspringen möchte.

lpszObjectName

Name des Objekts im Bild, das mit den Variablen verbunden ist, die für den Einsprung in die Hardware-Diagnose verwendet werden

Da die Bezeichnung "lpszObjectName" für das aktuelle Objekt steht, wird hier nur in den Fällen ein Eintrag benötigt, in denen man auf eine Variable eines anderen Objekts einspringen möchte.

lpProperties

Name des Attributs, das mit der Variablen verbunden ist, die für den Einsprung in die Hardware-Diagnose verwendet wird

Bei Angabe mehrerer Attribute müssen diese durch Semikolon (";") getrennt werden.

dwLevel

Levelnummer für die STEP7 Schreibberechtigung

Diese kann im User Administrator bestimmt werden.

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

GetKopFupAwl

Funktion

Diese Funktion führt den Netzwerkeinsprung von WinCC in den STEP7 Editor "KFA" durch.

Bei der Ausführung der Funktion werden zwei Teilaufgaben erledigt:

- Ermitteln der benötigten Daten für den Netzwerkeinsprung aus WinCC.
- Übergabe der Daten an Step7 und Auffinden der Verwendungsstelle des Operanden in einem STEP7 Programm mit Hilfe des AUTAPI.

Syntax

BOOL GetKopFupAwl(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, LPCTSTR lpProperties)

Parameter

lpszPictureName

Name des Bildes (PDL-Datei), das die Variablen enthält, die für den Netzwerkeinsprung verwendet werden

Da die Bezeichnung "lpszPictureName" für das aktuelle Bild steht, wird hier nur in den Fällen ein Eintrag benötigt, in denen man auf eine Variable eines Objekts in einem anderen Bild einspringen möchte.

lpszObjectName

Name des Objekts im Bild, das mit den Variablen verbunden ist, die für den Netzwerkeinsprung verwendet werden

Da die Bezeichnung "lpszObjectName" für das aktuelle Objekt steht, wird hier nur in den Fällen ein Eintrag benötigt, in denen man auf eine Variable eines anderen Objekts einspringen möchte.

lpProperties

Name des Attributs, das mit der Variablen verbunden ist, die für den Netzwerkeinsprung verwendet wird

Bei Angabe mehrerer Attribute sind diese durch Semikolon (";") zu trennen.

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

GetKopFupAwlLevel

Funktion

Führt eine Berechtigungsprüfung für den aktiven Benutzer anhand der unter dwLevel angeführten User Administrator Funktionsnummer durch und macht anschließend den Einsprung in den STEP7 Editor "KFA".

Bei der Ausführung der Funktion werden drei Teilaufgaben erledigt:

- Ermitteln der benötigten Daten für den Netzwerkeinsprung aus WinCC.
- Durchführung der Berechtigungsprüfung für den aktiven Benutzer innerhalb von WinCC.
- Übergabe der Daten an STEP7 und Auffinden der Verwendungsstelle des Operanden in einem STEP7 Programm mit Hilfe des AUTAPI.

Hinweis

Abhängig vom Ergebnis der Berechtigungsprüfung in WinCC hat der Benutzer in STEP7 entweder nur Leserecht oder die Berechtigung zum Verändern der S7-Daten.

Syntax

BOOL GetKopFupAwlLevel(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName,LPCTSTR lpProperties, DWORD dwLevel)

Parameter

lpszPictureName

Name des Bildes (PDL-Datei), das die Variablen enthält, die für den Netzwerkeinsprung verwendet werden

Da die Bezeichnung "lpszPictureName" für das aktuelle Bild steht, wird hier nur in den Fällen ein Eintrag benötigt, in denen man auf eine Variable eines Objekts in einem anderen Bild einspringen möchte.

lpszObjectName

Name des Objekts im Bild, das mit den Variablen verbunden ist, die für den Netzwerkeinsprung verwendet werden

Da die Bezeichnung "lpszObjectName" für das aktuelle Objekt steht, wird hier nur in den Fällen ein Eintrag benötigt, in denen man auf eine Variable eines anderen Objekts einspringen möchte.

lpProperties

Name des Attributs, das mit der Variablen verbunden ist, die für den Netzwerkeinsprung verwendet wird

Bei Angabe mehrerer Attribute sind diese durch Semikolon (";") zu trennen.

dwLevel

Levelnummer für die STEP7 Schreibberechtigung.

Diese kann im User Administrator bestimmt werden.

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

OnDeactivateExecute

Funktion

Diese Funktion wird beim Beenden von WinCC Runtime aufgerufen.

Da es sich um eine Standard-Funktion handelt, können Sie Anweisungen einfügen, die dann ausgeführt werden.

Hinweis

Bei den Anweisungen ist zu berücksichtigen, dass die Runtime im Begriff ist, sich zu beenden und daher nicht mehr alle Funktionalitäten zur Verfügung stehen.

Bitte beachten Sie, dass veränderte Standard-Funktionen durch eine WinCC Installation überschrieben werden und somit die Änderungen verloren gehen

Syntax

```
void OnDeactivateExecute()
```

OnErrorExecute

Funktion

OnErrorExecute wird vom System aufgerufen, wenn beim Ausführen einer Aktion oder einer Funktion ein Fehler aufgetreten ist.

Damit wird die Möglichkeit geschaffen, die genaue Fehlerursache zu bestimmen.

Die Funktion wird vom System aufgerufen und bedarf keines zusätzlichen Aufrufes.

Da diese Funktion als Standard-Funktion vorliegt, kann bei Bedarf die Art und Form der Ausgabe beeinflusst werden.

Hinweis

Bitte beachten Sie, dass bei einer Neuinstallation veränderte Standard-Funktionen überschrieben werden und somit die Änderungen verloren gehen.

Syntax

void OnErrorExecute(CCAPErrorExecute ErrorExecute)

Parameter

ErrorExecute

Struktur, die Auskunft über den aufgetretenen Fehler gibt

Diagnoseinformationen

Diese Informationen werden in einem Global Script Diagnosefenster ausgegeben.

SystemTime	Der Zeitpunkt (UTC), an dem der Fehler aufgetreten ist
dwErrorCode1	Die Fehlercodes und deren Bedeutung finden Sie in der Strukturdefinition
dwErrorCode2	Die Fehlercodes und deren Bedeutung finden Sie in der Strukturdefinition
szErrorText	Textuelle Beschreibung der Fehlerursache
bCycle	Zyklusart
szApplicationName	Fehlerauslösende Applikation
szFunctionName	FunktionsID
szTagName	Variablenname
dwCycle	Zyklusart
szErrorTextTagName	Textuelle Beschreibung des Variablenstatus
status	Variablenstatus

lpszPictureName	Das Bild, in dem der Fehler aufgetreten ist
lpszObjectName	Das Objekt, in dem der Fehler aufgetreten ist
lpszPropertyName	Die Objekteigenschaft, in der der Fehler aufgetreten ist
dwParamSize	wird nur intern verwendet
szErrortext	Textuelle Beschreibung der Fehlerursache, von der Fehlerstruktur "pError" geliefert

Siehe auch

Strukturdefinition CCAPErrorExecute

OnTime

Funktion

OnTime wird ausschließlich vom System aufgerufen. Die Funktion gibt die Laufzeit aller Aktionen aus oder ermittelt die Aktionen, die länger als eine vorgegebene Zeit laufen. Die Zeitmessung kann über APDIAG ein- und ausgeschaltet werden.

Da diese Funktion als Standard-Funktion vorliegt, kann durch Änderung am Funktionscode die Art der Ausgabe beeinflusst werden.

Hinweis

Bitte beachten Sie, dass veränderte Standard-Funktionen durch eine WinCC Installation überschrieben werden und somit die Änderungen verloren gehen

Syntax

void OnTime(CCAPTime time)

Parameter

time

Ergebnisstruktur

STRUKTUREN_TABELLEN_CCAPTIME

2.15.2.7 Windows

Windows - Kurzbeschreibung

Die Gruppe Windows enthält die Funktion ProgramExecute.

Mit dieser Funktion können Sie ein beliebiges Programm ausführen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

ProgramExecute

Funktion

Startet das Programm mit dem angegebenen Namen.

Syntax

unsigned int ProgramExecute(char* Program_Name)

Parameter

Program_Name

Zeiger auf den Programmnamen

Rückgabewert

Ist der Rückgabewert größer als 31, wurde die Funktion fehlerfrei durchlaufen.

Im Fehlerfall enthält der Rückgabewert einen der folgenden Fehlercodes:

0	out of memory
2	Die angegebene Datei konnte nicht gefunden werden.
3	Der angegebene Pfad konnte nicht gefunden werden
11	Das Programm konnte nicht gestartet werden.

Siehe auch

ProgramExecute

2.15.3 Interne Funktionen

2.15.3.1 Interne Funktionen - Kurzbeschreibung

Interne Funktionen verwenden Sie zur Dynamisierung von grafischen Objekten und Archiven sowie in Projekt-Funktionen, Standard-Funktionen und Global Script Aktionen.

Interne Funktionen sind projektübergreifend bekannt.

Sie können weder neue interne Funktionen erstellen, noch vorhandene interne Funktionen ändern.

Interne Funktionen sind in folgende Gruppen eingeteilt:

allocate

Funktionen zum Reservieren und Freigeben von Arbeitsspeicher

c_bib

Funktionen aus der C-Standardbibliothek

graphics

Funktionen zum Lesen und Setzen von Eigenschaften grafischer Objekte

tag

Funktionen zum Schreiben und Lesen von Variablen

wincc

Funktionen zur Sprachumschaltung, zum Deaktivieren des Runtime und zum Beenden von WinCC

2.15.3.2 allocate

SysFree

Funktion

Gibt den mit der Funktion SysMalloc reservierten Speicherbereich wieder frei.

Syntax

void SysFree(void* lpFree);

Parameter

lpFree

Zeiger auf den mit Funktion SysMalloc reservierten Speicherbereich

SysMalloc

Funktion

Reserviert Speicher für eine Aktion. Der Speicherbereich wird der Aktion zugeordnet. Ist die Aktion durchlaufen und das Ergebnis verschickt, wird der Speicher vom System wieder freigegeben.

Mit der Funktion SysFree kann der Speicher vorzeitig wieder freigegeben werden.

Syntax

```
void* SysMalloc(unsigned long int size);
```

Parameter

size

Größe des Speicherbereichs in Byte.

2.15.3.3 c_bib

c_bib - Kurzbeschreibung

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file

- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

ctype

isalnum

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

isalpha

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

isdigit

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

isgraph

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

islower

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

isprint

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

ispunct

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

isspace

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

isupper

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

isxdigit

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

tolower

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

toupper

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

math

acos

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

asin

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

atan

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

atan2

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

ceil

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

cos

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

cosh

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

exp

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

fabs

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

floor

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

fmod

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

frexp

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

ldexp

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

log

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

log10

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

modf

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

pow

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

sin

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

sinh

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

sqrt

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

tan

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

tanh

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

memory

memchr

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

memcmp

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib

- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

memcpy

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

memmove

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

memset

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

stdio

char_io

fgetc

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio

- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

fgets

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

fputc

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

fputs

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

getc

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

putc

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

ungetc

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

Directio

fread

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

fwrite

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib

- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

Error

clearerr

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file

- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

feof

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

ferror

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

File

fclose

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib

- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

fflush

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

fopen

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

freopen

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

remove

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

rename

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

setbuf

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

setvbuf

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

tmpfile

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

tmpnam

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

File_pos

fgetpos

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

fseek

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

fsetpos

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

ftell

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

rewind

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

Output

vfprintf

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

vsprintf

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib

- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

stdlib

abs

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file

- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

atof

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

atoi

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

atol

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

bsearch

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

calloc

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

div

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

free

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

getenv

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

labs

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

ldiv

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

malloc

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

qsort

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

rand

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

realloc

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

srand

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strtod

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strtol

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strtoul

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

string

strcat

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib

- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strchr

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strcmp

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strcpy

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strcspn

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strerror

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strlen

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strncat

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strncmp

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strncpy

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strpbrk

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strrchr

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strspn

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strstr

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strtok

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

time

asctime

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

clock

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib

- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

ctime

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

difftime

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

gmtime

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

localtime

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

mktime

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

strftime

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

time

Die Funktionsgruppe c_bib enthält C-Funktionen aus der C-Bibliothek und ist unterteilt in:

- ctype
- math
- memory
- stdio
- stdlib
- string
- time

stdio selbst ist noch einmal unterteilt in:

- char_io
- directio
- error
- file
- file_pos
- output

Die Beschreibung dieser Funktionen finden Sie in der einschlägigen Fachliteratur.

Hinweis

Die Funktion localtime hat bei der Datumsausgabe folgendes Verhalten:

Die Zählung der Monate beginnt mit 0.

Die Jahre werden ab dem Jahr 1900 gezählt, beginnend mit 0.

Die Funktionen der C-Bibliothek printf(), sprintf(), fprintf() können in WinCC nur 360 Zeichen verarbeiten.

2.15.3.4 graphics

Graphics - Kurzbeschreibung

Mit den Funktionen der Gruppe Graphics können Sie grafische Eigenschaften von WinCC Objekten verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

Hinweis

Bezieht sich der Aufruf der Funktion auf das Bild-Objekt, ist der Parameter lpszObjectName = NULL zu setzen.

get

axes

GetAlignment

Funktion

Gibt bei Balkenobjekten an, ob der Text rechts oder links vom Balken steht.

Syntax

BOOL GetAlignment(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Text steht rechts vom Balken

FALSE

Text steht links vom Balken

Siehe auch

Beispiel GetScaling

Beispiel GetScaling (Seite 1543)

GetAxisSection

Funktion

Liefert bei Balkenobjekten die Wertdifferenz zwischen zwei benachbarten Achsenbeschriftungen.

Syntax

double GetAxisSection(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Wertdifferenz zwischen zwei benachbarten Achsenbeschriftungen

GetExponent

Funktion

Gibt bei Balkenobjekten an, ob die Achsenbeschriftung der Dezimal- oder Exponentialschreibweise entspricht.

Syntax

BOOL GetExponent(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Achsenbeschriftung in Exponentialschreibweise

FALSE

Achsenbeschriftung in Dezimalschreibweise

Siehe auch

Beispiel GetScaling

Beispiel GetScaling (Seite 1543)

GetLeftComma

Funktion

Gibt bei Balkenobjekten die Anzahl der Vorkommastellen der Achsenbeschriftung an.

Syntax

long int GetLeftComma(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Anzahl der Vorkommastellen der Achsenbeschriftung

GetLongStrokesBold

Funktion

Gibt bei Balkenobjekten an, ob die Hauptteilstriche auf der Skala fett oder normal dargestellt werden.

Syntax

BOOL GetLongStrokesBold(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Fette Hauptteilstriche auf der Balkenskala

FALSE

Normale Hauptteilstriche auf der Balkenskala

Siehe auch

Beispiel GetScaling

Beispiel GetScaling (Seite 1543)

GetLongStrokesOnly

Funktion

Gibt bei Balkenobjekten an, ob auf der Skala auch Unterteilstriche verwendet werden.

Syntax

BOOL GetLongStrokesOnly(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Auf der Balkenskala werden nur Hauptteilstriche verwendet.

FALSE

Auf der Balkenskala werden sowohl Haupt- als auch Unterteilstriche verwendet.

Siehe auch

Beispiel GetScaling

Beispiel GetScaling (Seite 1543)

GetLongStrokesSize

Funktion

Gibt bei Balkenobjekten die Länge der Hauptteilstriche an.

Syntax

long int GetLongStrokesSize(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Länge der Hauptteilstriche als Zahlenwert

GetLongStrokesTextEach

Funktion

Gibt bei Balkenobjekten an, der wievielte Hauptteilstrich eine Beschriftung hat.

Syntax

long int GetLongStrokesTextEach(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Beschriftung der Hauptteilstriche als Zahlenwert

Beispiel:

Rückgabewert = 1 -> Jeder Hauptteilstrich hat eine Beschriftung.

Rückgabewert = 2 -> Jeder 2. Hauptteilstrich hat eine Beschriftung.

usw.

GetRightComma

Funktion

Gibt bei Balkenobjekten die Anzahl der Nachkommastellen der Achsenbeschriftung an.

Syntax

long int GetRightComma(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Anzahl der Nachkommastellen der Achsenbeschriftung

GetScaleTicks

Funktion

Liefert bei Balkenobjekten die Skaleneinteilung als Anzahl der Skalenabschnitte. Dabei ist ein Skalenabschnitt der Bereich, der von zwei Hauptteilstrichen begrenzt wird.

Syntax

long int GetScaleTicks(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Skaleneinteilung als Anzahl der Skalenabschnitte

Hinweis

Die Anzahl der Skalenabschnitte wird mit 0 angegeben, wenn das Balkenobjekt selbst eine geeignete Skaleneinteilung berechnet.

GetScaling

Funktion

Gibt bei Balkenobjekten zurück, ob die Skala ein- oder ausgeschaltet ist.

Syntax

BOOL GetScaling(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Darstellung mit Skala

FALSE

Darstellung ohne Skala

Siehe auch

Beispiel GetScaling

Beispiel GetScaling (Seite 1543)

GetScalingType

Funktion

Gibt bei Balkenobjekten die Art der Balkenskalierung an.

Syntax

long int GetScalingType(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Art der Balkenskalierung als Zahlenwert

Siehe auch

Balkenskalierung

Balkenskalierung (Seite 1593)

color

Color - Kurzbeschreibung

Mit den Funktionen der Gruppe Color können Sie an Objekten verschiedene Farbeigenschaften verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetBackColor

Funktion

Gibt die Hintergrundfarbe des Objekts als Zahlenwert an.

Syntax

long int GetBackColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Hintergrundfarbe des Objekts als Zahlenwert

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, so ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetBackColor2

Funktion

Gibt bei Balkenobjekten die Balkenfarbe als Zahlenwert an.

Syntax

long int GetBackColor2(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Balkenfarbe als Zahlenwert

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetBackColor3

Funktion

Gibt bei Balkenobjekten die Balkenhintergrundfarbe als Zahlenwert an.

Syntax

long int GetBackColor3(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Balkenhintergrundfarbe als Zahlenwert

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetBackColorBottom

Funktion

Gibt bei Slider-Objekten die Hintergrundfarbe unten rechts an.

Syntax

long int GetBackColorBottom(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Zahlenwert der Hintergrundfarbe unten rechts bei Slider-Objekten

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetBackColorTop

Funktion

Gibt bei Slider-Objekten die Hintergrundfarbe oben links an.

Syntax

long int GetBackColorTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Zahlenwert der Hintergrundfarbe oben links bei Slider-Objekten

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetBorderBackColor

Funktion

Gibt die Linien- oder Rahmenhintergrundfarbe an.

Syntax

long int GetBorderBackColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Linien- oder Rahmenhintergrundfarbe als Zahlenwert

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetBorderColor

Funktion

Gibt die Linien- oder Rahmenfarbe als Zahlenwert an.

Syntax

long int GetBorderColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Linien- oder Rahmenfarbe als Zahlenwert

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetBorderColorBottom

Funktion

Gibt die 3D-Rahmenfarbe unten an.

Syntax

long int GetBorderColorBottom(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

3D-Rahmenfarbe unten als Zahlenwert

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetBorderColorTop

Funktion

Gibt die 3D-Rahmenfarbe oben an.

Syntax

long int GetBorderColorTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

3D-Rahmenfarbe oben als Zahlenwert

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetButtonColor

Funktion

Gibt bei Slider-Objekten die Knopffarbe an.

Syntax

long int GetButtonColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Knopffarbe bei Slider-Objekten als Zahlenwert

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetColorBottom

Funktion

Gibt bei Slider-Objekten die Farbe des unteren Anschlags an.

Syntax

long int GetColorBottom(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Farbe des unteren Anschlags bei Slider-Objekten als Zahlenwert

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetColorTop

Funktion

Gibt für Slider-Objekte die Farbe des oberen Anschlags an.

Syntax

long int GetColorTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Farbe des oberen Anschlags bei Slider-Objekten als Zahlenwert

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetFillColor

Funktion

Gibt die Farbe des Füllmusters an.

Syntax

long int GetFillColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Farbe des Füllmusters als Zahlenwert

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetForeColor

Funktion

Gibt die Schriftfarbe an.

Syntax

long int GetForeColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Schriftfarbe als Zahlenwert

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetGridColor

Funktion

Gibt die Rasterfarbe des Graphics Designer an.

Syntax

long int GetGridColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Rasterfarbe des Graphics Designer als Zahlenwert

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetItemBorderBackColor

Funktion

Gibt für das Objekt "Textliste" die Hintergrundfarbe der Trennlinie an.

Syntax

long int GetItemBorderBackColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Hintergrundfarbe der Trennlinie beim Objekt "Textliste" als Zahlenwert

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetItemBorderColor

Funktion

Gibt für das Objekt "Textliste" die Trennlinienfarbe an.

Syntax

long int GetItemBorderColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Trennlinienfarbe beim Objekt "Textliste" als Zahlenwert

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetScaleColor

Funktion

Gibt bei Balkenobjekten die Skalenfarbe an.

Syntax

long int GetScaleColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Skalenfarbe bei Balkenobjekten als Zahlenwert

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetSelBGColor

Funktion

Gibt für das Objekt "Textliste" die Hintergrundfarbe für den selektierten Eintrag an.

Syntax

long int GetSelBGColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Zahlenwert der Hintergrundfarbe für den selektierten Eintrag

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetSelTextColor

Funktion

Gibt für das Objekt "Textliste" die Schriftfarbe für den selektierten Eintrag an.

Syntax

long int GetSelTextColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Zahlenwert der Schriftfarbe für den selektierten Eintrag

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetTrendColor

Funktion

Gibt für Balkenobjekte die Trendfarbe an.

Syntax

long int GetTrendColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Zahlenwert der Trendfarbe bei Balkenobjekten

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

GetUnselBGColor

Funktion

Gibt für das Objekt "Textliste" die Hintergrundfarbe für die nicht selektierten Einträge an.

Syntax

long int GetUnselBGColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Zahlenwert der Hintergrundfarbe für die nicht selektierten Einträge

Siehe auch

Beispiel GetBackColor

Farbtabelle

Farbtabelle (Seite 1595)

Beispiel GetBackColor (Seite 1526)

GetUnselTextColor

Funktion

Gibt für das Objekt "Textliste" die Schriftfarbe für die nicht selektierten Einträge an.

Syntax

long int GetUnselTextColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Zahlenwert der Schriftfarbe für die nicht selektierten Einträge

Siehe auch

Beispiel GetBackColor

Farbtabelle

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

fill

Fill - Kurzbeschreibung

Die Funktionen der Gruppe Fill steuern das dynamische Füllen von Objekten.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetFilling

Funktion

Gibt an, ob das dynamische Füllen mit der Hintergrundfarbe aktiviert ist.

Syntax

BOOL GetFilling(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Das dynamische Füllen mit der Hintergrundfarbe ist aktiviert.

FALSE

Das dynamische Füllen mit der Hintergrundfarbe ist nicht aktiviert.

Siehe auch

Beispiel GetFilling

Beispiel GetFilling (Seite 1527)

GetFillingIndex

Funktion

Gibt den aktuellen Füllstand an.

Syntax

long int GetFillingIndex(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Aktueller Füllstand als Zahlenwert (0 – 100)

Siehe auch

Beispiel GetFillingIndex

Beispiel GetFillingIndex (Seite 1528)

flash

Flash - Kurzbeschreibung

Mit den Funktionen der Gruppe Flash werden verschiedene, das Blinken betreffende Eigenschaften verändert bzw. abgefragt.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetBackFlashColorOff

Funktion

Gibt die Hintergrundblinkfarbe für den ausgeschalteten Zustand an.

Syntax

long int GetBackFlashColorOff(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Hintergrundblinkfarbe für den ausgeschalteten Zustand als Zahlenwert

Siehe auch

Farbtabelle (Seite 1595)

Beispiel GetFlashBackColorOn

Beispiel GetFlashBackColorOn (Seite 1530)

Farbtabelle

GetBackFlashColorOn

Funktion

Gibt die Hintergrundblinkfarbe für den eingeschalteten Zustand an.

Syntax

long int GetBackFlashColorOn(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Hintergrundblinkfarbe für den eingeschalteten Zustand als Zahlenwert

Siehe auch

Beispiel GetFlashBackColorOn (Seite 1530)

Farbtabelle (Seite 1595)

Farbtabelle

Beispiel GetFlashBackColorOn

GetBorderFlashColorOff

Funktion

Gibt die Rahmen- oder Linienblinkfarbe für den ausgeschalteten Zustand an.

Syntax

long int GetBorderFlashColorOff(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Rahmen- oder Linienblinkfarbe für den ausgeschalteten Zustand als Zahlenwert

Siehe auch

Beispiel GetFlashBackColorOn (Seite 1530)

Farbtabelle (Seite 1595)

Farbtabelle

Beispiel GetFlashBackColorOn

GetBorderFlashColorOn

Funktion

Gibt die Rahmen- oder Linienblinkfarbe für den eingeschalteten Zustand an.

Syntax

long int GetBorderFlashColorOn(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Rahmen- oder Linienblinkfarbe für den eingeschalteten Zustand als Zahlenwert

Siehe auch

Beispiel GetFlashBackColorOn (Seite 1530)

Farbtabelle (Seite 1595)

Farbtabelle

Beispiel GetFlashBackColorOn

GetFlashBackColor

Funktion

Gibt an, ob das Blinken des Hintergrundes aktiviert ist.

Syntax

BOOL GetFlashBackColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Blinken des Hintergrundes ist aktiviert.

FALSE

Blinken des Hintergrundes ist nicht aktiviert.

Siehe auch

Beispiel GetFlashBackColor (Seite 1529)

Beispiel GetFlashBackColor

GetFlashBorderColor

Funktion

Gibt an, ob das Blinken des Rahmens oder der Linie aktiviert ist.

Syntax

BOOL GetFlashBorderColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Blinken des Rahmens oder der Linie ist aktiviert.

FALSE

Blinken des Rahmens oder der Linie ist nicht aktiviert.

Siehe auch

Beispiel GetFlashBackColor (Seite 1529)

Beispiel GetFlashBackColor

GetFlashForeColor

Funktion

Gibt an, ob das Blinken der Schrift aktiviert ist.

Syntax

BOOL GetFlashForeColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Blinken der Schrift ist aktiviert.

FALSE

Blinken der Schrift ist nicht aktiviert.

Siehe auch

Beispiel GetFlashBackColor (Seite 1529)

Beispiel GetFlashBackColor

GetFlashRateBackColor

Funktion

Gibt die Blinkfrequenz des Hintergrundes an.

Syntax

long int GetFlashRateBackColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Blinkfrequenz des Hintergrundes

Siehe auch

Beispiel GetFlashBackColorOn (Seite 1530)

Blinkfrequenzen (Seite 1593)

Beispiel GetFlashBackColorOn

Blinkfrequenzen

GetFlashRateBorderColor

Funktion

Gibt die Blinkfrequenz der Linie oder des Rahmens an.

Syntax

long int GetFlashRateBorderColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Blinkfrequenz der Linie oder des Rahmens

Siehe auch

Beispiel GetFlashBackColorOn (Seite 1530)

Blinkfrequenzen (Seite 1593)

Beispiel GetFlashBackColorOn

Blinkfrequenzen

GetFlashRateForeColor

Funktion

Gibt die Blinkfrequenz der Schrift an.

Syntax

long int GetFlashRateForeColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Blinkfrequenz der Schrift

Siehe auch

Beispiel GetFlashBackColorOn (Seite 1530)

Blinkfrequenzen (Seite 1593)

Beispiel GetFlashBackColorOn

Blinkfrequenzen

GetForeFlashColorOff

Funktion

Gibt die Schriftblinkfarbe für den ausgeschalteten Zustand an.

Syntax

long int GetForeFlashColorOff(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Schriftblinkfarbe für den ausgeschalteten Zustand als Zahlenwert

Siehe auch

Beispiel GetFlashBackColorOn (Seite 1530)

Farbtabelle (Seite 1595)

Beispiel GetFlashBackColorOn

Blinkfrequenzen

GetForeFlashColorOn

Funktion

Gibt die Schriftblinkfarbe für den eingeschalteten Zustand an.

Syntax

long int GetForeFlashColorOn(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Schriftblinkfarbe für den eingeschalteten Zustand als Zahlenwert

Siehe auch

Beispiel GetFlashBackColorOn (Seite 1530)

Farbtabelle (Seite 1595)

Beispiel GetFlashBackColorOn

Farbtabelle

focus

Focus - Kurzbeschreibung

Mit den Funktionen der Gruppe Focus können Sie den Fokus auf ein Objekt setzen bzw. abfragen, welches Objekt den Fokus besitzt.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

Get_Focus

Funktion

Gibt den Namen des Objekts an, welches den Focus besitzt oder zuletzt besaß.

Syntax

char* Get_Focus();

Rückgabewert

Name des Objekts, das den Focus besitzt oder zuletzt besaß.

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
```

```
{
   . . . . . .
}
```

Siehe auch

Beispiel GetFocus (Seite 1531)

Beispiel GetFocus

font

Font - Kurzbeschreibung

Mit den Funktionen der Gruppe Font werden verschiedene, den Text betreffende Eigenschaften verändert bzw. abgefragt.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetAlignmentLeft

Funktion

Gibt die Textausrichtung horizontal (links, zentriert, rechts) an.

Syntax

long int GetAlignmentLeft(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Textausrichtung horizontal als Zahlenwert

Siehe auch

Textausrichtung (Seite 1601)
Beispiel GetFontSize (Seite 1532)
Beispiel GetFontSize
Textausrichtung

GetAlignmentTop

Funktion

Gibt die Textausrichtung vertikal (oben, zentriert, unten) an.

Syntax

long int GetAlignmentTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName
Name des Bildes

lpszObjectName
Name des Objekts

Rückgabewert

Textausrichtung vertikal als Zahlenwert

Siehe auch

Beispiel GetFontSize (Seite 1532)
Textausrichtung (Seite 1601)
Beispiel GetFontSize
Textausrichtung

GetFontBold

Funktion

Gibt an, ob der Schriftstil "fett" ist.

Syntax

BOOL GetFontBold(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Schriftstil "fett" ein

FALSE

Schriftstil "fett" aus

Siehe auch

Beispiel GetFontBold (Seite 1531)

Beispiel GetFontBold

GetFontItalic

Funktion

Gibt an, ob der Schriftstil "kursiv" ist.

Syntax

BOOL GetFontItalic(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Schriftstil "kursiv" ein

FALSE

Schriftstil "kursiv" aus

Siehe auch

Beispiel GetFontBold (Seite 1531)

Beispiel GetFontBold

GetFontName

Funktion

Gibt die aktuelle Schriftart an.

Syntax

char* GetFontName(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Zeiger auf den Namen der aktuell eingestellten Schriftart.

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Siehe auch

Beispiel GetText (Seite 1560)

Beispiel GetText

GetFontSize

Funktion

Gibt den Schriftgrad an.

Syntax

long int GetFontSize(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Aktueller Schriftgrad

Siehe auch

Beispiel GetFontSize (Seite 1532)

Beispiel GetFontSize

GetFontUnderline

Funktion

Gibt an, ob der Schriftstil "unterstrichen" ist.

Syntax

BOOL GetFontUnderline(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Schriftstil "unterstrichen" ein

FALSE

Schriftstil "unterstrichen" aus

Siehe auch

Beispiel GetFontBold (Seite 1531)

Beispiel GetFontBold

GetOrientation

Funktion

Gibt die Schreibrichtung (vertikal/horizontal) an.

Syntax

BOOL GetOrientation(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Schreibrichtung "vertikal"

FALSE

Schreibrichtung "horizontal"

Siehe auch

Beispiel GetFontBold (Seite 1531)

Beispiel GetFontBold

GetText

Funktion

Gibt bei Objekten wie statischem Text, Check- und Radio-Box den Wert der Eigenschaft "Text" an.

Syntax

char* GetText(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Zeiger auf einen Text.

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Hinweis

Bei Check- und Radio-Boxen ist vor Aufruf dieser Funktion das zu ermittelnde Element mit der Funktion "SetIndex" festzulegen.

Siehe auch

Beispiel GetText (Seite 1560)

general

GetLayer

Funktion

Gibt die Bildebene an, in der sich das Objekt befindet.

Syntax

long int GetLayer(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Bildebene, in der sich das Objekt befindet

geometry

Geometry - Kurzbeschreibung

Mit den Funktionen der Gruppe Geometry können Sie die Größe, die Position und andere geometrische Eigenschaften von Objekten verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetActualPointLeft

Funktion

Gibt für Polygone oder Polygonzüge den X-Wert des aktuellen Punktes an.

Syntax

long int GetActualPointLeft(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

X-Wert des aktuellen Punktes eines Polygons oder Polygonzugs

Hinweis

Mit der Funktion SetIndex kann der aktuelle Punkt des Polygons eingestellt werden.

Siehe auch

Beispiel GetLeft (Seite 1534)

Beispiel GetLeft

GetActualPointTop

Funktion

Gibt für Polygone oder Polygonzüge den Y-Wert des aktuellen Punktes an.

Syntax

long int GetActualPointTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Y-Wert des aktuellen Punktes eines Polygons oder Polygonzugs

Hinweis

Mit der Funktion SetIndex kann der aktuelle Punkt des Polygons eingestellt werden.

Siehe auch

Beispiel GetTop (Seite 1561)

Beispiel GetTop

GetBoxCount

Funktion

Gibt für Check- und Radio-Boxen die Anzahl der Felder an.

Syntax

long int GetBoxCount(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Anzahl der Felder in einer Check- oder Radio-Box

GetDirection

Funktion

Gibt bei Balkenobjekten die Balkenrichtung an.

Syntax

long int GetDirection(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Balkenrichtung bei Balkenobjekten als Zahlenwert

Siehe auch

Balkenrichtung (Seite 1593)

Balkenrichtung

GetEndAngle

Funktion

Gibt den Endwinkel bei Kreis- und Ellipsensegmenten sowie bei Kreis- und Ellipsenbögen an.

Syntax

long int GetEndAngle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Endwinkel bei Kreis- und Ellipsensegmenten sowie bei Kreis- und Ellipsenbögen

GetGrid

Funktion

Gibt an, ob der Raster in der Zeichenfläche des Graphics Designer eingeschaltet ist.

Syntax

BOOL GetGrid(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Raster im Graphics Designer ist eingeschaltet.

FALSE

Raster im Graphics Designer ist ausgeschaltet.

GetGridHeight

Funktion

Gibt die Höhe des Rasters in der Zeichenfläche des Graphics Designer an.

Syntax

long int GetGridHeight(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Höhe des Rasters im Graphics Designer

GetGridWidth

Funktion

Gibt die Weite des Rasters in der Zeichenfläche des Graphics Designer an.

Syntax

long int GetGridWidth(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Weite des Rasters im Graphics Designer

GetHeight

Funktion

Gibt die Höhe des umschreibenden Rechtecks eines Objekts an.

Syntax

long int GetHeight(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Höhe des umschreibenden Rechtecks eines Objekts

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Beispiel GetHeight (Seite 1532)

GetLeft

Funktion

Gibt die X-Position der linken oberen Ecke des umschreibenden Rechtecks eines Objekts an.

Syntax

long int GetLeft(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Aktueller X-Wert der linken oberen Ecke des umschreibenden Rechtecks eines Objekts

Siehe auch

Beispiel GetLeft (Seite 1534)

Beispiel GetLeft

GetPointCount

Funktion

Gibt die Anzahl der Eckpunkte eines Polygons oder eines Polygonzugs an.

Syntax

long int GetPointCount(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName
Name des Bildes

lpszObjectName
Name des Objekts

Rückgabewert

Anzahl der Eckpunkte eines Polygons oder eines Polygonzugs

GetRadius

Funktion

Gibt den Radius eines Kreises, Kreissegments oder Kreisbogens an.

Syntax

long int GetRadius(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName
Name des Bildes

lpszObjectName
Name des Objekts

Rückgabewert

Radius eines Kreises, Kreissegments oder Kreisbogens

Siehe auch

Beispiel GetHeight (Seite 1532)

Beispiel GetHeight

GetRadiusHeight

Funktion

Gibt den Radius einer Ellipse, eines Ellipsensegments oder eines Ellipsenbogens in vertikaler Richtung an.

Syntax

long int GetRadiusHeight(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Radius einer Ellipse, eines Ellipsensegments oder eines Ellipsenbogens in vertikaler Richtung

Siehe auch

Beispiel GetHeight (Seite 1532)

Beispiel GetHeight

GetRadiusWidth

Funktion

Gibt den Radius einer Ellipse, eines Ellipsensegments oder eines Ellipsenbogens in horizontaler Richtung an.

Syntax

long int GetRadiusWidth(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Radius einer Ellipse, eines Ellipsensegments oder eines Ellipsenbogens in horizontaler Richtung

Siehe auch

Beispiel GetHeight (Seite 1532)

Beispiel GetHeight

GetReferenceRotationLeft

Funktion

Gibt für Linie, Polygon und Polygonzug den X-Wert der Rotationsreferenz an (Drehpunkt, um den das Objekt gedreht werden kann).

Syntax

long int GetReferenceRotationLeft(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

X-Wert der Rotationsreferenz für Linie, Polygon und Polygonzug

Siehe auch

Beispiel GetLeft (Seite 1534)

Beispiel GetLeft

GetReferenceRotationTop

Funktion

Gibt für Linie, Polygon und Polygonzug den Y-Wert der Rotationsreferenz an (Drehpunkt, um den das Objekt gedreht werden kann).

Syntax

long int GetReferenceRotationTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Y-Wert der Rotationsreferenz für Linie, Polygon und Polygonzug

Siehe auch

Beispiel GetTop (Seite 1561)

Beispiel GetTop

GetRotationAngle

Funktion

Gibt für Linie, Polygon und Polygonzug den Rotationswinkel um den Drehpunkt an.

Syntax

long int GetRotationAngle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Rotationswinkel um den Drehpunkt

Siehe auch

Beispiel GetHeight (Seite 1532)

Beispiel GetHeight

GetRoundCornerHeight

Funktion

Gibt den vertikalen Radius der Ecke eines Rundrechtecks an.

Syntax

long int GetRoundCornerHeight(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Vertikaler Radius der Ecke eines Rundrechtecks

Siehe auch

Beispiel GetHeight (Seite 1532)

Beispiel GetHeight

GetRoundCornerWidth

Funktion

Gibt den horizontalen Radius der Ecke eines Rundrechtecks an.

Syntax

long int GetRoundCornerWidth(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Horizontaler Radius der Ecke eines Rundrechtecks

Siehe auch

Beispiel GetWidth (Seite 1562)

Beispiel GetWidth

GetStartAngle

Funktion

Gibt den Anfangswinkel bei Kreis- und Ellipsensegmenten sowie bei Kreis- und Ellipsenbögen an.

Syntax

long int GetStartAngle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Anfangswinkel bei Kreis- und Ellipsensegmenten sowie bei Kreis- und Ellipsenbögen

Siehe auch

Beispiel GetHeight (Seite 1532)

Beispiel GetHeight

GetTop

Funktion

Gibt die Y-Position der linken oberen Ecke des umschreibenden Rechtecks eines Objekts an.

Syntax

long int GetTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Aktueller Y-Wert der linken oberen Ecke des umschreibenden Rechtecks

Siehe auch

Beispiel GetTop (Seite 1561)

Beispiel GetTop

GetWidth

Funktion

Gibt die Breite des umschreibenden Rechtecks eines Objekts an.

Syntax

long int GetWidth(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Breite des umschreibenden Rechtecks eines Objekts

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Beispiel GetWidth (Seite 1562)

Beispiel GetWidth

GetZeroPoint

Funktion

Gibt bei Balkenobjekten den Nullpunkt an.

Syntax

long int GetZeroPoint(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Nullpunkt bei Balkenobjekten

Siehe auch

Beispiel GetHeight (Seite 1532)

Beispiel GetHeight

i_o

i_o - Kurzbeschreibung

Mit den Funktionen der Gruppe i_o werden verschiedene, die Ein- und Ausgabewerte betreffende Eigenschaften verändert bzw. abgefragt.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetAssignments

Funktion

Zuordnung von Text zum Wertebereich bei Listen.

Syntax

char* GetAssignments(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Die Zuordnung von Text zum Wertebereich ist abhängig von der Listenart.

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
```

```
}
```

Siehe auch

Listenarten (Seite 1599)

Listenarten

GetAssumeOnExit

Funktion

Gibt bei E/A-Feldern an, ob beim Verlassen des Feldes die Übernahme des eingegebenen Wertes erfolgt.

Syntax

```
BOOL GetAssumeOnExit(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Wertübernahme beim Verlassen des Feldes erfolgt.

FALSE

Wertübernahme beim Verlassen des Feldes erfolgt nicht.

GetAssumeOnFull

Funktion

Gibt bei E/A-Feldern an, ob die Übernahme des eingegebenen Wertes bei vollständiger Eingabe erfolgt.

Syntax

```
BOOL GetAssumeOnFull(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Wertübernahme bei vollständiger Eingabe erfolgt.

FALSE

Wertübernahme bei vollständiger Eingabe erfolgt nicht.

GetBitNumber

Funktion

Gibt bei Listenart "bit" das relevante Bit im Ausgabewert an.

Syntax

long int GetBitNumber(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Angabe des relevanten Bits im Ausgabewert bei Listenart "bit"

Siehe auch

Beispiel GetHiddenInput (Seite 1533)

Listenarten (Seite 1599)

Listenarten

Beispiel GetHiddenInput

GetClearOnError

Funktion

Gibt bei E/A-Feldern an, ob das Löschen des Inhalts bei fehlerhafter Eingabe aktiviert ist.

Syntax

BOOL GetClearOnError(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Löschen des Inhalts bei fehlerhafter Eingabe ist aktiviert

FALSE

Löschen des Inhalts bei fehlerhafter Eingabe ist nicht aktiviert

GetClearOnNew

Funktion

Gibt bei E/A-Feldern an, ob das Löschen des Inhalts bei Neueingabe aktiviert ist.

Syntax

BOOL GetClearOnNew(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Löschen des Inhalts bei Neueingabe ist aktiviert.

FALSE

Löschen des Inhalts bei Neueingabe ist nicht aktiviert.

GetDataFormat

Funktion

Gibt bei E/A-Feldern den Datentyp des Feldinhalts an.

Syntax

long int GetDataFormat(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Datentyp des Feldinhalts als Zahlenwert

Siehe auch

Beispiel GetHiddenInput (Seite 1533)

E/A-Feld, Datentyp des Feldinhalts (Seite 1595)

Beispiel GetHiddenInput

E/A-Feld, Datentyp des Feldinhalts

GetHiddenInput

Funktion

Gibt bei E/A-Feldern an, ob die verdeckte Eingabe aktiviert ist.

Syntax

BOOL GetHiddenInput(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Verdeckte Eingabe ist aktiviert

FALSE

Verdeckte Eingabe ist nicht aktiviert

Siehe auch

Beispiel GetHiddenInput (Seite 1533)

Beispiel GetHiddenInput

GetInputValueChar

Funktion

Gibt bei E/A-Feldern den Eingabewert im Datentyp "char" an.

Syntax

char* GetInputValueChar(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Zeiger auf den Eingabewert im Datentyp "char".

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

GetInputValueDouble

Funktion

Gibt bei E/A-Feldern den Eingabewert im Datentyp "double" an.

Syntax

double GetInputValueDouble(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Eingabewert im Datentyp "double"

GetListType

Funktion

Gibt für das Objekt "Textliste" die Listenart an.

Syntax

long int GetListType(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Listenart für das Objekt "Textliste"

Siehe auch

Beispiel GetHiddenInput (Seite 1533)

Listenarten (Seite 1599)

Beispiel GetHiddenInput

Listenarten

GetNumberLines

Funktion

Gibt für das Objekt "Textliste" die Anzahl der sichtbaren Zeilen an.

Syntax

long int GetNumberLines(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Anzahl der sichtbaren Zeilen beim Objekt "Textliste"

Hinweis

Ist die Anzahl der projektierten Texte größer als die Anzahl der sichtbaren Zeilen, so erhält das Objekt "Textliste" eine vertikale Bildlaufleiste.

Siehe auch

Beispiel GetHiddenInput (Seite 1533)

Beispiel GetHiddenInput

GetOutputFormat

Funktion

Gibt bei E/A-Feldern das Ausgabeformat an.

Syntax

char* GetOutputFormat(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Zeiger auf das Ausgabeformat.

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Siehe auch

E/A-Feld, Datentyp des Feldinhalts (Seite 1595)

E/A-Feld, Ausgabeformat (Seite 1594)

E/A-Feld, Datentyp des Feldinhalts

E/A-Feld, Ausgabeformat

GetOutputValueChar

Funktion

Ermittelt bei E/A-Feldern den Ausgabewert im Datentyp "char". Die Funktion sollte nur verwendet werden, wenn der Feldinhalt des E/A-Feldes vom Datentyp "string" ist.

Syntax

char* GetOutputValueChar(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Zeiger auf den Ausgabewert im Datentyp "char".

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

GetOutputValueDouble

Funktion

Ermittelt bei E/A-Feldern den Ausgabewert im Datentyp "double". Die Funktion sollte nur verwendet werden, wenn der Feldinhalt des E/A-Feldes nicht vom Datentyp "string" ist.

Syntax

double GetOutputValueDouble(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Ausgabewert im Datentyp "double"

Siehe auch

Beispiel GetOutputValueDouble (Seite 1537)

Beispiel GetOutputValueDouble

limits

Limits - Kurzbeschreibung

Mit den Funktionen der Gruppe Limits werden verschiedene, die Grenzwerte betreffende Eigenschaften verändert bzw. abgefragt.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetAlarmHigh

Funktion

Gibt bei Balkenobjekten die obere Alarmgrenze an.

Syntax

double GetAlarmHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Obere Alarmgrenze bei Balkenobjekten

Siehe auch

Beispiel GetAlarmHigh (Seite 1526)

Beispiel GetAlarmHigh

GetAlarmLow

Funktion

Gibt bei Balkenobjekten die untere Alarmgrenze an.

Syntax

double GetAlarmLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Untere Alarmgrenze bei Balkenobjekten

Siehe auch

Beispiel GetAlarmHigh (Seite 1526)

Beispiel GetAlarmHigh

GetCheckAlarmHigh

Funktion

Gibt bei Balkenobjekten an, ob die obere Alarmgrenze überwacht wird.

Syntax

BOOL GetCheckAlarmHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die obere Alarmgrenze überwacht.

FALSE

Bei Balkenobjekten wird die obere Alarmgrenze nicht überwacht.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetCheckAlarmLow

Funktion

Gibt bei Balkenobjekten an, ob die untere Alarmgrenze überwacht wird.

Syntax

BOOL GetCheckAlarmLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die untere Alarmgrenze überwacht.

FALSE

Bei Balkenobjekten wird die untere Alarmgrenze nicht überwacht.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetCheckLimitHigh4

Funktion

Gibt bei Balkenobjekten an, ob der obere Grenzwert Reserve 4 überwacht wird.

Syntax

```
BOOL GetCheckLimitHigh4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird der obere Grenzwert Reserve 4 überwacht.

FALSE

Bei Balkenobjekten wird der obere Grenzwert Reserve 4 nicht überwacht.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetCheckLimitHigh5

Funktion

Gibt bei Balkenobjekten an, ob der obere Grenzwert Reserve 5 überwacht wird.

Syntax

BOOL GetCheckLimitHigh5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird der obere Grenzwert Reserve 5 überwacht.

FALSE

Bei Balkenobjekten wird der obere Grenzwert Reserve 5 nicht überwacht.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetCheckLimitLow4

Funktion

Gibt bei Balkenobjekten an, ob der untere Grenzwert Reserve 4 überwacht wird.

Syntax

BOOL GetCheckLimitLow4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird der untere Grenzwert Reserve 4 überwacht.

FALSE

Bei Balkenobjekten wird der untere Grenzwert Reserve 4 nicht überwacht.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetCheckLimitLow5

Funktion

Gibt bei Balkenobjekten an, ob der untere Grenzwert Reserve 5 überwacht wird.

Syntax

BOOL GetCheckLimitLow5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird der untere Grenzwert Reserve 5 überwacht.

FALSE

Bei Balkenobjekten wird der untere Grenzwert Reserve 5 nicht überwacht.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetCheckToleranceHigh

Funktion

Gibt bei Balkenobjekten an, ob die obere Toleranzgrenze überwacht wird.

Syntax

```
BOOL GetCheckToleranceHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die obere Toleranzgrenze überwacht.

FALSE

Bei Balkenobjekten wird die obere Toleranzgrenze nicht überwacht.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetCheckToleranceLow

Funktion

Gibt bei Balkenobjekten an, ob die untere Toleranzgrenze überwacht wird.

Syntax

BOOL GetCheckToleranceLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die untere Toleranzgrenze überwacht.

FALSE

Bei Balkenobjekten wird die untere Toleranzgrenze nicht überwacht.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetCheckWarningHigh

Funktion

Gibt bei Balkenobjekten an, ob die obere Warngrenze überwacht wird.

Syntax

BOOL GetCheckWarningHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die obere Warngrenze überwacht.

FALSE

Bei Balkenobjekten wird die obere Warngrenze nicht überwacht.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetCheckWarningLow

Funktion

Gibt bei Balkenobjekten an, ob die untere Warngrenze überwacht wird.

Syntax

BOOL GetCheckWarningLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die untere Warngrenze überwacht.

FALSE

Bei Balkenobjekten wird die untere Warngrenze nicht überwacht.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetColorAlarmHigh

Funktion

Gibt bei Balkenobjekten die Balkenfarbe bei Erreichen der oberen Alarmgrenze an.

Syntax

long int GetColorAlarmHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Balkenfarbe als Zahlenwert bei Erreichen der oberen Alarmgrenze

Siehe auch

Beispiel GetBackColor (Seite 1526)
Farbtabelle (Seite 1595)
Beispiel GetBackColor
Farbtabelle

GetColorAlarmLow

Funktion

Gibt bei Balkenobjekten die Balkenfarbe bei Erreichen der unteren Alarmgrenze an.

Syntax

long int GetColorAlarmLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName
Name des Bildes

lpszObjectName
Name des Objekts

Rückgabewert

Balkenfarbe als Zahlenwert bei Erreichen der unteren Alarmgrenze

Siehe auch

Beispiel GetBackColor (Seite 1526)
Farbtabelle (Seite 1595)
Beispiel GetBackColor
Farbtabelle

GetColorLimitHigh4

Funktion

Gibt bei Balkenobjekten die Balkenfarbe bei Erreichen der oberen Grenze Reserve 4 an.

Syntax

long int GetColorLimitHigh4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Balkenfarbe als Zahlenwert bei Erreichen der oberen Grenze Reserve 4

Siehe auch

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

Beispiel GetBackColor

Farbtabelle

GetColorLimitHigh5

Funktion

Gibt bei Balkenobjekten die Balkenfarbe bei Erreichen der oberen Grenze Reserve 5 an.

Syntax

long int GetColorLimitHigh5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Balkenfarbe als Zahlenwert bei Erreichen der oberen Grenze Reserve 5

Siehe auch

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

Beispiel GetBackColor

Farbtabelle

GetColorLimitLow4

Funktion

Gibt bei Balkenobjekten die Balkenfarbe bei Erreichen der unteren Grenze Reserve 4 an.

Syntax

long int GetColorLimitLow4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Balkenfarbe als Zahlenwert bei Erreichen der unteren Grenze Reserve 4

Siehe auch

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

Beispiel GetBackColor

Farbtabelle

GetColorLimitLow5

Funktion

Gibt bei Balkenobjekten die Balkenfarbe bei Erreichen der unteren Grenze Reserve 5 an.

Syntax

long int GetColorLimitLow5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Balkenfarbe als Zahlenwert bei Erreichen der unteren Grenze Reserve 5

Siehe auch

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

Beispiel GetBackColor

Farbtabelle

GetColorToleranceHigh

Funktion

Gibt bei Balkenobjekten die Balkenfarbe bei Erreichen der oberen Toleranzgrenze an.

Syntax

long int GetColorToleranceHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Balkenfarbe als Zahlenwert bei Erreichen der oberen Toleranzgrenze

Siehe auch

Beispiel GetBackColor (Seite 1526)
Farbtabelle (Seite 1595)
Beispiel GetBackColor
Farbtabelle

GetColorToleranceLow

Funktion

Gibt bei Balkenobjekten die Balkenfarbe bei Erreichen der unteren Toleranzgrenze an.

Syntax

long int GetColorToleranceLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName
Name des Bildes

lpszObjectName
Name des Objekts

Rückgabewert

Balkenfarbe als Zahlenwert bei Erreichen der unteren Toleranzgrenze

Siehe auch

Beispiel GetBackColor (Seite 1526)
Farbtabelle (Seite 1595)
Beispiel GetBackColor
Farbtabelle

GetColorWarningHigh

Funktion

Gibt bei Balkenobjekten die Balkenfarbe bei Erreichen der oberen Warngrenze an.

Syntax

long int GetColorWarningHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Balkenfarbe als Zahlenwert bei Erreichen der oberen Warngrenze

Siehe auch

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

Beispiel GetBackColor

Farbtabelle

GetColorWarningLow

Funktion

Gibt bei Balkenobjekten die Balkenfarbe bei Erreichen der unteren Warngrenze an.

Syntax

long int GetColorWarningLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Balkenfarbe als Zahlenwert bei Erreichen der unteren Warngrenze

Siehe auch

Beispiel GetBackColor (Seite 1526)

Farbtabelle (Seite 1595)

Beispiel GetBackColor

Farbtabelle

GetLimitHigh4

Funktion

Gibt bei Balkenobjekten den oberen Grenzwert für Reserve 4 an.

Syntax

double GetLimitHigh4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Oberer Grenzwert für Reserve 4 bei Balkenobjekten

Siehe auch

Beispiel GetAlarmHigh (Seite 1526)

Beispiel GetAlarmHigh

GetLimitHigh5

Funktion

Gibt bei Balkenobjekten den oberen Grenzwert für Reserve 5 an.

Syntax

double GetLimitHigh5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Oberer Grenzwert für Reserve 5 bei Balkenobjekten

Siehe auch

Beispiel GetAlarmHigh (Seite 1526)

Beispiel GetAlarmHigh

GetLimitLow4

Funktion

Gibt bei Balkenobjekten den unteren Grenzwert für Reserve 4 an.

Syntax

double GetLimitLow4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Unterer Grenzwert für Reserve 4 bei Balkenobjekten

Siehe auch

Beispiel GetAlarmHigh (Seite 1526)

Beispiel GetAlarmHigh

GetLimitLow5

Funktion

Gibt bei Balkenobjekten den unteren Grenzwert für Reserve 5 an.

Syntax

double GetLimitLow5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Unterer Grenzwert für Reserve 5 bei Balkenobjekten

Siehe auch

Beispiel GetAlarmHigh (Seite 1526)

Beispiel GetAlarmHigh

GetLimitMax

Funktion

Gibt bei E/A-Feldern den oberen Grenzwert an.

Syntax

double GetLimitMax(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Oberer Grenzwert bei E/A-Feldern

Siehe auch

Beispiel GetAlarmHigh (Seite 1526)

Beispiel GetAlarmHigh

GetLimitMin

Funktion

Gibt bei E/A-Feldern den unteren Grenzwert an.

Syntax

double GetLimitMin(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Unterer Grenzwert bei E/A-Feldern

Siehe auch

Beispiel GetAlarmHigh (Seite 1526)

Beispiel GetAlarmHigh

GetMarker

Funktion

Gibt bei Balkenobjekten an, ob der Grenzwertmarkierer angezeigt wird.

Syntax

BOOL GetMarker(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Grenzwertmarkierer bei Balkenobjekten wird angezeigt.

FALSE

Grenzwertmarkierer bei Balkenobjekten wird nicht angezeigt.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetToleranceHigh

Funktion

Gibt bei Balkenobjekten die obere Toleranzgrenze an.

Syntax

double GetToleranceHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Obere Toleranzgrenze bei Balkenobjekten

Siehe auch

Beispiel GetAlarmHigh (Seite 1526)

Beispiel GetAlarmHigh

GetToleranceLow

Funktion

Gibt bei Balkenobjekten die untere Toleranzgrenze an.

Syntax

double GetToleranceLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Untere Toleranzgrenze bei Balkenobjekten

Siehe auch

Beispiel GetAlarmHigh (Seite 1526)

Beispiel GetAlarmHigh

GetTypeAlarmHigh

Funktion

Gibt bei Balkenobjekten an, ob die obere Alarmgrenze prozentual oder absolut angegeben wird.

Syntax

BOOL GetTypeAlarmHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die obere Alarmgrenze prozentual angegeben.

FALSE

Bei Balkenobjekten wird die obere Alarmgrenze absolut angegeben.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetTypeAlarmLow

Funktion

Gibt bei Balkenobjekten an, ob die untere Alarmgrenze prozentual oder absolut angegeben wird.

Syntax

```
BOOL GetTypeAlarmLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die untere Alarmgrenze prozentual angegeben.

FALSE

Bei Balkenobjekten wird die untere Alarmgrenze absolut angegeben.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetTypeLimitHigh4

Funktion

Gibt bei Balkenobjekten an, ob die obere Grenze Reserve 4 prozentual oder absolut angegeben wird.

Syntax

BOOL GetTypeLimitHigh4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die obere Grenze Reserve 4 prozentual angegeben.

FALSE

Bei Balkenobjekten wird die obere Grenze Reserve 4 absolut angegeben.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetTypeLimitHigh5

Funktion

Gibt bei Balkenobjekten an, ob die obere Grenze Reserve 5 prozentual oder absolut angegeben wird.

Syntax

BOOL GetTypeLimitHigh5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die obere Grenze Reserve 5 prozentual angegeben.

FALSE

Bei Balkenobjekten wird die obere Grenze Reserve 5 absolut angegeben.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetTypeLimitLow4

Funktion

Gibt bei Balkenobjekten an, ob die untere Grenze Reserve 4 prozentual oder absolut angegeben wird.

Syntax

BOOL GetTypeLimitLow4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die untere Grenze Reserve 4 prozentual angegeben.

FALSE

Bei Balkenobjekten wird die untere Grenze Reserve 4 absolut angegeben.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetTypeLimitLow5

Funktion

Gibt bei Balkenobjekten an, ob die untere Grenze Reserve 5 prozentual oder absolut angegeben wird.

Syntax

```
BOOL GetTypeLimitLow5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die untere Grenze Reserve 5 prozentual angegeben.

FALSE

Bei Balkenobjekten wird die untere Grenze Reserve 5 absolut angegeben.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetTypeToleranceHigh

Funktion

Gibt bei Balkenobjekten an, ob die obere Toleranzgrenze prozentual oder absolut angegeben wird.

Syntax

BOOL GetTypeToleranceHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die obere Toleranzgrenze prozentual angegeben.

FALSE

Bei Balkenobjekten wird die obere Toleranzgrenze absolut angegeben.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetTypeToleranceLow

Funktion

Gibt bei Balkenobjekten an, ob die untere Toleranzgrenze prozentual oder absolut angegeben wird.

Syntax

BOOL GetTypeToleranceLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die untere Toleranzgrenze prozentual angegeben.

FALSE

Bei Balkenobjekten wird die untere Toleranzgrenze absolut angegeben.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetTypeWarningHigh

Funktion

Gibt bei Balkenobjekten an, ob die obere Warngrenze prozentual oder absolut angegeben wird.

Syntax

BOOL GetTypeWarningHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die obere Warngrenze prozentual angegeben.

FALSE

Bei Balkenobjekten wird die obere Warngrenze absolut angegeben.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetTypeWarningLow

Funktion

Gibt bei Balkenobjekten an, ob die untere Warngrenze prozentual oder absolut angegeben wird.

Syntax

```
BOOL GetTypeWarningLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Balkenobjekten wird die untere Warngrenze prozentual angegeben.

FALSE

Bei Balkenobjekten wird die untere Warngrenze absolut angegeben.

Siehe auch

Beispiel GetMarker (Seite 1536)

Beispiel GetMarker

GetWarningHigh

Funktion

Gibt bei Balkenobjekten die obere Warngrenze an.

Syntax

double GetWarningHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Obere Warngrenze bei Balkenobjekten

Siehe auch

Beispiel GetAlarmHigh (Seite 1526)

Beispiel GetAlarmHigh

GetWarningLow

Funktion

Gibt bei Balkenobjekten die untere Warngrenze an.

Syntax

double GetWarningLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Untere Warngrenze bei Balkenobjekten

Siehe auch

Beispiel GetAlarmHigh (Seite 1526)

Beispiel GetAlarmHigh

link

Link - Kurzbeschreibung

Mit den Funktionen der Gruppe Link kann die Variablenanbindung einer Eigenschaft erzeugt bzw. abgefragt werden.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetLink

Funktion

Gibt die aktuelle Variablenverbindung von Objekteigenschaften an.

Syntax

BOOL GetLink(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, LPCTSTR lpszPropertyName, LPLINKINFO *pLink);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lpszPropertyName

Objekteigenschaft

pLink

Zeiger auf eine Struktur vom Typ: LINKINFO

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Strukturdefinition LINKINFO (Seite 1608)

Beispiel GetLink (Seite 1535)

Beispiel GetLink

Strukturdefinition LINKINFO

miscs

Miscs - Kurzbeschreibung

Mit den Funktionen der Gruppe Miscs können Sie verschiedene Eigenschaften von Objekten verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetAdaptBorder

Funktion

Gibt für statische Texte, E/A-Felder, Check- und Radio-Boxen an, ob der Rahmen des Feldes dynamisch an die Textgröße angepasst wird.

Syntax

BOOL GetAdaptBorder(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Rahmen wird angepasst

FALSE

Rahmen wird nicht angepasst

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetAdaptPicture

Funktion

Gibt für Bildfenster an, ob das Bild an die Fenstergröße angepasst wird.

Syntax

BOOL GetAdaptPicture(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bild wird angepasst

FALSE

Bild wird nicht angepasst

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetAdaptSize

Funktion

Gibt bei Bildfenstern an, ob das Fenster angepasst werden soll.

Syntax

```
BOOL GetAdaptSize(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Fenster wird angepasst

FALSE

Fenster wird nicht angepasst

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetAverage

Funktion

Gibt bei Balkenobjekten an, ob die Mittelwertbildung aktiviert ist.

Syntax

BOOL GetAverage(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Mittelwertbildung bei Balkenobjekten ist aktiviert

FALSE

Mittelwertbildung bei Balkenobjekten ist nicht aktiviert

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetBoxType

Funktion

Gibt für E/A-Felder den Feldtyp (Eingabefeld, Ausgabefeld, Ein-/Ausgabefeld) an.

Syntax

long int GetBoxType(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Feldtyp eines E/A-Feldes

Siehe auch

E/A-Feld, Feldtyp (Seite 1595)

E/A-Feld, Feldtyp

GetCaption

Funktion

Gibt bei einem Bild- und Applikationsfenster an, ob es einen Titel besitzt.

Syntax

BOOL GetCaption(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bild-/Applikationsfenster besitzt einen Titel

FALSE

Bild-/Applikationsfenster besitzt keinen Titel

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetCloseButton

Funktion

Gibt bei einem Bildfenster an, ob das Bildfenster schließbar ist.

Syntax

BOOL GetCloseButton(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bildfenster ist schließbar

FALSE

Bildfenster ist nicht schließbar

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetColorChangeType

Funktion

Gibt bei Balkenobjekten an, ob der Farbumschlag bei Erreichen eines Grenzwertes nur in einem Balkensegment erfolgt oder den gesamten Balken betrifft.

Syntax

BOOL GetColorChangeType(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Farbumschlag erfolgt im Balkensegment

FALSE

Farbumschlag erfolgt im gesamten Balken

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetCursorControl

Funktion

Gibt bei E/A-Feldern an, ob die Cursorsteuerung eingeschaltet ist.

Syntax

BOOL GetCursorControl(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Cursorsteuerung bei E/A-Feldern ist eingeschaltet

FALSE

Cursorsteuerung bei E/A-Feldern ist ausgeschaltet

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetCursorMode

Funktion

Gibt an, ob der Cursormodus für das Bild Alpha- oder Schalt-Cursor ist.

Syntax

```
BOOL GetCursorMode(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Cursormodus für das Bild ist "Alpha-Cursor"

FALSE

Cursormodus für das Bild ist "Schalt-Cursor"

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetEditAtOnce

Funktion

Gibt bei E/A-Feldern an, ob die Eigenschaft "Eingabe sofort" aktiviert ist.

Syntax

BOOL GetEditAtOnce(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Eigenschaft "Eingabe sofort" ist aktiviert

FALSE

Eigenschaft "Eingabe sofort" ist deaktiviert

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetExtendedOperation

Funktion

Gibt für Slider-Objekte an, ob die Eigenschaft "Erweiterte Bedienung" aktiviert ist.

Syntax

BOOL GetExtendedOperation(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Eigenschaft "Erweiterte Bedienung" ist aktiviert

FALSE

Eigenschaft "Erweiterte Bedienung" ist deaktiviert

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetHotkey

Funktion

Gibt bei Check-Boxen die Tastenkombination an.

Syntax

long int GetHotkey(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Tastencode für die Tastenkombinationen bei Check-Boxen

GetHysteresis

Funktion

Gibt für Balkenobjekte an, ob die Anzeige mit oder ohne Hysterese erfolgt.

Syntax

BOOL GetHysteresis(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Anzeige bei Balkenobjekten erfolgt mit Hysterese

FALSE

Anzeige bei Balkenobjekten erfolgt ohne Hysterese

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetHysteresisRange

Funktion

Gibt für Balkenobjekte den Wert der Hysterese in der Anzeige an.

Syntax

double GetHysteresisRange(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Hysterese in der Anzeige bei Balkenobjekten

GetLanguageSwitch

Funktion

Gibt für das Objekt "Textliste" an, ob die Zuordnungstexte in der Textbibliothek oder im Objekt selbst gespeichert werden.

Syntax

BOOL GetLanguageSwitch(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Die Zuordnungstexte werden in der Textbibliothek gespeichert

FALSE

Die Zuordnungstexte werden im Texlisten-Objekt gespeichert

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetLastChange

Funktion

Gibt das Datum an, zu dem die letzte Änderung des Bildes erfolgte.

Syntax

char* GetLastChange(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Datum der letzten Änderung des Bildes.

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Siehe auch

Beispiel GetPictureName (Seite 1539)

Beispiel GetPictureName

GetMax

Funktion

Gibt für Balken- und Slider-Objekte den Maximalwert an.

Syntax

double GetMax(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Maximalwert für Balken- und Slider-Objekte

GetMaximizeButton

Funktion

Gibt für Bild- oder Applikationsfenster an, ob das Fenster maximierbar ist.

Syntax

BOOL GetMaximizeButton(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bild- oder Applikationsfenster ist maximierbar

FALSE

Bild- oder Applikationsfenster ist nicht maximierbar

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetMin

Funktion

Gibt für Balken- und Slider-Objekte den Minimalwert an.

Syntax

double GetMin(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Minimalwert für Balken- und Slider-Objekte

GetMoveable

Funktion

Gibt für Bild- oder Applikationsfenster an, ob das Fenster verschiebbar ist.

Syntax

BOOL GetMoveable(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bild- oder Applikationsfenster ist verschiebbar

FALSE

Bild- oder Applikationsfenster ist nicht verschiebbar

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetOffsetLeft

Funktion

Gibt bei Bildfenstern den horizontalen Bildabstand vom linken Fensterrand an.

Syntax

long int GetOffsetLeft(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Horizontaler Bildabstand vom linken Fensterrand bei Bildfenstern

GetOffsetTop

Funktion

Gibt bei Bildfenstern den vertikalen Bildabstand vom oberen Fensterrand an.

Syntax

long int GetOffsetTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Vertikaler Bildabstand vom oberen Fensterrand bei Bildfenstern

GetOnTop

Funktion

Gibt für Bild- oder Applikationsfenster an, ob das Fenster immer im Vordergrund ist.

Syntax

BOOL GetOnTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bild- oder Applikationsfenster ist immer im Vordergrund

FALSE

Bild- oder Applikationsfenster kann von anderen Fenstern überdeckt werden

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetOperation

Funktion

Gibt an, ob das Objekt bedienbar ist.

Syntax

BOOL GetOperation(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Objekt ist bedienbar

FALSE

Objekt ist nicht bedienbar

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetOperationMessage

Funktion

Gibt für E/A-Felder, Check-Boxen, Radio-Boxen oder Slider an, ob bei Bedienung eine Meldung ausgegeben wird.

Syntax

BOOL GetOperationMessage(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bei Bedienung wird eine Meldung ausgegeben

FALSE

Bei Bedienung wird keine Meldung ausgegeben

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetOperationReport

Funktion

Gibt für alle Objekte, außer Applikations- und Bildfenstern sowie OLE-Control, an, ob der Grund für die Bedienung protokolliert wird.

Syntax

BOOL GetOperationReport(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Grund für die Bedienung wird protokolliert

FALSE

Grund für die Bedienung wird nicht protokolliert

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetPasswordLevel

Funktion

Gibt für alle Objekte, außer Applikations- und Bildfenstern sowie OLE-Control, die Berechtigungsstufe für die Bedienung des Objekts an.

Syntax

long int GetPasswordLevel(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Berechtigungsstufe für die Bedienung des Objekts

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

GetPictureName

Funktion

Gibt bei Bildfenstern den Namen des aktuell angezeigten Bildes an.

Syntax

char* GetPictureName(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Bildfensters

Rückgabewert

Zeiger auf den Bildnamen des aktuell angezeigten Bildes

Hinweis

Sind beide Parameter NULL, erhält man einen Zeiger auf den Namen des Grundbildes.

Siehe auch

Beispiel GetPictureName (Seite 1539)

Beispiel GetPictureName

GetProcess

Funktion

Gibt für Balken- und Slider-Objekte den Wert der Voreinstellung für den anzuzeigenden Prozesswert an.

Gibt für Check-Boxen und Radio-Box die selektierten Felder an.

Syntax

double GetProcess(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

- Für Balken- und Slider-Objekte: Wert der Voreinstellung für den anzuzeigenden Prozesswert
- Für Check- und Radio-Boxen: Jedes Feld wird in einem 32 Bit-Wort durch ein Bit repräsentiert (Feld 1 entspricht der Bit-Wertigkeit 0). Selektierte Felder werden durch ein gesetztes Bit markiert. Nicht vorhandene Felder werden mit 0 belegt.

GetScrollBars

Funktion

Gibt für Bildfenster an, ob das Bildfenster einen Rollbalken besitzt.

Syntax

BOOL GetScrollBars(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Bildfenster besitzt einen Rollbalken

FALSE

Bildfenster besitzt keinen Rollbalken

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetServerName

Funktion

Gibt für OLE-Control und OLE-Objekt die Voreinstellung für den anzuzeigenden Prozesswert an.

Syntax

char* GetServerName(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Namen des Objekts (OLE-Control und OLE-Objekt), unter dem es in WINDOWS registriert ist.

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Siehe auch

Beispiel GetPictureName (Seite 1539)

Beispiel GetPictureName

GetSizeable

Funktion

Gibt für Applikations- oder Bildfenster an, ob die Größe des Fensters veränderbar ist.

Syntax

BOOL GetSizeable(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Applikations- oder Bildfenster ist in der Größe veränderbar

FALSE

Applikations- oder Bildfenster ist in der Größe nicht veränderbar

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetSmallChange

Funktion

Gibt für Slider-Objekte die Anzahl der Schritte an, um die der Schieber bei einem Mausklick verschoben wird.

Syntax

long int GetSmallChange(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Anzahl der Schritte, um die der Schieber bei einem Mausklick verschoben wird

GetTagPrefix

Funktion

Liefert bei Bildfenstern das Variablenpräfix eines Bildfensters zurück.

Syntax

char* GetTagPrefix(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Variablenpräfix des Bildfensters.

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Siehe auch

Beispiel GetTagPrefix (Seite 1554)

Beispiel GetTagPrefix

GetTrend

Funktion

Gibt für Balkenobjekte an, ob die Trendanzeige aktiviert ist.

Syntax

BOOL GetTrend(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Trendanzeige bei einem Balkenobjekt ist aktiviert

FALSE

Trendanzeige bei einem Balkenobjekt ist nicht aktiviert

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetUpdateCycle

Funktion

Gibt den Aktualisierungszyklus für das gesamte Bild an.

Syntax

long int GetUpdateCycle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Aktualisierungszyklus als Zahlenwert

Siehe auch

Strukturdefinition LINKINFO (Seite 1608)

Strukturdefinition LINKINFO

GetVisible

Funktion

Gibt an, ob das Objekt angezeigt wird.

Syntax

BOOL GetVisible(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Objekt wird angezeigt

FALSE

Objekt wird nicht angezeigt

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetWindowBorder

Funktion

Gibt für Applikations- oder Bildfenster an, ob das Objekt mit Rahmen dargestellt wird.

Syntax

BOOL GetWindowBorder(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Applikations- oder Bildfenster wird mit Rahmen dargestellt

FALSE

Applikations- oder Bildfenster wird ohne Rahmen dargestellt

Siehe auch

Beispiel GetVisible (Seite 1561)

Beispiel GetVisible

GetZeroPointValue

Funktion

Gibt für Balkenobjekte den absoluten Wert des Nullpunkts an.

Syntax

double GetZeroPointValue(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Absoluter Wert des Nullpunkts bei der Balkenanzeige

GetZoom

Funktion

Gibt für Bildfenster den Skalierungsfaktor an.

Syntax

long int GetZoom(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Skalierungsfaktor eines Bildfensters

ole_control

OLE_control - Kurzbeschreibung

Die Funktionen der Gruppe ole_Control sind nur auf OCX-Slider-Objekte anwendbar.

Mit diesen Funktionen können Sie verschiedene Eigenschaften und Einstellungen eines OCX-Slider-Objekts verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetPosition

Funktion

Gibt für OCX-Slider-Objekte die Position des Schiebers an.

Syntax

long int GetPosition(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Schieberposition des OCX-Slider-Objektes als Zahlenwert

Siehe auch

Beispiel GetPosition (Seite 1540)

Beispiel GetPosition

GetRangeMax

Funktion

Gibt für OCX-Slider-Objekte den Einstellbereich "Max" an.

Syntax

long int GetRangeMax(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Einstellbereich "Max" des OCX-Slider-Objekts als Zahlenwert

Siehe auch

Beispiel GetRangeMax (Seite 1542)

Beispiel GetRangeMax

GetRangeMin

Funktion

Gibt für OCX-Slider-Objekte den Einstellbereich "Min" an.

Syntax

long int GetRangeMin(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Einstellbereich "Min" des OCX-Slider-Objekts als Zahlenwert

Siehe auch

Beispiel GetRangeMin (Seite 1543)

Beispiel GetRangeMin

pictures

Pictures - Kurzbeschreibung

Mit den Funktionen der Gruppe Pictures können Sie verschiedene Eigenschaften der Bilder von Grafik-Objekten und Rundbuttons verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetPicDeactReferenced

Funktion

Gibt bei Rundbuttons an, ob das Bild für den Zustand "deaktiviert" referenziert ist.

Syntax

BOOL GetPicDeactReferenced(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Das dem Zustand "deaktiviert" zugeordnete Bild wurde nicht im Objekt gespeichert.

FALSE

Das dem Zustand "deaktiviert" zugeordnete Bild wurde im Objekt gespeichert.

GetPicDeactTransparent

Funktion

Gibt bei Rundbuttons die Transparentfarbe für den Zustand "deaktiviert" an.

Syntax

long int GetPicDeactTransparent(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Transparentfarbe für Zustand "deaktiviert" als Zahlenwert

Hinweis

Diese Funktion gilt nur für Bitmap-Grafiken (*.bmp).

Siehe auch

Farbtabelle (Seite 1595)
Beispiel GetBackColor (Seite 1526)
Beispiel GetBackColor
Farbtabelle

GetPicDeactUseTransColor

Funktion

Gibt bei Rundbuttons an, ob die Transparentfarbe für den Zustand "deaktiviert" verwendet wird.

Syntax

BOOL GetPicDeactUseTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Transparentfarbe für Zustand "deaktiviert" wird verwendet

FALSE

Transparentfarbe für Zustand "deaktiviert" wird nicht verwendet

GetPicDownReferenced

Funktion

Gibt bei Rundbuttons an, ob das Bild für den Zustand "Ein/gedrückt" referenziert ist.

Syntax

BOOL GetPicDownReferenced(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Das dem Zustand "Ein/gedrückt" zugeordnete Bild wurde nicht im Objekt gespeichert.

FALSE

Das dem Zustand "Ein/gedrückt" zugeordnete Bild wurde im Objekt gespeichert.

GetPicDownTransparent

Funktion

Gibt bei Rundbuttons die Transparentfarbe für den Zustand "Ein/gedrückt" an.

Syntax

long int GetPicDownTransparent(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Transparentfarbe für Zustand "Ein/gedrückt" als Zahlenwert

Hinweis

Diese Funktion gilt nur für Bitmap-Grafiken (*.bmp).

Siehe auch

Farbtabelle (Seite 1595)
Beispiel GetPictureDown (Seite 1538)
Farbtabelle
Beispiel GetPictureDown

GetPicDownUseTransColor

Funktion

Gibt bei Rundbuttons an, ob die Transparentfarbe für den Zustand "Ein/gedrückt" verwendet wird.

Syntax

BOOL GetPicDownUseTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Transparentfarbe für Zustand "Ein/gedrückt" wird verwendet

FALSE

Transparentfarbe für Zustand "Ein/gedrückt" wird nicht verwendet

GetPicReferenced

Funktion

Gibt bei Grafikobjekten an, ob das Bild referenziert ist.

Syntax

BOOL GetPicReferenced(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Das zugeordnete Bild wurde nicht im Objekt gespeichert.

FALSE

Das zugeordnete Bild wurde im Objekt gespeichert.

GetPicTransColor

Funktion

Gibt bei Graphikobjekten die Transparentfarbe für ein Hintergrundbild an.

Syntax

long int GetPicTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Transparentfarbe für Hintergrundbild eines Graphikobjekts als Zahlenwert

Hinweis

Diese Funktion gilt nur für Bitmap-Grafiken (*.bmp).

Siehe auch

Farbtabelle (Seite 1595)

Beispiel GetBackColor (Seite 1526)

Beispiel GetBackColor

Farbtabelle

GetPictureDeactivated

Funktion

Gibt bei Rundbuttons den Bildnamen für den Zustand "deaktiviert" an.

Syntax

char* GetPictureDeactivated(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Bildname für den Zustand "deaktiviert".

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Hinweis

Es können Bitmap-Dateien (*.bmp, *.dib) sowie Metafiles (*.emf, *.wmf) eingebunden werden.

GetPictureDown

Funktion

Gibt bei Rundbuttons den Bildnamen für den Zustand "Ein/gedrückt" an.

Syntax

char* GetPictureDown(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Bildname für den Zustand "Ein/gedrückt".

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Hinweis

Es können Bitmap-Dateien (*.bmp, *.dib) sowie Metafiles (*.emf, *.wmf) eingebunden werden.

Siehe auch

Beispiel GetPictureDown (Seite 1538)

Beispiel GetPictureDown

GetPictureUp

Funktion

Gibt bei Rundbuttons den Bildnamen für den Zustand "Aus/nicht gedrückt" an.

Syntax

char* GetPictureUp(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Bildname für den Zustand "Aus/nicht gedrückt".

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
```

```
    . . . . . .
}
```

Hinweis

Es können Bitmap-Dateien (*.bmp, *.dib) sowie Metafiles (*.emf, *.wmf) eingebunden werden.

Siehe auch

Beispiel GetPictureUp (Seite 1540)

Beispiel GetPictureUp

GetPicUpReferenced

Funktion

Gibt bei Rundbuttons an, ob das Bild für den Zustand "Aus/nicht gedrückt" referenziert ist.

Syntax

BOOL GetPicUpReferenced(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Das dem Zustand "Aus/nicht gedrückt" zugeordnete Bild wurde nicht im Objekt gespeichert.

FALSE

Das dem Zustand "Aus/nicht gedrückt" zugeordnete Bild wurde im Objekt gespeichert.

GetPicUpTransparent

Funktion

Gibt bei Rundbuttons die Transparentfarbe für den Zustand "Aus/nicht gedrückt" an.

Syntax

long int GetPicUpTransparent(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Transparentfarbe für Zustand "Aus/nicht gedrückt" als Zahlenwert

Hinweis

Diese Funktion gilt nur für Bitmap-Grafiken (*.bmp).

Siehe auch

Farbtabelle (Seite 1595)

Beispiel GetBackColor (Seite 1526)

Beispiel GetBackColor

Farbtabelle

GetPicUpUseTransColor

Funktion

Gibt bei Rundbuttons an, ob die Transparentfarbe für den Zustand "Aus/nicht gedrückt" verwendet wird.

Syntax

BOOL GetPicUpUseTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Transparentfarbe für den Zustand "Aus/nicht gedrückt" wird verwendet

FALSE

Transparentfarbe für den Zustand "Aus/nicht gedrückt" wird nicht verwendet

GetPicUseTransColor

Funktion

Gibt bei Grafikobjekten an, ob die Transparentfarbe für ein Hintergrundbild verwendet wird.

Syntax

BOOL GetPicUseTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Transparentfarbe wird für ein Hintergrundbild verwendet.

FALSE

Transparentfarbe wird nicht für ein Hintergrundbild verwendet.

property

Property - Kurzbeschreibung

Mit den Funktionen der Gruppe Property können Sie Eigenschaften von Objekten, für die es keine eigene Funktion gibt, verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetPropBOOL

Funktion

Gibt den aktuellen Zustand einer Eigenschaft vom Datentyp "BOOL" an.

Syntax

BOOL GetPropBOOL(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, LPCTSTR lpszPropertyName)

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lpszPropertyName

Name der Objekteigenschaft

Rückgabewert

Wert des Attributs im Datentyp "BOOL"

Siehe auch

Beispiel GetPropBOOL (Seite 1541)

Beispiel GetPropBOOL

GetPropChar

Funktion

Gibt den aktuellen Zustand einer Eigenschaft vom Datentyp "char" an.

Syntax

char* GetPropChar(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, LPCTSTR lpszPropertyName)

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lpszPropertyName

Name der Objekteigenschaft

Rückgabewert

Zeiger auf eine Zeichenkette, die den Wert der Objekteigenschaft enthält.

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Siehe auch

Beispiel GetPropChar (Seite 1541)

Beispiel GetPropChar

GetPropDouble

Funktion

Gibt den aktuellen Zustand einer Eigenschaft vom Datentyp "double" an.

Syntax

double GetPropDouble(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, LPCTSTR lpszPropertyName)

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lpszPropertyName

Name der Objekteigenschaft

Rückgabewert

Wert des Attributs im Datentyp "double"

GetPropWord

Funktion

Gibt den aktuellen Zustand einer Eigenschaft vom Datentyp "long" an.

Syntax

long GetPropWord(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, LPCTSTR lpszPropertyName)

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lpszPropertyName

Name der Objekteigenschaft

Rückgabewert

Wert des Attributs im Typ "long"

state

State - Kurzbeschreibung

Mit den Funktionen der Gruppe State können Sie verschiedene Eigenschaften von Zustandsanzeigen verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetBasePicReferenced

Funktion

Gibt bei der Zustandsanzeige an, ob das Grundbild referenziert ist.

Syntax

BOOL GetBasePicReferenced(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Das Grundbild wurde nicht im Objekt gespeichert.

FALSE

Das Grundbild wurde im Objekt gespeichert.

GetBasePicTransColor

Funktion

Gibt bei der Zustandsanzeige die Transparentfarbe des Grundbildes an.

Syntax

long int GetBasePicTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Transparentfarbe des Grundbildes als Zahlenwert

Hinweis

Diese Funktion gilt nur für Bitmap-Graphiken (*.bmp).

Siehe auch

Farbtabelle (Seite 1595)

Beispiel GetBackColor (Seite 1526)

Beispiel GetBackColor

Farbtabelle

GetBasePicture

Funktion

Gibt den Grundbildnamen für die Zustandsanzeige an.

Syntax

char* GetBasePicture(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Grundbildname für die Zustandsanzeige.

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

GetBasePicUseTransColor

Funktion

Gibt bei der Zustandsanzeige an, ob die Transparentfarbe für das Grundbild verwendet wird.

Syntax

BOOL GetBasePicUseTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Transparentfarbe wird für das Grundbild verwendet

FALSE

Transparentfarbe wird nicht für das Grundbild verwendet

GetFlashFlashPicture

Funktion

Gibt bei der Zustandsanzeige an, ob das Blinkbild dynamisch oder statisch animiert wird.

Syntax

BOOL GetFlashFlashPicture(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Das Blinkbild wird dynamisch animiert.

FALSE

Das Blinkbild wird statisch animiert.

GetFlashPicReferenced

Funktion

Gibt bei der Zustandsanzeige an, ob das Blinkbild referenziert ist.

Syntax

BOOL GetFlashPicReferenced(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Das Blinkbild wurde nicht im Objekt gespeichert.

FALSE

Das Blinkbild wurde im Objekt gespeichert.

GetFlashPicTransColor

Funktion

Gibt bei der Zustandsanzeige die Transparentfarbe des Blinkbildes an.

Syntax

long int GetFlashPicTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Transparentfarbe des Blinkbildes als Zahlenwert

Hinweis

Diese Funktion gilt nur für Bitmap-Graphiken (*.bmp).

Siehe auch

Farbtabelle (Seite 1595)

Beispiel GetBackColor (Seite 1526)

Beispiel GetBackColor

Farbtabelle

GetFlashPicture

Funktion

Gibt bei der Zustandsanzeige den Blinkbildnamen an.

Syntax

char* GetFlashPicture(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Blinkbildname (Dateiname der Graphik).

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

GetFlashPicUseTransColor

Funktion

Gibt bei der Zustandsanzeige an, ob die Transparentfarbe für das Blinkbild verwendet wird.

Syntax

BOOL GetFlashPicUseTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Transparentfarbe wird für das Blinkbild verwendet

FALSE

Transparentfarbe wird nicht für das Blinkbild verwendet

GetFlashRateFlashPic

Funktion

Gibt bei der Zustandsanzeige die Blinkfrequenz eines Blinkbildes an.

Syntax

long int GetFlashRateFlashPic(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Blinkfrequenz eines Blinkbildes als Zahlenwert

Hinweis

Da es sich beim Blinken um eine softwaretechnische Realisierung handelt, ist die Frequenz system- und hardwareabhängig (Anzahl der Objekte, Prozessor, Speicher, Aktualisierungszeit u. a. m.).

Siehe auch

Blinkfrequenzen (Seite 1593)

Beispiel GetFlashRateFlashPic (Seite 1530)

Blinkfrequenzen

Beispiel GetFlashRateFlashPic

GetIndex

Funktion

Gibt bei Polygonen oder Polygonzügen den Index des aktuellen Punktes an.

Gibt bei Check-Boxen und Radio-Boxen den Index des aktuellen Feldes an.

Syntax

long int GetIndex(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Index des aktuellen Punktes oder Feldes

style

Style - Kurzbeschreibung

Mit den Funktionen der Gruppe Style werden verschiedene, das Aussehen von Objekten betreffende Eigenschaften verändert bzw. abgefragt.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

GetBackBorderWidth

Funktion

Gibt die Breite der Umrandung von 3D-Rahmen und Slider-Objekten an.

Syntax

long int GetBackBorderWidth(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Zahlenwert für die Breite der Umrandung von 3D-Rahmen und Slider-Objekten

Siehe auch

Beispiel GetBorderStyle (Seite 1527)

Beispiel GetBorderStyle

GetBorderEndStyle

Funktion

Gibt die Art des Linienendes an.

Syntax

long int GetBorderEndStyle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Art des Linienendes als Zahlenwert

Siehe auch

Beispiel GetBorderStyle (Seite 1527)

Linienenden (Seite 1599)

Beispiel GetBorderStyle

Linienenden

GetBorderStyle

Funktion

Gibt die Linien- oder Rahmenart an.

Syntax

long int GetBorderStyle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Linien- oder Rahmenart als Zahlenwert

Siehe auch

Beispiel GetBorderStyle (Seite 1527)

Linienarten (Seite 1598)

Beispiel GetBorderStyle

Linienarten

GetBorderWidth

Funktion

Gibt die Breite der Linien oder der Rahmenlinie an.

Syntax

long int GetBorderWidth(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Breite der Linien oder der Rahmenlinie als Zahlenwert

Siehe auch

Beispiel GetBorderStyle (Seite 1527)

Beispiel GetBorderStyle

GetBoxAlignment

Funktion

Gibt die Anordnung der Bedienelemente (links- oder rechtsbündig) in Check- oder Radio-Boxen an.

Syntax

long int GetBoxAlignment(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Anordnung der Bedienelemente in Check- oder Radio-Boxen als Zahlenwert

Siehe auch

Beispiel GetBorderStyle (Seite 1527)

Textausrichtung (Seite 1601)

Beispiel GetBorderStyle

Textausrichtung

GetFillStyle

Funktion

Gibt die Art des Füllmusters an.

Syntax

long int GetFillStyle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Art des Füllmusters als Zahlenwert

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Füllmuster (Seite 1597)

Beispiel GetFillStyle (Seite 1529)

Füllmuster

Beispiel GetFillStyle

GetFillStyle2

Funktion

Gibt bei einer Balkenanzeige das Füllmuster des Balkens an.

Syntax

long int GetFillStyle2(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Füllmuster des Balkens als Zahlenwert

Siehe auch

Füllmuster (Seite 1597)

Beispiel GetFillStyle (Seite 1529)

Beispiel GetFillStyle

Füllmuster

GetItemBorderStyle

Funktion

Gibt die Trennlinienart beim Objekt "Textliste" an.

Syntax

long int GetItemBorderStyle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Trennlinienart beim Objekt "Textliste"

Siehe auch

Beispiel GetBorderStyle (Seite 1527)

Linienarten (Seite 1598)

Beispiel GetBorderStyle

Linienarten

GetItemBorderWidth

Funktion

Gibt die Trennlinienbreite beim Objekt "Textliste" an.

Syntax

long int GetItemBorderWidth(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

Trennlinienbreite beim Objekt "Textliste" als Zahlenwert

Siehe auch

Beispiel GetBorderStyle (Seite 1527)

Beispiel GetBorderStyle

GetPressed

Funktion

Gibt für Buttons oder Rundbuttons an, ob die Schalterstellung "gedrückt" oder "nicht gedrückt" ist.

Syntax

BOOL GetPressed(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Schalterstellung ist "gedrückt"

FALSE

Schalterstellung ist "nicht gedrückt"

GetToggle

Funktion

Gibt bei Buttons oder Rundbuttons an, ob die Schalter rastbar ist oder nicht.

Syntax

BOOL GetToggle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Schalter ist rastbar

FALSE

Schalter ist nicht rastbar

GetWindowsStyle

Funktion

Gibt bei Buttons an, ob sie im Windows-Stil dargestellt werden.

Syntax

BOOL GetWindowsStyle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Button wird so dargestellt, wie in Windows üblich.

FALSE

Das Aussehen des Buttons bestimmen Sie selbst.

set

axes

Axes - Kurzbeschreibung

Die Funktionen der Gruppe Axes sind nur auf Balkenobjekte anwendbar.

Mit diesen Funktionen können Sie verschiedene Eigenschaften eines Balkenobjekts verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetAlignment

Funktion

Stellt bei Balkenobjekten ein, ob der Text rechts oder links vom Balken steht.

Syntax

BOOL SetAlignment(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bAlignment);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bAlignment

Textausrichtung

TRUE	Text steht rechts vom Balken
FALSE	Text steht links vom Balken

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetScaling (Seite 1576)

Beispiel SetScaling

SetAxisSection

Funktion

Stellt bei Balkenobjekten den Achsenabschnitt ein, d. h. die Wertdifferenz zwischen zwei benachbarten Achsenbeschriftungen.

Syntax

BOOL SetAxisSection(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dAxisSection);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dAxisSection

Achsenabschnitt

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetScaling (Seite 1576)

SetExponent

Funktion

Stellt bei Balkenobjekten die Darstellung der Achsenbeschriftung ein (exponential/dezimal).

Syntax

BOOL SetExponent(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bExponent);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bExponent

Achsenbeschriftung

TRUE	Achsenbeschriftung in Exponentialdarstellung
FALSE	Achsenbeschriftung in Dezimaldarstellung

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetScaling (Seite 1576)

Beispiel SetScaling

SetLeftComma

Funktion

Stellt bei Balkenobjekten die Anzahl der Vorkommastellen der Achsenbeschriftung ein.

Syntax

BOOL SetLeftComma(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lLeftComma);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lLeftComma

Anzahl der Vorkommastellen

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetScaling (Seite 1576)

Beispiel SetScaling

SetLongStrokesBold

Funktion

Stellt bei Balkenobjekten ein, ob die Hauptteilstriche fett oder normal dargestellt werden.

Syntax

```
BOOL SetLongStrokesBold(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL
bLongStrokesBold);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bLongStrokesBold

Hauptteilstriche auf der Balkenskala

TRUE	fett
FALSE	normal

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetScaling (Seite 1576)

Beispiel SetScaling

SetLongStrokesOnly

Funktion

Legt bei Balkenobjekten fest, ob auf der Balkenskala nur Haupt- oder auch Unterteilstriche verwendet werden.

Syntax

BOOL SetLongStrokesOnly(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bLongStrokesOnly);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bLongStrokesOnly

Ausschließlich Hauptteilstriche ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetScaling (Seite 1576)

Beispiel SetScaling

SetLongStrokesSize

Funktion

Stellt bei Balkenobjekten die Länge der Hauptteilstriche der Balkenskala ein.

Syntax

BOOL SetLongStrokesSize(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lLongStrokesSize);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lLongStrokesSize

Länge der Hauptteilstriche in Pixel

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetScaling (Seite 1576)

Beispiel SetScaling

SetRightComma

Funktion

Stellt bei Balkenobjekten die Anzahl der Nachkommastellen der Achsenbeschriftung ein.

Syntax

BOOL SetRightComma(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lRightComma);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lRightComma

Anzahl der Nachkommastellen

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetScaling (Seite 1576)

Beispiel SetScaling

SetScaleTicks

Funktion

Stellt bei Balkenobjekten die Skaleneinteilung als Anzahl der Skalenabschnitte ein. Dabei ist ein Skalenabschnitt der Bereich, der von zwei Hauptteilstrichen begrenzt wird.

Syntax

BOOL SetScaleTicks(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lScaleTicks);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lScaleTicks

Anzahl der Skalenabschnitte

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Wird die Anzahl der Skalenabschnitte mit 0 angegeben, berechnet das Balkenobjekt selbst eine geeignete Skaleneinteilung.

Siehe auch

Beispiel SetScaling (Seite 1576)

Beispiel SetScaling

SetScaling

Funktion

Schaltet bei Balkenobjekten die Balkenskala ein oder aus.

Syntax

BOOL SetScaling(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bScaling);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bScaling

Skala ein/aus.

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetScaling (Seite 1576)

Beispiel SetScaling

SetScalingType

Funktion

Legt bei Balkenobjekten die Art der Balkenskalierung fest.

Syntax

BOOL SetScalingType(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lScalingType);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lScalingType

Art der Balkenskalierung als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Balkenskalierung (Seite 1593)

Beispiel SetScaling (Seite 1576)

Beispiel SetScaling

Balkenskalierung

color

Color - Kurzbeschreibung

Mit den Funktionen der Gruppe Color können Sie an Objekten verschiedene Farbeigenschaften verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetBackColor

Funktion

Stellt die Hintergrundfarbe des Objekts ein.

Syntax

BOOL SetBackColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBackColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBackColor

Hintergrundfarbe des Objekts als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetBackColor2

Funktion

Stellt bei Balkenobjekten die Balkenfarbe ein.

Syntax

BOOL SetBackColor2(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBackColor2);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBackColor2

Balkenfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetBackColor3

Funktion

Stellt bei Balkenobjekten die Balkenhintergrundfarbe ein.

Syntax

BOOL SetBackColor3(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBackColor3);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBackColor3

Balkenhintergrundfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetBackColorBottom

Funktion

Stellt bei Slider-Objekten die Hintergrundfarbe unten rechts ein.

Syntax

BOOL SetBackColorBottom(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBackColorBottom);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBackColorBottom

Hintergrundfarbe von Slider-Objekten als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetBackColorTop

Funktion

Stellt bei Slider-Objekten die Hintergrundfarbe oben links ein.

Syntax

BOOL SetBackColorTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBackColorTop);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBackColorTop

Hintergrundfarbe von Slider-Objekten als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetBorderBackColor

Funktion

Stellt die Linien- oder Rahmenhintergrundfarbe ein.

Syntax

BOOL SetBorderBackColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBorderBackColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBorderBackColor

Linien- oder Rahmenhintergrundfarbe

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetBorderColor

Funktion

Stellt die Linien- oder Rahmenfarbe ein.

Syntax

BOOL SetBorderColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBorderColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBorderColor

Linien- oder Rahmenfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetBorderColorBottom

Funktion

Stellt die 3D-Rahmenfarbe unten ein.

Syntax

BOOL SetBorderColorBottom(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBorderColorBottom);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBorderColorBottom

3D-Rahmenfarbe unten als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetBorderColorTop

Funktion

Stellt die 3D-Rahmenfarbe oben ein.

Syntax

BOOL SetBorderColorTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBorderColorTop);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBorderColorTop

3D-Rahmenfarbe oben als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetButtonColor

Funktion

Stellt bei Slider-Objekten die Knopffarbe ein.

Syntax

BOOL SetButtonColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lButtonColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lButtonColor

Knopffarbe bei Slider-Objekten als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetColorBottom

Funktion

Stellt bei Slider-Objekten die Farbe des unteren Anschlags ein.

Syntax

BOOL SetColorBottom(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lColorBottom);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lColorBottom

Farbe des unteren Anschlags bei Slider-Objekten als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetColorTop

Funktion

Stellt bei Slider-Objekten die Farbe des oberen Anschlags ein.

Syntax

BOOL SetColorTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lColorTop);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lColorTop

Farbe des oberen Anschlags bei Slider-Objekten als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetFillColor

Funktion

Stellt die Farbe des Füllmusters ein.

Syntax

BOOL SetFillColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lFillColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lFillColor

Farbe des Füllmusters als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetForeColor

Funktion

Stellt die Schriftfarbe ein.

Syntax

BOOL SetForeColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lForeColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lForeColor

Schriftfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetItemBorderBackColor

Funktion

Stellt die Hintergrundfarbe der Trennlinie beim Objekt "Textliste" ein.

Syntax

BOOL SetItemBorderBackColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lItemBorderBackColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lItemBorderBackColor

Hintergrundfarbe der Trennlinie als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetItemBorderColor

Funktion

Stellt die Trennlinienfarbe beim Objekt "Textliste" ein.

Syntax

BOOL SetItemBorderColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lItemBorderColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lItemBorderColor

Trennlinienfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetScaleColor

Funktion

Stellt bei Balkenobjekten die Skalenfarbe ein.

Syntax

BOOL SetScaleColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lScaleColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lScaleColor

Skalenfarbe bei Balkenobjekten als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetSelBGColor

Funktion

Stellt beim Objekt "Textliste" die Hintergrundfarbe für den selektierten Eintrag ein.

Syntax

BOOL SetSelBGColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lSelBGColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lSelBGColor

Hintergrundfarbe für selektierten Eintrag als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetSelTextColor

Funktion

Stellt beim Objekt "Textliste" die Schriftfarbe für einen selektierten Eintrag ein.

Syntax

BOOL SetSelTextColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lSelTextColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lSelTextColor

Schriftfarbe für selektierten Eintrag als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetTrendColor

Funktion

Stellt bei Balkenobjekten die Trendfarbe ein.

Syntax

BOOL SetTrendColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lTrendColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lTrendColor

Trendfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetUnselBGColor

Funktion

Stellt beim Objekt "Textliste" die Hintergrundfarbe für nicht selektierte Einträge ein.

Syntax

BOOL SetUnselBGColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lUnselBGColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lUnselBGColor

Hintergrundfarbe für nicht selektierte Einträge als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetUnselTextColor

Funktion

Stellt beim Objekt "Textliste" die Schriftfarbe für nicht selektierte Einträge ein.

Syntax

BOOL SetUnselTextColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lUnselTextColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lUnselTextColor

Schriftfarbe für nicht selektierte Einträge als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

fill

Fill - Kurzbeschreibung

Die Funktionen der Gruppe Fill steuern das dynamische Füllen von Objekten.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetFilling

Funktion

Aktiviert oder deaktiviert das dynamische Füllen mit der Hintergrundfarbe.

Syntax

BOOL SetFilling(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bFilling);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bFilling

Dynamisches Füllen mit der Hintergrundfarbe ein/aus

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetFilling (Seite 1567)

Beispiel SetFilling

SetFillingIndex

Funktion

Stellt den Füllstand ein.

Syntax

BOOL SetFillingIndex(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lFillingIndex);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lFillingIndex

Füllstand als Zahlenwert (0 – 100)

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetFillingIndex (Seite 1568)

Beispiel SetFillingIndex

flash

Flash - Kurzbeschreibung

Mit den Funktionen der Gruppe Flash werden verschiedene, das Blinken betreffende Eigenschaften verändert bzw. abgefragt.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetBackFlashColorOff

Funktion

Stellt die Hintergrundblinkfarbe für den ausgeschalteten Zustand ein.

Syntax

BOOL SetBackFlashColorOff(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBackFlashColorOff);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBackFlashColorOff

Hintergrundblinkfarbe für den ausgeschalteten Zustand als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetFlashBackColorOn (Seite 1569)

Farbtabelle

Beispiel SetFlashBackColorOn

SetBackFlashColorOn

Funktion

Stellt die Hintergrundblinkfarbe für den eingeschalteten Zustand ein.

Syntax

BOOL SetBackFlashColorOn(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBackFlashColorOn);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBackFlashColorOn

Hintergrundblinkfarbe für den eingeschalteten Zustand als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetFlashBackColorOn (Seite 1569)

Farbtabelle

Beispiel SetFlashBackColorOn

SetBorderFlashColorOff

Funktion

Stellt die Rahmen- oder Linienblinkfarbe für den ausgeschalteten Zustand ein.

Syntax

```
BOOL SetBorderFlashColorOff(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName,
long int lBorderFlashColorOff);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBorderFlashColorOff

Rahmen- oder Linienblinkfarbe für den ausgeschalteten Zustand als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetFlashBackColorOn (Seite 1569)

Farbtabelle (Seite 1595)

Farbtabelle

Beispiel SetFlashBackColorOn

SetBorderFlashColorOn

Funktion

Stellt die Rahmen- oder Linienblinkfarbe für den eingeschalteten Zustand ein.

Syntax

```
BOOL SetBorderFlashColorOn(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName,
long int lBorderFlashColorOn);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBorderFlashColorOn

Rahmen- oder Linienblinkfarbe für den eingeschalteten Zustand als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetFlashBackColorOn (Seite 1569)

Farbtabelle

Beispiel SetFlashBackColorOn

SetFlashBackColor

Funktion

Aktiviert oder deaktiviert das Blinken des Hintergrundes.

Syntax

BOOL SetFlashBackColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bFlashBackColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bFlashBackColor

Blinken des Hintergrundes ein/aus

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetFlashBackColor (Seite 1568)

Beispiel SetFlashBackColor

SetFlashBorderColor

Funktion

Aktiviert oder deaktiviert das Blinken des Rahmens oder der Linie.

Syntax

BOOL SetFlashBorderColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bFlashBorderColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bFlashBorderColor

Blinken des Rahmens oder der Linie ein/aus

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetFlashBackColor (Seite 1568)

Farbtabelle

SetFlashForeColor

Funktion

Aktiviert oder deaktiviert das Blinken der Schrift.

Syntax

BOOL SetFlashForeColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bFlashForeColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bFlashForeColor

Blinken der Schrift ein/aus

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetFlashBackColor (Seite 1568)

Beispiel SetFlashBackColor

SetFlashRateBackColor

Funktion

Stellt die Blinkfrequenz des Hintergrunds ein.

Syntax

BOOL SetFlashRateBackColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lFlashRateBackColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lFlashRateBackColor

Blinkfrequenz des Hintergrunds

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Blinkfrequenzen (Seite 1593)

Beispiel SetFlashBackColor (Seite 1568)

Beispiel SetFlashBackColor

Blinkfrequenzen

SetFlashRateBorderColor

Funktion

Stellt die Blinkfrequenz der Linie oder des Rahmens ein.

Syntax

BOOL SetFlashRateBorderColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lFlashRateBorderColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lFlashRateBorderColor

Blinkfrequenz der Linie oder des Rahmens

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Blinkfrequenzen (Seite 1593)

Beispiel SetFlashBackColor (Seite 1568)

Beispiel SetFlashBackColor

Blinkfrequenzen

SetFlashRateForeColor

Funktion

Stellt die Blinkfrequenz der Schrift ein.

Syntax

BOOL SetFlashRateForeColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lFlashRateForeColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lFlashRateForeColor

Blinkfrequenz der Schrift

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Blinkfrequenzen (Seite 1593)

Beispiel SetFlashBackColor (Seite 1568)

Blinkfrequenzen

Beispiel SetFlashBackColor

SetForeFlashColorOff

Funktion

Stellt die Schriftblinkfarbe für den ausgeschalteten Zustand ein.

Syntax

BOOL SetForeFlashColorOff(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lForeFlashColorOff);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lForeFlashColorOff

Schriftblinkfarbe für den ausgeschalteten Zustand als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetFlashBackColorOn (Seite 1569)

Beispiel SetFlashBackColorOn

Farbtabelle

SetForeFlashColorOn

Funktion

Stellt die Schriftblinkfarbe für den eingeschalteten Zustand ein.

Syntax

BOOL SetForeFlashColorOn(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lForeFlashColorOn);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lForeFlashColorOn

Schriftblinkfarbe für den eingeschalteten Zustand als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetFlashBackColorOn (Seite 1569)

Beispiel SetFlashBackColorOn

Farbtabelle

focus

Focus - Kurzbeschreibung

Mit den Funktionen der Gruppe Focus können Sie den Fokus auf ein Objekt setzen bzw. abfragen, welches Objekt den Fokus besitzt.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

Set_Focus

Funktion

Setzt den Focus auf das angegebene Objekt.

Syntax

BOOL Set_Focus(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetFocus (Seite 1569)

Beispiel SetFocus

font

Font - Kurzbeschreibung

Mit den Funktionen der Gruppe Font werden verschiedene, den Text betreffende Eigenschaften verändert bzw. abgefragt.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetAlignmentLeft

Funktion

Stellt die Textausrichtung horizontal (links, zentriert, rechts) ein.

Syntax

BOOL SetAlignmentLeft(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lAlignmentLeft);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lAlignmentLeft

Textausrichtung horizontal als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Textausrichtung (Seite 1601)

Beispiel SetFontSize (Seite 1570)

Textausrichtung

Beispiel SetFontSize

SetAlignmentTop

Funktion

Stellt die Textausrichtung vertikal (oben, zentriert, unten) ein.

Syntax

BOOL SetAlignmentTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lAlignmentTop);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lAlignmentTop

Textausrichtung vertikal als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Textausrichtung (Seite 1601)

Beispiel SetFontSize (Seite 1570)

Textausrichtung

Beispiel SetFontSize

SetFontBold

Funktion

Schaltet den Schriftstil "fett" ein oder aus.

Syntax

BOOL SetFontBold(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bFontBold);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bFontBold

Schriftstil "fett" ein/aus

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetFontBold (Seite 1570)

Beispiel SetFontBold

SetFontItalic

Funktion

Schaltet den Schriftstil "kursiv" ein oder aus.

Syntax

BOOL SetFontItalic(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bFontItalic);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bFontItalic

Schriftstil "kursiv" ein/aus

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetFontBold (Seite 1570)

Beispiel SetFontBold

SetFontName

Funktion

Stellt eine Schriftart ein.

Syntax

BOOL SetFontName(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, char* szFontName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

szFontName

Zeiger auf den Namen der Schriftart

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetText (Seite 1584)

Beispiel SetText

SetFontSize

Funktion

Stellt den Schriftgrad ein.

Syntax

BOOL SetFontSize(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lFontSize);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lFontSize

Schriftgrad

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetFontSize (Seite 1570)

Beispiel SetFontSize

SetFontUnderline

Funktion

Schaltet den Schriftstil "unterstrichen" ein oder aus.

Syntax

BOOL SetFontUnderline(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bFontUnderline);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bFontUnderline

Schriftstil "unterstrichen" ein/aus

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetFontBold (Seite 1570)

Beispiel SetFontBold

SetOrientation

Funktion

Legt die Schreibrichtung (vertikal/horizontal) fest.

Syntax

BOOL SetOrientation(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bOrientation);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bOrientation

Schreibrichtung

TRUE	vertikal
FALSE	horizontal

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetFontBold (Seite 1570)

Beispiel SetFontBold

SetText

Funktion

Setzt bei Objekten wie statischem Text, Check- oder Radiobox den Wert der Eigenschaft "Text".

Syntax

BOOL SetText(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, char* szText);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

szText

Zeiger auf einen Text

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Bei Check- und Radio-Boxen ist vor Aufruf dieser Funktion das zu ändernde Element mit der Funktion SetIndex festzulegen.

Siehe auch

Beispiel SetText (Seite 1584)

Beispiel SetText

geometry

Geometry - Kurzbeschreibung

Mit den Funktionen der Gruppe Geometry können Sie die Größe, die Position und andere geometrische Eigenschaften von Objekten verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetActualPointLeft

Funktion

Stellt den X-Wert des aktuellen Punktes eines Polygons oder Polygonzugs ein.

Syntax

BOOL SetActualPointLeft(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lActualPointLeft);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lActualPointLeft

X-Wert des aktuellen Punktes eines Polygons oder Polygonzugs

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Mit der Funktion SetIndex kann der aktuelle Punkt des Polygons eingestellt werden.

Siehe auch

Beispiel SetLeft (Seite 1571)

Beispiel SetLeft

SetActualPointTop

Funktion

Stellt den Y-Wert des aktuellen Punktes eines Polygons oder Polygonzugs ein.

Syntax

BOOL SetActualPointTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lActualPointTop);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lActualPointTop

Y-Wert des aktuellen Punktes eines Polygons oder Polygonzugs

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Mit der Funktion SetIndex kann der aktuelle Punkt des Polygons eingestellt werden.

Siehe auch

Beispiel SetTop (Seite 1584)

Beispiel SetTop

SetBoxCount

Funktion

Stellt die Anzahl der Felder in einer Check- oder einer Radio-Box ein.

Syntax

BOOL SetBoxCount(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBoxCount);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBoxCount

Anzahl der Felder einer Check- oder einer Radio-Box.

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetDirection

Funktion

Stellt bei Balkenobjekten die Balkenrichtung ein.

Syntax

BOOL SetDirection(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lDirection);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lDirection

Balkenrichtung als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Balkenrichtung (Seite 1593)

Beispiel SetTop (Seite 1584)

Beispiel SetTop

Balkenrichtung

SetEndAngle

Funktion

Stellt den Endwinkel bei Kreis- und Ellipsensegmenten sowie bei Kreis- und Ellipsenbögen ein.

Syntax

BOOL SetEndAngle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lEndAngle);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lEndAngle

Endwinkel bei Kreis- und Ellipsensegmenten sowie bei Kreis- und Ellipsenbögen

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTop (Seite 1584)

Beispiel SetTop

SetHeight

Funktion

Stellt die Höhe des umschreibenden Rechtecks eines Objekts ein.

Syntax

BOOL SetHeight(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lHeight);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lHeight

Höhe des umschreibenden Rechtecks

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Beispiel SetHeight (Seite 1570)

Beispiel SetHeight

SetLeft

Funktion

Stellt den X-Wert der linken oberen Ecke des umschreibenden Rechtecks eines Objekts ein.

Syntax

BOOL SetLeft(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lLeft);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lLeft

X-Wert der linken oberen Ecke des umschreibenden Rechtecks

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetLeft (Seite 1571)

Beispiel SetLeft

SetPointCount

Funktion

Stellt die Anzahl der Eckpunkte eines Polygons oder eines Polygonzugs ein.

Syntax

BOOL SetPointCount(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lPointCount);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lPointCount

Anzahl der Eckpunkte

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetLeft (Seite 1571)

Beispiel SetLeft

SetRadius

Funktion

Stellt den Radius eines Kreises, Kreissegments oder Kreisbogens ein.

Syntax

BOOL SetRadius(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lRadius);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lRadius

Radius

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetHeight (Seite 1570)

Beispiel SetHeight

SetRadiusHeight

Funktion

Stellt den Radius einer Ellipse, eines Ellipsensegments oder eines Ellipsenbogens in vertikaler Richtung ein.

Syntax

BOOL SetRadiusHeight(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lRadiusHeight);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lRadiusHeight

Radius in vertikaler Richtung

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetHeight (Seite 1570)

Beispiel SetHeight

SetRadiusWidth

Funktion

Stellt den Radius einer Ellipse, eines Ellipsensegments oder eines Ellipsenbogens in horizontaler Richtung ein.

Syntax

BOOL SetRadiusWidth(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lRadiusWidth);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lRadiusWidth

Radius in horizontaler Richtung

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetWidth (Seite 1585)

Beispiel SetWidth

SetReferenceRotationLeft

Funktion

Stellt für Linie, Polygon und Polygonzug den X-Wert der Rotationsreferenz ein (Drehpunkt, um den das Objekt gedreht werden kann).

Syntax

BOOL SetReferenceRotationLeft(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lReferenceRotationLeft);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lReferenceRotationLeft

X-Wert der Rotationsreferenz

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetLeft (Seite 1571)

Beispiel SetLeft

SetReferenceRotationTop

Funktion

Stellt bei Linie, Polygon und Polygonzug den Y-Wert der Rotationsreferenz ein (Drehpunkt, um den das Objekt gedreht werden kann).

Syntax

BOOL SetReferenceRotationTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lReferenceRotationTop);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lReferenceRotationTop

Y-Wert der Rotationsreferenz

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTop (Seite 1584)

Beispiel SetTop

SetRotationAngle

Funktion

Stellt für Linie, Polygon und Polygonzug den Rotationswinkel um den Drehpunkt ein.

Syntax

BOOL SetRotationAngle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lRotationAngle);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lRotationAngle

Rotationswinkel

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetLeft (Seite 1571)

Beispiel SetLeft

SetRoundCornerHeight

Funktion

Stellt den vertikalen Radius der Ecke eines Rundrechtecks ein.

Syntax

BOOL SetRoundCornerHeight(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lRoundCornerHeight);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lRoundCornerHeight

Vertikaler Radius

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetHeight (Seite 1570)

Beispiel SetHeight

SetRoundCornerWidth

Funktion

Stellt den horizontalen Radius der Ecke eines Rundrechtecks ein.

Syntax

BOOL SetRoundCornerWidth(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lRoundCornerWidth);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lRoundCornerWidth

Horizontaler Radius

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetWidth (Seite 1585)

Beispiel SetWidth

SetStartAngle

Funktion

Stellt den Anfangswinkel bei Kreis- und Ellipsensegmenten sowie bei Kreis- und Ellipsenbögen ein.

Syntax

BOOL SetStartAngle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lStartAngle);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lStartAngle

Anfangswinkel

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetHeight (Seite 1570)

Beispiel SetHeight

SetTop

Funktion

Stellt den Y-Wert der linken oberen Ecke des umschreibenden Rechtecks eines Objekts ein.

Syntax

BOOL SetTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lTop);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lTop

Y-Wert der linken oberen Ecke des umschreibenden Rechtecks

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTop (Seite 1584)

Beispiel SetTop

SetWidth

Funktion

Stellt die Breite des umschreibenden Rechtecks eines Objekts ein.

Syntax

BOOL SetWidth(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lWidth);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lWidth

Breite des umschreibenden Rechtecks

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Beispiel SetWidth (Seite 1585)

Beispiel SetWidth

SetZeroPoint

Funktion

Stellt bei Balkenobjekten den Nullpunkt ein.

Syntax

BOOL SetZeroPoint(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lZeroPoint);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lZeroPoint

Nullpunkt

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTop (Seite 1584)

Beispiel SetTop

i_o

i_o - Kurzbeschreibung

Mit den Funktionen der Gruppe i_o werden verschiedene, die Ein- und Ausgabewerte betreffende Eigenschaften verändert bzw. abgefragt.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetAssumeOnExit

Funktion

Stellt bei E/A-Feldern ein, ob beim Verlassen des Feldes die Übernahme des eingegebenen Wertes erfolgt.

Syntax

BOOL SetAssumeOnExit(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bAssumeOnExit);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bAssumeOnExit

Wertübernahme beim Verlassen des Feldes ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetHiddenInput (Seite 1571)

Beispiel SetHiddenInput

SetAssumeOnFull

Funktion

Stellt bei E/A-Feldern ein, ob die Übernahme des eingegebenen Wertes bei vollständiger Eingabe erfolgt.

Syntax

BOOL SetAssumeOnFull(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bAssumeOnFull);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bAssumeOnFull

Wertübernahme bei vollständiger Eingabe ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetHiddenInput (Seite 1571)

Beispiel SetHiddenInput

SetBitNumber

Funktion

Stellt bei Listenart "bit" das relevante Bit im Ausgabewert ein.

Syntax

BOOL SetBitNumber(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBitNumber);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBitNumber

Relevantes Bit im Ausgabewert bei Listenart "bit"

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Listenarten (Seite 1599)

Listenarten

SetClearOnError

Funktion

Stellt bei E/A-Feldern ein, ob das Löschen des Inhalts bei fehlerhafter Eingabe aktiviert ist.

Syntax

BOOL SetClearOnError(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bClearOnError);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bClearOnError

Löschen der Eingabe bei fehlerhafter Eingabe ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetHiddenInput (Seite 1571)

Beispiel SetHiddenInput

SetClearOnNew

Funktion

Stellt bei E/A-Feldern das Löschen des Inhalts bei Neueingabe ein.

Syntax

BOOL SetClearOnNew(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bClearOnNew);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bClearOnNew

Löschen des Inhalts bei Neueingabe ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetHiddenInput (Seite 1571)

Beispiel SetHiddenInput

SetHiddenInput

Funktion

Steuert bei E/A-Feldern die verdeckte Eingabe.

Syntax

```
BOOL SetHiddenInput(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL
bHiddenInput);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bHiddenInput

Verdeckte Eingabe ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetHiddenInput (Seite 1571)

Beispiel SetHiddenInput

SetNumberLines

Funktion

Stellt beim Objekt "Textliste" die Anzahl der sichtbaren Zeilen ein.

Syntax

BOOL SetNumberLines(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lNumberLines);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lNumberLines

Anzahl der sichtbaren Zeilen

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Ist die Anzahl der projektierten Texte größer als die Anzahl der sichtbaren Zeilen, so erhält das Objekt "Textliste" eine vertikale Bildlaufleiste.

SetOutputValueChar

Funktion

Setzt bei E/A-Feldern einen Zeiger auf den Ausgabewert.

Syntax

BOOL SetOutputValueChar(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, char* szOutputValueChar);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

szOutputValueChar

Zeiger auf den Ausgabewert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetOutputValueDouble

Funktion

Stellt bei E/A-Feldern den Ausgabewert ein.

Syntax

BOOL SetOutputValueDouble(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dOutputValueDouble);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dOutputValueDouble

Ausgabewert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetOutputValueDouble (Seite 1573)

Beispiel SetOutputValueDouble

limits

Limits - Kurzbeschreibung

Mit den Funktionen der Gruppe Limits werden verschiedene, die Grenzwerte betreffende Eigenschaften verändert bzw. abgefragt.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetAlarmHigh

Funktion

Stellt bei Balkenobjekten die obere Alarmgrenze ein.

Syntax

BOOL SetAlarmHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dAlarmHigh);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dAlarmHigh

Obere Alarmgrenze

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetAlarmHigh (Seite 1565)

Beispiel SetAlarmHigh

SetAlarmLow

Funktion

Stellt bei Balkenobjekten die untere Alarmgrenze ein.

Syntax

BOOL SetAlarmLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dAlarmLow);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dAlarmLow

Untere Alarmgrenze

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetAlarmHigh (Seite 1565)

Beispiel SetAlarmHigh

SetCheckAlarmHigh

Funktion

Steuert bei Balkenobjekten die Überwachung der oberen Alarmgrenze.

Syntax

BOOL SetCheckAlarmHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bCheckAlarmHigh);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bCheckAlarmHigh

Überwachung ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetCheckAlarmLow

Funktion

Steuert bei Balkenobjekten die Überwachung der unteren Alarmgrenze.

Syntax

BOOL SetCheckAlarmLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bCheckAlarmLow);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bCheckAlarmLow

Überwachung ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetCheckLimitHigh4

Funktion

Steuert bei Balkenobjekten die Überwachung des oberen Grenzwerts Reserve 4.

Syntax

BOOL SetCheckLimitHigh4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bCheckLimitHigh4);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bCheckLimitHigh4

Überwachung ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetCheckLimitHigh5

Funktion

Steuert bei Balkenobjekten die Überwachung des oberen Grenzwerts Reserve 5.

Syntax

BOOL SetCheckLimitHigh5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bCheckLimitHigh5);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bCheckLimitHigh5

Überwachung ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetCheckLimitLow4

Funktion

Steuert bei Balkenobjekten die Überwachung des unteren Grenzwerts Reserve 4.

Syntax

BOOL SetCheckLimitLow4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bCheckLimitLow4);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bCheckLimitLow4

Überwachung ja/nein.

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetCheckLimitLow5

Funktion

Steuert bei Balkenobjekten die Überwachung des unteren Grenzwerts Reserve 5.

Syntax

BOOL SetCheckLimitLow5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bCheckLimitLow5);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bCheckLimitLow5

Überwachung ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetCheckToleranceHigh

Funktion

Steuert bei Balkenobjekten die Überwachung der oberen Toleranzgrenze.

Syntax

BOOL SetCheckToleranceHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bCheckToleranceHigh);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bCheckToleranceHigh

Überwachung ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetCheckToleranceLow

Funktion

Steuert bei Balkenobjekten die Überwachung der unteren Toleranzgrenze.

Syntax

BOOL SetCheckToleranceLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bCheckToleranceLow);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bCheckToleranceLow

Überwachung ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetCheckWarningHigh

Funktion

Steuert bei Balkenobjekten die Überwachung der oberen Warngrenze.

Syntax

BOOL SetCheckWarningHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bCheckWarningHigh);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bCheckWarningHigh

Überwachung ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetCheckWarningLow

Funktion

Steuert bei Balkenobjekten die Überwachung der unteren Warngrenze.

Syntax

BOOL SetCheckWarningLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bCheckWarningLow);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bCheckWarningLow

Überwachung ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetColorAlarmHigh

Funktion

Stellt bei Balkenobjekten die Balkenfarbe bei Erreichen der oberen Alarmgrenze ein.

Syntax

BOOL SetColorAlarmHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lColorAlarmHigh);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lColorAlarmHigh

Balkenfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetColorAlarmLow

Funktion

Stellt bei Balkenobjekten die Balkenfarbe bei Erreichen der unteren Alarmgrenze ein.

Syntax

BOOL SetColorAlarmLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lColorAlarmLow);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lColorAlarmLow

Balkenfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetColorLimitHigh4

Funktion

Stellt bei Balkenobjekten die Balkenfarbe bei Erreichen der oberen Grenze Reserve 4 ein.

Syntax

BOOL SetColorLimitHigh4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lColorLimitHigh4);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lColorLimitHigh4

Balkenfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetColorLimitHigh5

Funktion

Stellt bei Balkenobjekten die Balkenfarbe bei Erreichen der oberen Grenze Reserve 5 ein.

Syntax

BOOL SetColorLimitHigh5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lColorLimitHigh5);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lColorLimitHigh5

Balkenfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetColorLimitLow4

Funktion

Stellt bei Balkenobjekten die Balkenfarbe bei Erreichen der unteren Grenze Reserve 4 ein.

Syntax

BOOL SetColorLimitLow4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lColorLimitLow4);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lColorLimitLow4

Balkenfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetColorLimitLow5

Funktion

Stellt bei Balkenobjekten die Balkenfarbe bei Erreichen der unteren Grenze Reserve 5 ein.

Syntax

BOOL SetColorLimitLow5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lColorLimitLow5);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lColorLimitLow5

Balkenfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetColorToleranceHigh

Funktion

Stellt bei Balkenobjekten die Balkenfarbe bei Erreichen der oberen Toleranzgrenze ein.

Syntax

BOOL SetColorToleranceHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lColorToleranceHigh);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lColorToleranceHigh

Balkenfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetColorToleranceLow

Funktion

Stellt bei Balkenobjekten die Balkenfarbe bei Erreichen der unteren Toleranzgrenze ein.

Syntax

BOOL SetColorToleranceLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lColorToleranceLow);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lColorToleranceLow

Balkenfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetColorWarningHigh

Funktion

Stellt bei Balkenobjekten die Balkenfarbe bei Erreichen der oberen Warngrenze ein.

Syntax

BOOL SetColorWarningHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lColorWarningHigh);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lColorWarningHigh

Balkenfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetColorWarningLow

Funktion

Stellt bei Balkenobjekten die Balkenfarbe bei Erreichen der unteren Warngrenze ein.

Syntax

BOOL SetColorWarningLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lColorWarningLow);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lColorWarningLow

Balkenfarbe als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetLimitHigh4

Funktion

Stellt bei Balkenobjekten den oberen Grenzwert für Reserve 4 ein.

Syntax

BOOL SetLimitHigh4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dLimitHigh4);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dLimitHigh4

Oberer Grenzwert für Reserve 4

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetAlarmHigh (Seite 1565)

Beispiel SetAlarmHigh

SetLimitHigh5

Funktion

Stellt bei Balkenobjekten den oberen Grenzwert für Reserve 5 ein.

Syntax

BOOL SetLimitHigh5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dLimitHigh5);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dLimitHigh5

Oberer Grenzwert für Reserve 5

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetAlarmHigh (Seite 1565)

Beispiel SetAlarmHigh

SetLimitLow4

Funktion

Stellt bei Balkenobjekten den unteren Grenzwert für Reserve 4 ein.

Syntax

BOOL SetLimitLow4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dLimitLow4);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dLimitLow4

Unterer Grenzwert für Reserve 4

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetAlarmHigh (Seite 1565)

Beispiel SetAlarmHigh

SetLimitLow5

Funktion

Stellt bei Balkenobjekten den unteren Grenzwert für Reserve 5 ein.

Syntax

BOOL SetLimitLow5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dLimitLow5);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dLimitLow5

Unterer Grenzwert für Reserve 5

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetAlarmHigh (Seite 1565)

Beispiel SetAlarmHigh

SetLimitMax

Funktion

Stellt bei E/A-Feldern den oberen Grenzwert ein.

Syntax

BOOL SetLimitMax(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dLimitMax);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dLimitMax

Oberer Grenzwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetAlarmHigh (Seite 1565)

Beispiel SetAlarmHigh

SetLimitMin

Funktion

Stellt bei E/A-Feldern den unteren Grenzwert ein.

Syntax

BOOL SetLimitMin(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dLimitMin);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dLimitMin

Unterer Grenzwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetAlarmHigh (Seite 1565)

Beispiel SetAlarmHigh

SetMarker

Funktion

Steuert bei Balkenobjekten die Anzeige des Grenzwertmarkierers.

Syntax

BOOL SetMarker(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bMarker);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bMarker

Grenzwertmarkierer ein/aus

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetToleranceHigh

Funktion

Stellt bei Balkenobjekten die obere Toleranzgrenze ein.

Syntax

BOOL SetToleranceHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dToleranceHigh);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dToleranceHigh

Obere Toleranzgrenze

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetAlarmHigh (Seite 1565)

Beispiel SetAlarmHigh

SetToleranceLow

Funktion

Stellt bei Balkenobjekten die untere Toleranzgrenze ein.

Syntax

BOOL SetToleranceLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dToleranceLow);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dToleranceLow

Untere Toleranzgrenze

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetAlarmHigh (Seite 1565)

Beispiel SetAlarmHigh

SetTypeAlarmHigh

Funktion

Steuert bei Balkenobjekten, ob die obere Alarmgrenze prozentual oder absolut angegeben wird.

Syntax

BOOL SetTypeAlarmHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bTypeAlarmHigh);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bTypeAlarmHigh

Obere Alarmgrenze

TRUE	Angabe prozentual
FALSE	Angabe absolut

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetTypeAlarmLow

Funktion

Steuert bei Balkenobjekten, ob die untere Alarmgrenze prozentual oder absolut angegeben wird.

Syntax

BOOL SetTypeAlarmLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bTypeAlarmLow);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bTypeAlarmLow

Untere Alarmgrenze

TRUE	Angabe prozentual
FALSE	Angabe absolut

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetTypeLimitHigh4

Funktion

Steuert bei Balkenobjekten, ob die obere Grenze für Reserve 4 prozentual oder absolut angegeben wird.

Syntax

BOOL SetTypeLimitHigh4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bTypeLimitHigh4);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bTypeLimitHigh4

Obere Grenze

TRUE	Angabe prozentual
FALSE	Angabe absolut

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetTypeLimitHigh5

Funktion

Steuert bei Balkenobjekten, ob die obere Grenze für Reserve 5 prozentual oder absolut angegeben wird.

Syntax

BOOL SetTypeLimitHigh5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bTypeLimitHigh5);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bTypeLimitHigh5

Obere Grenze

TRUE	Angabe prozentual
FALSE	Angabe absolut

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetTypeLimitLow4

Funktion

Steuert bei Balkenobjekten, ob die untere Grenze für Reserve 4 prozentual oder absolut angegeben wird.

Syntax

BOOL SetTypeLimitLow4(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bTypeLimitLow4);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bTypeLimitLow4

Untere Grenze

TRUE	Angabe prozentual
FALSE	Angabe absolut

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetTypeLimitLow5

Funktion

Steuert bei Balkenobjekten, ob die untere Grenze für Reserve 5 prozentual oder absolut angegeben wird.

Syntax

BOOL SetTypeLimitLow5(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bTypeLimitLow5);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bTypeLimitLow5

Untere Grenze

TRUE	Angabe prozentual
FALSE	Angabe absolut

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetTypeToleranceHigh

Funktion

Steuert bei Balkenobjekten, ob die obere Toleranzgrenze prozentual oder absolut angegeben wird.

Syntax

BOOL SetTypeToleranceHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bTypeToleranceHigh);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bTypeToleranceHigh

Obere Toleranzgrenze

TRUE	Angabe prozentual
FALSE	Angabe absolut

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetTypeToleranceLow

Funktion

Steuert bei Balkenobjekten, ob die untere Toleranzgrenze prozentual oder absolut angegeben wird.

Syntax

BOOL SetTypeToleranceLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bTypeToleranceLow);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bTypeToleranceLow

Untere Toleranzgrenze

TRUE	Angabe prozentual
FALSE	Angabe absolut

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetTypeWarningHigh

Funktion

Steuert bei Balkenobjekten, ob die obere Warngrenze prozentual oder absolut angegeben wird.

Syntax

BOOL SetTypeWarningHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bTypeWarningHigh);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bTypeWarningHigh

Obere Warngrenze

TRUE	Angabe prozentual
FALSE	Angabe absolut

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetTypeWarningLow

Funktion

Steuert bei Balkenobjekten, ob die untere Warngrenze prozentual oder absolut angegeben wird.

Syntax

BOOL SetTypeWarningLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bTypeWarningLow);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bTypeWarningLow

Untere Warngrenze

TRUE	Angabe prozentual
FALSE	Angabe absolut

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetWarningHigh

Funktion

Stellt bei Balkenobjekten die obere Warngrenze ein.

Syntax

BOOL SetWarningHigh(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dWarningHigh);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dWarningHigh

Obere Warngrenze

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

SetWarningLow

Funktion

Stellt bei Balkenobjekten die untere Warngrenze ein.

Syntax

BOOL SetWarningLow(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dWarningLow);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dWarningLow

Untere Warngrenze

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetMarker (Seite 1572)

Beispiel SetMarker

link

Link - Kurzbeschreibung

Mit den Funktionen der Gruppe Link kann die Variablenanbindung einer Eigenschaft erzeugt bzw. abgefragt werden.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetLink

Funktion

Erstellen einer Variablenverbindung von Objekteigenschaften

Syntax

BOOL SetLink(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, LPCTSTR lpszPropertyName, LPLINKINFO *pLink);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lpszPropertyName

Name der Objekteigenschaft

pLink

Zeiger auf eine Struktur vom Typ: LINKINFO

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Strukturdefinition LINKINFO (Seite 1608)

Beispiel SetLink (Seite 1572)

Strukturdefinition LINKINFO

Beispiel SetLink

miscs

Miscs - Kurzbeschreibung

Mit den Funktionen der Gruppe Miscs können Sie verschiedene Eigenschaften von Objekten verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetAverage

Funktion

Steuert bei Balkenobjekten die Mittelwertbildung.

Syntax

BOOL SetAverage(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bAverage);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bAverage

Mittelwertbildung ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetVisible (Seite 1584)

Beispiel SetVisible

SetBoxType

Funktion

Legt für ein E/A-Objekt den Feldtyp (Eingabefeld, Ausgabefeld, Ein-/Ausgabefeld) fest.

Syntax

BOOL SetBoxType(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBoxType);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBoxType

Feldtyp

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

E/A-Feld, Feldtyp (Seite 1595)

E/A-Feld, Feldtyp

SetColorChangeType

Funktion

Legt bei Balkenobjekten fest, ob der Farbumschlag bei Erreichen eines Grenzwertes nur in einem Balkensegment erfolgt oder den gesamten Balken betrifft.

Syntax

```
BOOL SetColorChangeType(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName,
BOOL bColorChangeType);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bColorChangeType

Typ des Farbumschlags

TRUE	Farbumschlag wirkt auf ein Segment
FALSE	Farbumschlag wirkt auf den gesamten Balken

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetVisible (Seite 1584)

Beispiel SetVisible

SetCursorControl

Funktion

Stellt bei E/A-Feldern die Cursorsteuerung ein.

Syntax

BOOL SetCursorControl(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bCursorControl);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bCursorControl

Cursorsteuerung ein/aus

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetVisible (Seite 1584)

Beispiel SetVisible

SetCursorMode

Funktion

Stellt bei Bildern die Cursorsteuerung ein.

Syntax

BOOL SetCursorMode(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bCursorMode);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bCursorMode

Cursormodus

TRUE	Schalt-Cursor
FALSE	Alpha-Cursor

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetCursorMode (Seite 1567)

Beispiel SetCursorMode

SetEditAtOnce

Funktion

Stellt bei E/A-Feldern ein, ob die Eigenschaft "Eingabe sofort" aktiviert ist.

Syntax

BOOL SetEditAtOnce(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bEditAtOnce);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bEditAtOnce

Eingabe sofort ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetVisible (Seite 1584)

Beispiel SetVisible

SetExtendedOperation

Funktion

Steuert bei Slider-Objekten die Eigenschaft "Erweiterte Bedienung".

Syntax

BOOL SetExtendedOperation(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bExtendedOperation);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bExtendedOperation

Erweiterte Bedienung ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetVisible (Seite 1584)

Beispiel SetVisible

SetHysteresis

Funktion

Steuert bei Balkenobjekten, ob die Anzeige mit oder ohne Hysterese erfolgt.

Syntax

BOOL SetHysteresis(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bHysteresis);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bHysteresis

Anzeige mit/ohne Hysterese

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetVisible (Seite 1584)

Beispiel SetVisible

SetHysteresisRange

Funktion

Stellt bei Balkenobjekten den Wert der Hysterese in der Anzeige ein.

Syntax

BOOL SetHysteresisRange(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dHysteresisRange);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dHysteresisRange

Wert der Hysterese

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetMax

Funktion

Stellt bei Balken- oder Slider-Objekten den Maximalwert ein.

Syntax

BOOL SetMax(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dMax);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dMax

Maximalwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetMin

Funktion

Stellt bei Balken- oder Slider-Objekten den Minimalwert ein.

Syntax

BOOL SetMin(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dMin);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dMin

Minimalwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetOffsetLeft

Funktion

Stellt bei Bildfenstern den horizontalen Bildabstand vom linken Fensterrand ein.

Syntax

BOOL SetOffsetLeft(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lOffsetLeft);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lOffsetLeft

Bildabstand

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetOffsetTop

Funktion

Stellt bei Bildfenstern den vertikalen Bildabstand vom oberen Fensterrand ein.

Syntax

BOOL SetOffsetTop(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lOffsetTop);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lOffsetTop

Bildabstand

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetOperation

Funktion

Steuert die Bedienbarkeit von Objekten.

Syntax

BOOL SetOperation(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bOperation);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bOperation

Objekt bedienbar ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Bezieht sich der Aufruf der Funktion auf das Bildobjekt, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Beispiel SetVisible (Seite 1584)

Beispiel SetVisible

SetOperationMessage

Funktion

Steuert die Ausgabe einer Meldung bei Bedienung der Objekte "E/A-Feld", "Check-Box", "Radio-Box" und "Slider".

Syntax

BOOL SetOperationMessage(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bOperationMessage);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bOperationMessage

Ausgabe einer Meldung bei Bedienung ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetVisible (Seite 1584)

Beispiel SetVisible

SetOperationReport

Funktion

Steuert die Protokollierung des Bediengrundes für alle Objekte außer Applikations- und Bildfenstern sowie OLE-Control.

Syntax

BOOL SetOperationReport(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bOperationReport);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bOperationReport

Protokollierung des Bediengrundes ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Bezieht sich der Aufruf der Funktion auf das Bildobjekt, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Beispiel SetVisible (Seite 1584)

Beispiel SetVisible

SetPasswordLevel

Funktion

Stellt die Berechtigungsstufe zur Bedienung von Objekten für alle Objekte, außer Applikations- und Bildfenstern sowie OLE-Control, ein.

Syntax

BOOL SetPasswordLevel(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lPasswordLevel);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lPasswordLevel

Berechtigungsstufe

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

SetPictureName

Funktion

Setzt den Namen des Bildes, das in einem Bildfenster oder einem Grafik-Objekt angezeigt werden soll.

Syntax

BOOL SetPictureName(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, char* szPictureName);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Bildfensters oder des Grafik-Objekts

szPictureName

Zeiger auf den Bildnamen

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetPictureName (Seite 1573)

Beispiel SetPictureName

SetProcess

Funktion

Setzt bei Balken- und Slider-Objekten die Voreinstellung für den anzuzeigenden Wert.

Setzt bei Check-Boxen und Radio-Boxen die selektierten Felder.

Syntax

BOOL SetProcess(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dProcess);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dProcess

- Bei Balken und Slider-Objekten wird im Runtime dieser Wert verwendet, wenn beim Start des Bildes die zugehörige Variable nicht angeschlossen oder nicht aktualisiert ist.
- Bei Check- und Radio-Boxen werden die selektierten Felder festgelegt. In dem 32 Bit-Wort wird jedes Feld durch ein Bit repräsentiert (Feld 1 entspricht der Bit-Wertigkeit 0). Selektierte Felder werden durch ein gesetztes Bit markiert. Nicht vorhandene Felder werden mit 0 belegt.

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetSmallChange

Funktion

Stellt für Slider-Objekte die Anzahl der Schritte ein, um die der Schieber bei einem Mausklick verschoben wird.

Syntax

BOOL SetSmallChange(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lSmallChange);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lSmallChange

Anzahl der Einstellschritte

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetTagPrefix

Funktion

Diese Funktion setzt bei Bildfenstern das Variablen-Präfix eines Bildfensters:

In einem Bildfenster wird an einem Objekt die Variable "Temperatur" angefordert. Wenn dem Bildfenster ein Variablen-Präfix "Motor1." zugeordnet ist, dann wird die Variable "Motor1.Temperatur" angefordert.

Das Setzen des Variablen-Präfixes wird erst wirksam, wenn der Bildname neu versorgt wird.

Sie müssen also entweder das Präfix vor Bildanwahl setzen oder den Bildnamen neu versorgen, falls das Bild nicht gewechselt wird.

Syntax

BOOL SetTagPrefix(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName,char* szTagPrefix);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

szTagPrefix

Zu setzendes Variablen-Präfix

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Wenn das Variablen-Präfix für ein Bildfenster gesetzt ist, wird das Variablen-Präfix allen Variablen vorangestellt, die im anzuzeigenden Bild enthalten sind. Das gilt auch, wenn die Anforderung in einer Funktion erfolgt. Wenn eine Variable ohne Variablen-Präfix gelesen werden soll, müssen Sie dem Variablennamen den Zusatz "@NOTP::" voranstellen.

Durch den Einsatz eines Variablen-Präfixes wird die Bildbausteintechnik stark vereinfacht.

Siehe auch

Beispiel SetTagPrefix (Seite 1580)

Beispiel SetTagPrefix

SetTrend

Funktion

Steuert bei Balkenobjekten die Trendanzeige.

Syntax

BOOL SetTrend(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bTrend);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bTrend

Trendanzeige ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetVisible (Seite 1584)

Beispiel SetVisible

SetVisible

Funktion

Steuert die Anzeige eines Objekts.

Syntax

BOOL SetVisible(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bVisible);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bVisible

Objektanzeige ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Beispiel SetVisible (Seite 1584)

Beispiel SetVisible

SetZeroPointValue

Funktion

Stellt bei Balkenobjekten den Absolutwert des Nullpunkts ein.

Syntax

BOOL SetZeroPointValue(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, double dZeroPointValue);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

dZeroPointValue

Absolutwert des Nullpunkts

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetZoom

Funktion

Stellt den Skalierungsfaktor für ein Bildfenster ein.

Syntax

BOOL SetZoom(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lZoom);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lZoom

Skalierungsfaktor

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

ole_control

OLE_control - Kurzbeschreibung

Die Funktionen der Gruppe ole_Control sind nur auf OCX-Slider-Objekte anwendbar.

Mit diesen Funktionen können Sie verschiedene Eigenschaften und Einstellungen eines OCX-Slider-Objekts verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetPosition

Funktion

Stellt die Schieberposition des OCX-Slider-Objekts ein.

Syntax

BOOL SetPosition(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lPosition);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lPosition

Schieberposition des OCX-Slider-Objekts

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetPosition (Seite 1574)

Beispiel SetPosition

SetRangeMax

Funktion

Legt den Einstellbereich "Max" des OCX-Slider-Objekts fest.

Syntax

BOOL SetRangeMax(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lRangeMax);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lRangeMax

Einstellbereich "Max" des OCX-Slider-Objekts

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetRangeMax (Seite 1575)

Beispiel SetRangeMax

SetRangeMin

Funktion

Legt den Einstellbereich "Min" des OCX-Slider-Objekts fest.

Syntax

BOOL SetRangeMin(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lRangeMin);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lRangeMin

Einstellbereich "Min" des OCX-Slider-Objekts

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetRangeMin (Seite 1575)

Beispiel SetRangeMin

pictures

Pictures - Kurzbeschreibung

Mit den Funktionen der Gruppe Pictures können Sie verschiedene Eigenschaften der Bilder von Grafik-Objekten und Rundbuttons verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetPicDeactTransparent

Funktion

Stellt bei einem Rundbutton die Transparentfarbe für den Zustand "deaktiviert" ein.

Syntax

BOOL SetPicDeactTransparent(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lPicDeactTransparent);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lPicDeactTransparent

Transparentfarbe für Zustand "deaktiviert"

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Diese Funktion gilt nur für Bitmap-Grafiken (*.bmp).

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetPicDeactUseTransColor

Funktion

Steuert bei einem Rundbutton die Transparentfarbe für den Zustand "deaktiviert".

Syntax

BOOL SetPicDeactUseTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bPicDeactUseTransColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bPicDeactUseTransColor

Transparentfarbe ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetPicDownTransparent

Funktion

Stellt bei einem Rundbutton die Transparentfarbe für den Zustand "Ein/gedrückt" ein.

Syntax

BOOL SetPicDownTransparent(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lPicDownTransparent);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lPicDownTransparent

Transparentfarbe für Zustand "Ein/gedrückt"

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Diese Funktion gilt nur für Bitmap-Grafiken (*.bmp).

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetPicDownUseTransColor

Funktion

Steuert bei einem Rundbutton die Transparentfarbe für den Zustand "Ein/gedrückt".

Syntax

BOOL SetPicDownUseTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bPicDownUseTransColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bPicDownUseTransColor

Transparentfarbe ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetPicTransColor

Funktion

Stellt die Transparentfarbe des Hintergrundbildes eines Grafik-Objekts ein.

Syntax

BOOL SetPicTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lPicTransColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lPicTransColor

Transparentfarbe des Hintergrundbildes

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Diese Funktion gilt nur für Bitmap-Grafiken (*.bmp).

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetPictureDeactivated

Funktion

Legt bei einem Rundbutton den Bildnamen für den Zustand "deaktiviert" fest.

Syntax

BOOL SetPictureDeactivated(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, char* szPictureDeactivated);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

szPictureDeactivated

Bildname für Zustand "deaktiviert"

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Es können Bitmap-Dateien (*.bmp, *.dib) sowie Metafiles (*.emf, *.wmf) eingebunden werden.

Siehe auch

Beispiel SetPictureDown (Seite 1573)

Beispiel SetPictureDown

SetPictureDown

Funktion

Legt bei einem Rundbutton den Bildnamen für den Zustand "Ein/gedrückt" fest.

Syntax

```
BOOL SetPictureDown(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, char*
szPictureDown);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

szPictureDown

Bildname für Zustand "Ein/gedrückt"

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Es können Bitmap-Dateien (*.bmp, *.dib) sowie Metafiles (*.emf, *.wmf) eingebunden werden.

Siehe auch

Beispiel SetPictureDown (Seite 1573)

Beispiel SetPictureDown

SetPictureUp

Funktion

Legt bei einem Rundbutton den Bildnamen für den Zustand "Aus/nicht gedrückt" fest.

Syntax

BOOL SetPictureUp(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, char* szPictureUp);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

szPictureUp

Bildname für Zustand "Aus/nicht gedrückt"

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Es können Bitmap-Dateien (*.bmp, *.dib) sowie Metafiles (*.emf, *.wmf) eingebunden werden.

Siehe auch

Beispiel SetPictureUp (Seite 1574)

Beispiel SetPictureUp

SetPicUpTransparent

Funktion

Stellt bei einem Rundbutton die Transparentfarbe für den Zustand "Aus/nicht gedrückt" ein.

Syntax

BOOL SetPicUpTransparent(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lPicUpTransparent);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lPicUpTransparent

Transparentfarbe für den Zustand "Aus/nicht gedrückt"

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Diese Funktion gilt nur für Bitmap-Grafiken (*.bmp).

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetPicUpUseTransColor

Funktion

Steuert bei einem Rundbutton die Transparentfarbe für den Zustand "Aus/nicht gedrückt".

Syntax

BOOL SetPicUpUseTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bPicUpUseTransColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bPicUpUseTransColor

Transparentfarbe ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetPicUseTransColor

Funktion

Steuert die Transparentfarbe des Hintergrundbildes eines Grafik-Objektes.

Syntax

BOOL SetPicUseTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bPicUseTransColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bPicUseTransColor

Transparentfarbe ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

property

Property - Kurzbeschreibung

Mit den Funktionen der Gruppe Property können Sie Eigenschaften von Objekten, für die es keine eigene Funktion gibt, verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetPropBOOL

Funktion

Setzt eine Eigenschaft mit dem Wert "bValue".

Syntax

BOOL SetPropBOOL(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, LPCTSTR lpszPropertyName, BOOL bValue)

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lpszPropertyName

Name der Objekteigenschaft

bValue

Wert im Datenformat BOOL

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Bezieht sich der Aufruf der Funktion auf das Bild-Objekt, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Beispiel SetPropBOOL (Seite 1574)

SetPropChar

Funktion

Setzt eine Eigenschaft mit dem Wert, auf den der Zeiger "szValue" zeigt.

Syntax

BOOL SetPropChar(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, LPCTSTR lpszPropertyName, char* szValue)

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lpszPropertyName

Name der Objekteigenschaft

szValue

Zeiger auf den Wert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Bezieht sich der Aufruf der Funktion auf das Bild-Objekt, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Beispiel GetPropChar (Seite 1541)

SetPropDouble

Funktion

Setzt eine Eigenschaft mit dem Wert "dValue".

Syntax

BOOL SetPropDouble(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, LPCTSTR lpszPropertyName, double dValue)

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lpszPropertyName

Name der Objekteigenschaft

dValue

Wert im Datenformat "double"

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Bezieht sich der Aufruf der Funktion auf das Bild-Objekt, ist der Parameter lpszObjectName = NULL zu setzen.

SetPropWord

Funktion

Setzt eine Eigenschaft mit dem Wert "lValue".

Syntax

BOOL SetPropWord(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, LPCTSTR lpszPropertyName, long lValue)

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lpszPropertyName

Name der Objekteigenschaft

lValue

Wert im Datenformat "long"

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Bezieht sich der Aufruf der Funktion auf das Bild-Objekt, ist der Parameter lpszObjectName = NULL zu setzen.

state

State - Kurzbeschreibung

Mit den Funktionen der Gruppe State können Sie verschiedene Eigenschaften von Zustandsanzeigen verändern bzw. abfragen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetBasePicTransColor

Funktion

Stellt bei der Zustandsanzeige die Transparentfarbe des Grundbildes ein.

Syntax

BOOL SetBasePicTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBasePicTransColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBasePicTransColor

Transparentfarbe des Grundbildes

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Diese Funktion gilt nur für Bitmap-Grafiken (*.bmp).

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Farbtabelle

Beispiel SetBackColor

SetBasePicUseTransColor

Funktion

Steuert die Transparentfarbe des Grundbildes einer Zustandsanzeige.

Syntax

BOOL SetBasePicUseTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bBasePicUseTransColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bBasePicUseTransColor

Transparentfarbe ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetFlashFlashPicture

Funktion

Stellt bei der Zustandsanzeige ein, ob das Blinkbild dynamisch oder statisch animiert wird.

Syntax

BOOL SetFlashFlashPicture(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bFlashFlashPicture);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bFlashFlashPicture

Art des Blinkbildes

TRUE	dynamisch animiertes Blickbild
FALSE	statisch animiertes Blinkbild

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetFlashPicTransColor

Funktion

Stellt bei einer Zustandsanzeige die Transparentfarbe des Blinkbildes ein.

Syntax

BOOL SetFlashPicTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lFlashPicTransColor);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lFlashPicTransColor

Transparentfarbe des Blinkbildes

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Diese Funktion gilt nur für Bitmap-Grafiken (*.bmp).

Siehe auch

Farbtabelle (Seite 1595)

Beispiel SetBackColor (Seite 1565)

Beispiel SetBackColor

Farbtabelle

SetFlashPicUseTransColor

Funktion

Steuert bei einer Zustandsanzeige die Transparentfarbe des Blinkbildes.

Syntax

```
BOOL SetFlashPicUseTransColor(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName,
BOOL bFlashPicUseTransColor);
```

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bFlashPicUseTransColor

Transparentfarbe ja/nein

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetFlashRateFlashPic

Funktion

Stellt bei einer Zustandsanzeige die Blinkfrequenz des Blinkbildes ein.

Syntax

BOOL SetFlashRateFlashPic(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lFlashRateFlashPic);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lFlashRateFlashPic

Blinkfrequenz des Blinkbildes

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Da es sich beim Blinken um eine softwaretechnische Realisierung handelt, ist die Frequenz system- und hardwareabhängig (Anzahl der Objekte, Prozessor, Speicher, Aktualisierungszeit u. a. m.).

Siehe auch

Blinkfrequenzen (Seite 1593)

Beispiel SetFlashRateFlashPic (Seite 1569)

Blinkfrequenzen

SetIndex

Funktion

Setzt den Index eines Polygons oder Polygonzugs und stellt damit den aktuellen Punkt des Objekts ein.

Syntax

BOOL SetIndex(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lIndex);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lIndex

Indexwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

style

Style - Kurzbeschreibung

Mit den Funktionen der Gruppe Style werden verschiedene, das Aussehen von Objekten betreffende Eigenschaften verändert bzw. abgefragt.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

SetBackBorderWidth

Funktion

Stellt die Breite der Umrahmung von 3D-Rahmen und Slider-Objekten ein.

Syntax

BOOL SetBackBorderWidth(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBackBorderWidth);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBackBorderWidth

Breite der Umrahmung in Pixel

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetBorderStyle (Seite 1566)

Beispiel SetBorderStyle

SetBorderEndStyle

Funktion

Stellt die Art des Linienendes ein.

Syntax

BOOL SetBorderEndStyle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBorderEndStyle);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBorderEndStyle

Art des Linienendes als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Linienenden (Seite 1599)

Beispiel SetBorderEndStyle (Seite 1565)

Beispiel SetBorderEndStyle

Linienenden

SetBorderStyle

Funktion

Stellt die Linien- oder Rahmenart ein.

Syntax

BOOL SetBorderStyle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBorderStyle);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBorderStyle

Linien- oder Rahmenart als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Linienarten (Seite 1598)

Beispiel SetBorderStyle (Seite 1566)

Beispiel SetBorderStyle

Linienarten

SetBorderWidth

Funktion

Stellt die Breite der Linien oder der Rahmenlinie ein.

Syntax

BOOL SetBorderWidth(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBorderWidth);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBorderWidth

Linienbreite oder Breite der Rahmenlinie

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetBorderStyle (Seite 1566)

Beispiel SetBorderStyle

SetBoxAlignment

Funktion

Stellt die Anordnung der Bedienelemente (links- oder rechtsbündig) in Check- oder Radio-Boxen ein.

Syntax

BOOL SetBoxAlignment(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lBoxAlignment);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lBoxAlignment

Anordnung der Bedienelemente

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Elementausrichtung in Check- und Radioboxen (Seite 1595)

Beispiel SetBorderStyle (Seite 1566)

Beispiel SetBorderStyle

Elementausrichtung in Check- und Radioboxen

SetFillStyle

Funktion

Stellt die Art des Füllmusters ein.

Syntax

BOOL SetFillStyle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lFillStyle);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lFillStyle

Art des Füllmusters als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Hinweis

Bezieht sich der Aufruf der Funktion auf das Gesamtbild, ist der Parameter lpszObjectName = NULL zu setzen.

Siehe auch

Füllmuster (Seite 1597)

Beispiel SetFillStyle (Seite 1568)

Füllmuster

Beispiel SetFillStyle

SetFillStyle2

Funktion

Stellt bei einer Balkenanzeige das Füllmuster des Balkens ein.

Syntax

BOOL SetFillStyle2(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lFillStyle2);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lFillStyle2

Füllmuster des Balkens als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Füllmuster (Seite 1597)

Beispiel SetFillStyle (Seite 1568)

Füllmuster

Beispiel SetFillStyle

SetItemBorderStyle

Funktion

Stellt die Trennlinienart beim Objekt "Textliste" ein.

Syntax

BOOL SetItemBorderStyle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lItemBorderStyle);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lItemBorderStyle

Trennlinienart als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Linienarten (Seite 1598)

Beispiel SetBorderStyle (Seite 1566)

Linienarten

Beispiel SetBorderStyle

SetItemBorderWidth

Funktion

Stellt die Trennlinienbreite beim Objekt "Textliste" ein.

Syntax

BOOL SetItemBorderWidth(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, long int lItemBorderWidth);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

lItemBorderWidth

Trennlinienbreite als Zahlenwert

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetBorderStyle (Seite 1566)

Beispiel SetBorderStyle

SetPressed

Funktion

Stellt für Buttons oder Rundbuttons ein, ob die Schalterstellung "gedrückt" oder "nicht gedrückt" ist.

Syntax

BOOL SetPressed(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bPressed);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bPressed

Schalterstellung des Buttons

TRUE	Schalterstellung "gedrückt"
FALSE	Schalterstellung "nicht gedrückt"

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetToggle

Funktion

Stellt bei Buttons oder Rundbuttons ein, ob der Schalter rastbar ist oder nicht.

Syntax

BOOL SetToggle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bToggle);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bToggle

Schalter rastbar/nicht rastbar

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

SetWindowsStyle

Funktion

Stellt bei Buttons ein, ob sie im Windows-Stil dargestellt werden.

Syntax

BOOL SetWindowsStyle(LPCTSTR lpszPictureName, LPCTSTR lpszObjectName, BOOL bWindowStyle);

Parameter

lpszPictureName

Name des Bildes

lpszObjectName

Name des Objekts

bWindowStyle

"Windows-Stil" ein/aus

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

OpenHomePicture

Funktion

Öffnet das eingetragene Startbild.

Syntax

```
BOOL OpenHomePicture();
```

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

OpenNextPicture

Funktion

WinCC speichert die Namen der Bilder, die zur Laufzeit durch den Anwender geöffnet wurden, sowie die Reihenfolge, in der diese Bilder geöffnet wurden.

Die maximale Anzahl der Bildnamen, die so gespeichert werden, ist im WinCC Explorer in den Rechnereigenschaften auf der Registerkarte "Graphics Runtime" unter "Bildpuffergröße" einstellbar.

Die Funktion OpenNextPicture öffnet nun das Bild, das vor dem letzen Aufruf von OpenPrevPicture geöffnet wurde.

Syntax

```
BOOL OpenNextPicture();
```

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

OpenPrevPicture

Funktion

WinCC speichert die Namen der Bilder, die zur Laufzeit durch den Anwender geöffnet wurden, sowie die Reihenfolge, in der diese Bilder geöffnet wurden.

Die maximale Anzahl der Bildnamen, die so gespeichert werden, ist im WinCC Explorer in den Rechnereigenschaften Registerkarte "Graphics Runtime" unter "Bildpuffergröße" einstellbar.

Die Funktion OpenPrevPicture öffnet nun das Bild, das vor dem aktuell geöffneten Bild geöffnet wurde.

Syntax

```
BOOL OpenPrevPicture();
```

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

OpenStoredPicture

Funktion

Öffnet das Bild, das mit der Funktion StorePicture gespeichert worden ist.

Syntax

```
BOOL OpenStoredPicture();
```

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

StorePicture

Funktion

Speichert das aktuelle Bild; dieses gespeicherte Bild kann mit der Funktion OpenStoredPicture geöffnet werden.

Syntax

```
BOOL StorePicture();
```

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

2.15.3.5 tag

tag - Kurzbeschreibung

Mit den Funktionen der Gruppe Tag können Sie Variablen setzen bzw. abfragen.

GetTag oder GetTagWait?

Mit GetTag angeforderte Prozessvariablen werden in ein Abbild aufgenommen. Da die Aktualisierung und das Auslesen des Abbildes zwei voneinander getrennt laufende Vorgänge sind, wird der GetTag-Aufruf nicht direkt von der Kopplung beeinflusst. Er kann somit schneller und unabhängiger ausgeführt werden als ein GetTagWait-Aufruf.

Mit GetTagWait angeforderte Prozessvariablen werden nicht in das Abbild aufgenommen. Ein GetTagWait-Aufruf liest den Wert explizit aus der AS aus. Dies beinhaltet immer Hin- und Rückweg über die Kopplung und die Reaktionszeit des AS. Während dieser Laufzeit ist die Abarbeitung der C-Aktionen blockiert und es ist nicht vorhersehbar wie lange dieser Aufruf dauert. Werden mehrere Variablen gelesen, so summiert sich diese Zeit.

Ein GetTagWait-Aufruf ist erforderlich, wenn

- schnelle Schreib / Lese-Vorgänge synchronisiert werden sollen
- ein Wert explizit aus dem AS gelesen
- oder eine Anmeldung im Abbild bewusst umgangen werden soll.

Zu vermeiden ist der GetTagWait-Aufruf in zyklischen C-Aktionen, dies ist der Hauptgrund für Performance-Probleme.

SetTag oder SetTagWait?

Der SetTag-Aufruf erteilt einen Schreibauftrag ohne eine Bestätigung des AS abzuwarten.

Der SetTagWait-Aufruf erteilt einen Schreibauftrag und wartet eine Bestätigung des AS ab. Dies beinhaltet immer Hin- und Rückweg über die Kopplung und die Reaktionszeit des AS. Während dieser Laufzeit ist die Abarbeitung der C-Aktionen blockiert und es ist nicht vorhersehbar wie lange dieser Aufruf dauert. Werden mehrere Variablen geschrieben, so summiert sich diese Zeit.

Ein SetTagWait-Aufruf wird eingesetzt um zu gewährleisten, dass der Wert geschrieben wurde bevor die C-Aktion weiter abgearbeitet wird. Zu vermeiden ist der SetTagWait-Aufruf in zyklischen C-Aktionen.

Hinweis

Der Unterschied zwischen GetTag und GetTagWait besteht auch bei internen Variablen. Allerdings ist der Unterschied hier nicht so gravierend, da keine Kopplung im Spiel ist. Um schnelle Schreib / Lese-Vorgänge zu synchronisieren, ist es auch bei internen Variablen erforderlich die entsprechende Wait-Funktion zu verwenden.

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

get

Funktionsweise der GetTag Funktionen

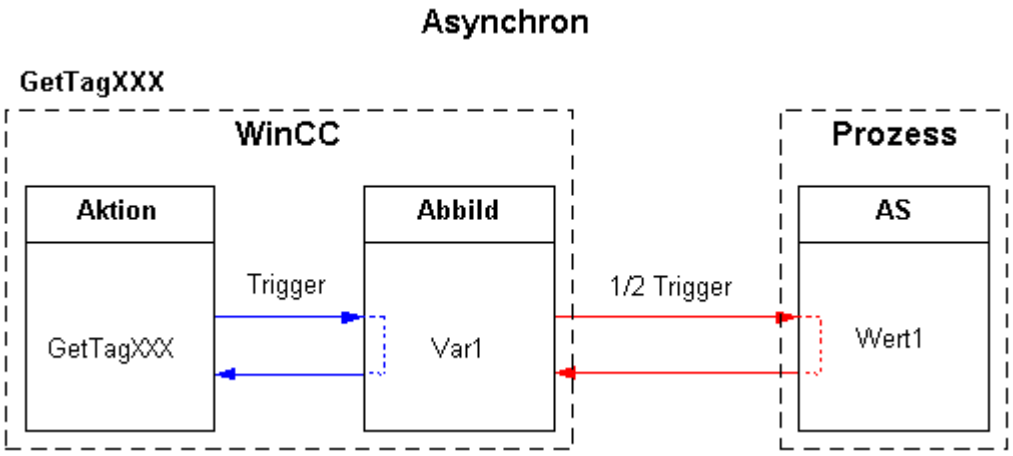
GetTagXXX

Durch den Aufruf der Funktion wird die Variable angemeldet; sie wird von nun an zyklisch aus dem AS angefordert. Der Zyklus der Anmeldung ist hierbei vom Trigger abhängig (siehe nachfolgende Beschreibung). Bei GetTagXXX Aufrufen wird der in WinCC vorliegende Wert geliefert. Bei Bildabwahl werden die Variablen wieder abgemeldet.

Der Aufruf zeichnet sich somit durch Folgendes aus:

- Der Wert wird aus dem Variablenabbild von WinCC gelesen.
- Der Aufruf ist im Vergleich zu GetTagXXXWait schneller (außer beim ersten Aufruf; dieser dauert prinzipiell länger, da der Wert aus dem AS gelesen und angemeldet werden muß).

- Die Dauer des Aufrufs ist nicht von der Buslast oder vom AS abhängig.
- Die Funktion liefert keine Informationen über den Status der Variablen



Hinweis

Wenn eine Variable in einer Global Script Aktion angefordert wird, bleibt sie über die gesamte Runtime-Laufzeit von WinCC angemeldet.

In Callback-Funktionen muss die entsprechende GetTagXXXWait Funktion verwendet werden.

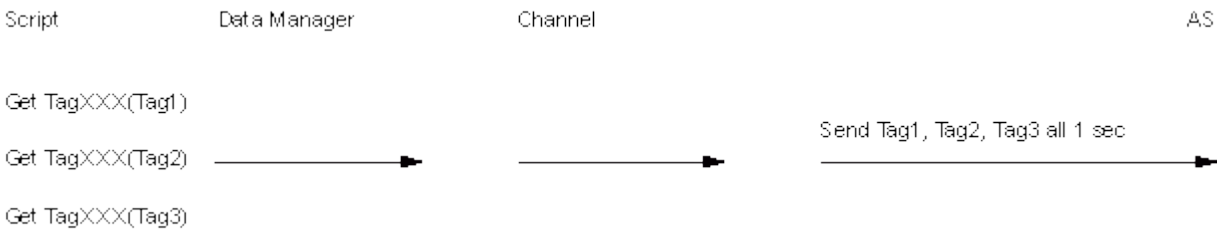
Verhalten in Aktionen mit Variablentrigger (empfohlen):

Alle im Variablentrigger enthaltenen Variablen sind bereits bei Bildanwahl bekannt und werden mit der angegebenen Überwachungszeit angemeldet.

Da alle Variablen auf einmal angefordert werden, kann vom Kanal die bestmögliche Optimierung erzielt werden. Wird innerhalb einer C-Aktion eine Variable mit GetTagXXX() angefordert, die im Trigger enthalten ist, so liegt der Wert bereits vor und wird dem Aufruf direkt übergeben (performant).

Anmelden von Variablen in Aktionen mit Variablentrigger

Da die Variablen bereits bei Bildanwahl bekannt sind, können sie in einem Auftrag an den Datenmanager übergeben und so gesammelt beim Kanal angemeldet werden.



Hinweis

Wenn eine Variable angefordert wird, die nicht im Trigger enthalten ist, so ist das Verhalten wie beim Standardtrigger.

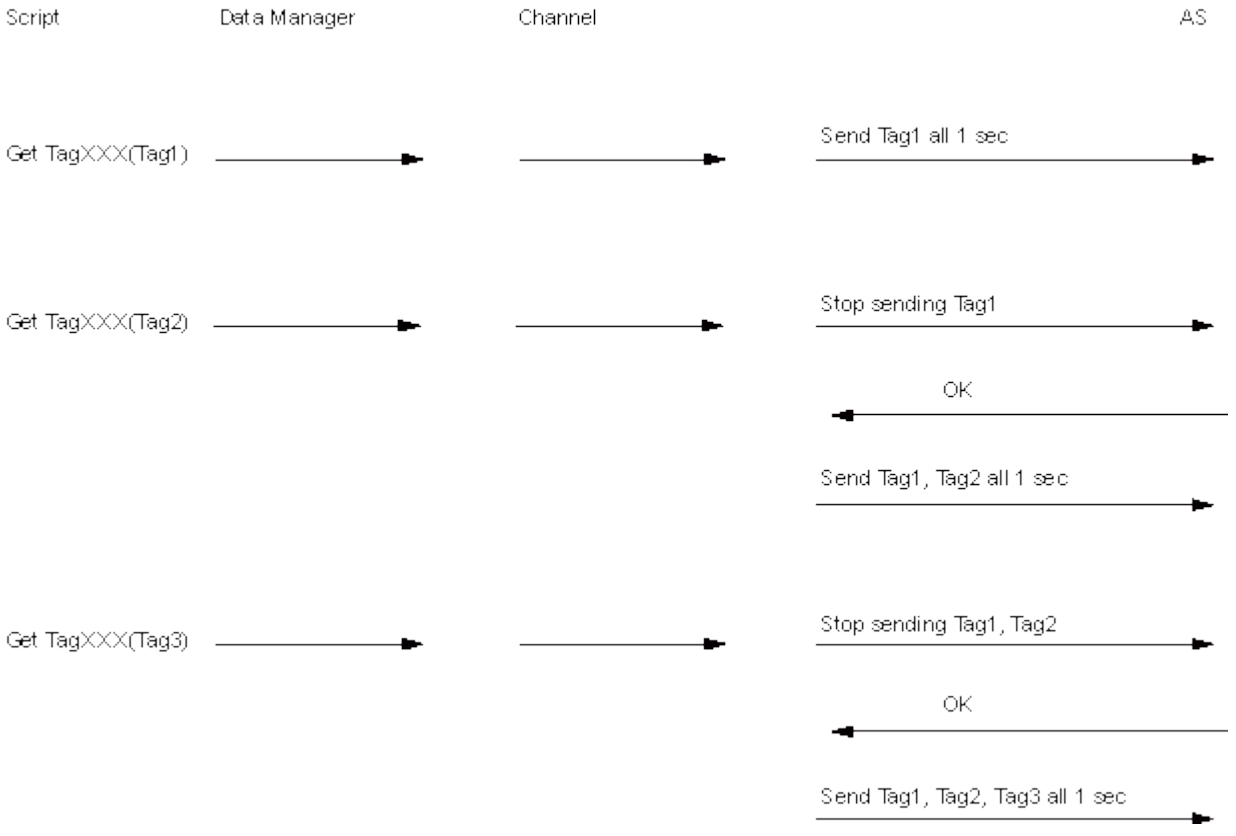
Verhalten in Aktionen mit Standardtrigger:

Beim ersten Aufruf wird die Variable mit der halben Zykluszeit angemeldet. Bei jedem weiteren Aufruf liegt dann der Wert vor.

Anmelden von Variablen in Aktionen mit Standardtrigger und Ereignistrigger

Erst beim Ausführen der einzelnen Aktionen, wird erkannt welche Variablen in dem Bild benötigt werden. Dadurch werden die Variablen in vielen einzelnen Aufträgen beim Kanal angemeldet. Bei der Anwahl eines Bildes mit zyklischen Aktionen kann die Kommunikation durch die ständige Umorganisation stark belastet werden.

Beispiel: Der Kanal unterstützt eigene Zyklusbildung. In der Regel wird die Zyklusbildung durch den Kanal direkt vom AS vorgenommen. Die Ressourcen für diese Zyklen sind durch das AS begrenzt. Dadurch stoppt der Kanal die aktuellen Aufträge für diesen Zyklus und richtet den Zyklus erneut beim AS ein.



Verhalten in ereignisgetriggerten Aktionen:

Beim ersten Aufruf wird die Variable im Modus "bei Änderung" angemeldet. Prozessvariablen, die im Modus "bei Änderung" angemeldet sind, entsprechen einem zyklischen Leseauftrag mit einer Zykluszeit von 1s.

Hinweis

Wenn mit einem Ereignis z.B. "Mausklick" ein Wert durch GetTagXXX() angefordert wird, wird die Variable in das Variablenabbild aufgenommen. Die Variable wird ab diesem Zeitpunkt zyklisch aus dem AS angefordert und erhöht somit die Grundlast.

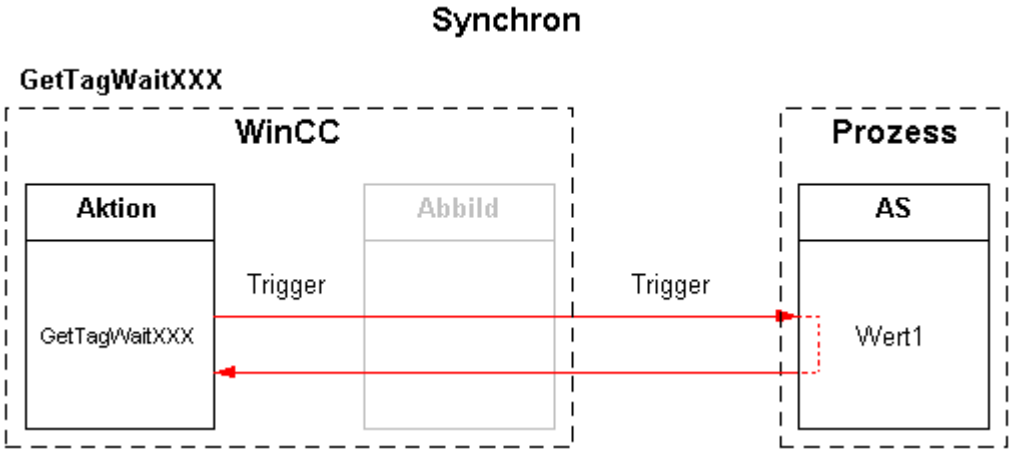
Um diese Erhöhung der Grundlast zu umgehen, kann der Wert durch GetTagXXXWait() angefordert werden. Der Aufruf GetTagXXXWait() verursacht zwar einmalig eine höhere Kommunikationsbelastung, aber die Variable wird nicht in das Variablenabbild aufgenommen.

GetTagXXXWait

Die Funktion liefert den aktuellen Wert zurück. Die Variable wird nicht zyklisch angemeldet, sondern der Wert wird einmalig aus dem AS angefordert.

Der Aufruf zeichnet sich somit durch Folgendes aus:

- Der Wert wird explizit aus dem AS gelesen.
- Der Aufruf dauert im Vergleich zu GetTagXXX länger.
- Die Dauer des Aufrufs ist unter anderem von der Buslast und vom AS abhängig.
- Die Funktion liefert keine Informationen über den Status der Variable.



GetTagXXXState

Die Funktion GetTagXXXState besitzt die gleichen Merkmale wie GetTagXXX, zusätzlich liefert die Funktion Informationen über den Status der Variablen. Da der Status intern immer mitgeliefert wird, besteht kein Performanceunterschied zu GetTagXXX.

GetTagXXXStateWait

Die Funktion GetTagXXXStateWait besitzt die gleichen Merkmale wie GetTagXXXWait, zusätzlich liefert die Funktion Informationen über den Status der Variablen. Da der Status intern immer mitgeliefert wird, besteht kein Performanceunterschied zu GetTagXXXWait.

Der Unterschied zwischen den Funktionen GetTagXXXStateWait und GetTagXXXState entspricht dem Unterschied zwischen GetTagXXXWait und GetTagXXX. Da bei Prozessvariablen der Wert explizit aus dem AS gelesen wird, kann sowohl der Wert, als auch der Status aktueller als bei GetTagXXXState sein.

GetTagXXXStateQC

Die Funktion GetTagXXXStateQC besitzt die gleichen Merkmale wie GetTagXXXState, zusätzlich liefert die Funktion Informationen über den Quality Code der Variablen.

GetTagXXXStateQCWait

Die Funktion GetTagXXXStateQCWait besitzt die gleichen Merkmale wie GetTagXXXStateWait, zusätzlich liefert die Funktion Informationen über den Quality Code der Variablen.

GetTagMultiWait

Die Funktion GetTagMultiWait besitzt die gleichen Merkmale wie GetTagXXXWait. Sie bietet jedoch die Möglichkeit, mehrere Variablen in einem Auftrag anfordern zu können. Dadurch können die Leseaufträge in Richtung AS in den meisten Fällen optimiert werden, so dass nur ein Auftrag an das AS abgesetzt wird.

GetTagMultiStateWait

Die Funktion GetTagMultiStateWait besitzt die gleichen Merkmale wie GetTagMultiWait; zusätzlich liefert die Funktion Informationen über die Stati der Variablen.

GetTagMultiStateQCWait

Die Funktion GetTagMultiStateQCWait besitzt die gleichen Merkmale wie GetTagMultiStateWait; zusätzlich liefert die Funktion Informationen über die Quality Codes der Variablen.

state

wait

getTagBitStateWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Binäre Variable". Der Wert wird explizit aus dem AS gelesen. Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL GetTagBitStateWait(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "BOOL"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagBitStateWait (Seite 1547)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Beispiel GetTagBitStateWait

Funktionsweise der GetTag Funktionen

GetTagByteStateWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 8-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BYTE GetTagByteStateWait(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "BYTE"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagWordStateWait (Seite 1560)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagWordStateWait

GetTagCharStateWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Textvariable 8-Bit" oder "Textvariable 16-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

char* GetTagCharStateWait(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Zeiger auf den Wert der Variablen im Datentyp "char".

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Siehe auch

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati (Seite 1601)

Beispiel GetTagCharStateWait (Seite 1550)

Beispiel GetTagCharStateWait

Funktionsweise der GetTag Funktionen

Variablenstati

GetTagDoubleStateWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Gleitkommazahl 64-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

double GetTagDoubleStateWait(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "double"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagFloatStateWait (Seite 1551)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagFloatStateWait

Funktionsweise der GetTag Funktionen

Variablenstati

GetTagDWordStateWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 32-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

DWORD GetTagDWordStateWait(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "DWORD"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagWordStateWait (Seite 1560)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagWordStateWait

Funktionsweise der GetTag Funktionen

Variablenstati

GetTagFloatStateWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Gleitkommazahl 32-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

float GetTagFloatStateWait(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "float"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagFloatStateWait (Seite 1551)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagFloatStateWait

Variablenstati

Funktionsweise der GetTag Funktionen

GetTagMultiStateWait

Funktion

Die Werte und Stati mehrerer Variablen werden ermittelt und an den entsprechenden Adressen im angegebenen Format abgelegt. Die Werte werden explizit aus dem AS gelesen.

Der Funktion muss ein DWORD Array übergeben werden, in dessen Member sich nach dem Aufruf der Funktion die Stati der einzelnen Variablen befinden. Das Array muss so groß gewählt werden, dass für diese Stati genügend Speicherplatz zur Verfügung steht.

Syntax

BOOL GetTagMultiStateWait(DWORD* pdwState, const char* pFormat)

Parameter

pdwState

Feld, in dem die Variablenstati gespeichert werden

pFormat

Formatbeschreibung für alle angeforderten Variablen und für jede Variable Name und Adresse des Wertes

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Formatbeschreiber (Seite 1596)

Variablenstati (Seite 1601)

Beispiel GetTagMultiStateWait (Seite 1553)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Formatbeschreiber

Funktionsweise der GetTag Funktionen

Beispiel GetTagMultiStateWait

GetTagRawStateWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Rohdatentyp". Der Wert wird explizit aus dem AS gelesen. Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL GetTagRawStateWait(Tag Tag_Name, BYTE pValue, DWORD size, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

pValue

Zeiger auf ein Byte-Feld, das den Wert der Rohdatenvariablen enthält

size

Größe des Byte-Feldes in Byte

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagRawStateWait (Seite 1556)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagRawStateWait

Funktionsweise der GetTag Funktionen

Variablenstati

GetTagSByteStateWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 8-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

signed char GetTagSByteStateWait(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "signed char"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagSByteStateWait (Seite 1558)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagSByteStateWait

Funktionsweise der GetTag Funktionen

Variablenstati

GetTagSDWordStateWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 32-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

long GetTagSDWordStateWait(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "long"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagSByteStateWait (Seite 1558)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagSByteStateWait

Funktionsweise der GetTag Funktionen

Variablenstati

GetTagSWordStateWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 16-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

```
short GetTagSWordStateWait(Tag Tag_Name, PDWORD lp_dwstate);
```

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "short"

Siehe auch

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati (Seite 1601)

Beispiel GetTagSByteStateWait (Seite 1558)

Beispiel GetTagSByteStateWait

Funktionsweise der GetTag Funktionen

Variablenstati

GetTagWordStateWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 16-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

WORD GetTagWordStateWait(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "WORD"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagWordStateWait (Seite 1560)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagWordStateWait

Funktionsweise der GetTag Funktionen

Variablenstati

GetTagBitState

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Binäre Variable". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL GetTagBitState(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "BOOL"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagBitStateWait (Seite 1547)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Beispiel GetTagBitStateWait

Funktionsweise der GetTag Funktionen

GetTagByteState

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 8-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BYTE GetTagByteState(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "BYTE"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagWordStateWait (Seite 1560)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagWordStateWait

GetTagCharState

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Textvariable 8-Bit" oder "Textvariable 16-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

char* GetTagCharState(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Zeiger auf den Wert der Variablen im Datentyp "char".

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagCharStateWait (Seite 1550)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagCharStateWait

GetTagDoubleState

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Gleitkommazahl 64-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

double GetTagDoubleState(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "double"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagFloatStateWait (Seite 1551)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagFloatStateWait

GetTagDWordState

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 32-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

DWORD GetTagDWordState(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "DWORD"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagWordStateWait (Seite 1560)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagWordStateWait

GetTagFloatState

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Gleitkommazahl 32-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

float GetTagFloatState(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "float"

Siehe auch

Beispiel GetTagFloatStateWait (Seite 1551)

Variablenstati (Seite 1601)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagFloatStateWait

Variablenstati

Funktionsweise der GetTag Funktionen

GetTagRawState

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Rohdatentyp". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL GetTagRawState(Tag Tag_Name, BYTE* pValue, DWORD size, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

pValue

Zeiger auf ein Byte-Feld, das den Wert der Rohdatenvariablen enthält

size

Größe des Byte-Feldes in Byte

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagRawStateWait (Seite 1556)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagRawStateWait

GetTagSByteState

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 8-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

signed char GetTagSByteState(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "signed char"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagSByteStateWait (Seite 1558)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagSByteStateWait

GetTagSDWordState

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 32-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

long GetTagSDWordState(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "long"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagSByteStateWait (Seite 1558)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagSByteStateWait

GetTagSWordState

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 16-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

```
short GetTagSWordState(Tag Tag_Name, PDWORD lp_dwstate);
```

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "short"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagSByteStateWait (Seite 1558)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagSByteStateWait

GetTagWordState

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 16-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

WORD GetTagWordState(Tag Tag_Name, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "WORD"

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagWordStateWait (Seite 1560)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagWordStateWait

stateqc

wait

GetTagBitStateQCWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Binäre Variable". DerWert wird explizit aus dem AS gelesen. Zusätzlich werden der Status und der Quality Code der Variablen zurückgegeben.

Syntax

BOOL GetTagBitStateQCWait(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "BOOL".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagWordStateQCWait (Seite 1559)

Funktionsweise der GetTag Funktionen (Seite 1403)

Funktionsweise der GetTag Funktionen

Variablenstati

Beispiel GetTagWordStateQCWait

GetTagByteStateQCWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 8-Bit". DerWert wird explizit aus dem AS gelesen. Zusätzlich werden der Status und der Quality Code der Variablen zurückgegeben.

Syntax

BYTE GetTagByteStateQCWait(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "BYTE".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagWordStateQCWait (Seite 1559)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagWordStateQCWait

Variablenstati

Funktionsweise der GetTag Funktionen

GetTagCharStateQCWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Textvariable 8-Bit" oder"Textvariable 16-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich werden der Status und der Quality Code der Variablen zurückgegeben.

Syntax

char* GetTagCharStateQCWait(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Zeiger auf den Wert der Variablen im Datentyp "char".

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagCharStateQCWait (Seite 1549)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagCharStateQCWait

GetTagDoubleStateQCWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Gleitkommazahl 64-Bit". DerWert wird explizit aus dem AS gelesen. Zusätzlich werden der Status und der Quality Code der Variablen zurückgegeben.

Syntax

double GetTagDoubleStateQCWait(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "double".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagFloatStateQCWait (Seite 1551)

Funktionsweise der GetTag Funktionen (Seite 1403)

Funktionsweise der GetTag Funktionen

Variablenstati

Beispiel GetTagFloatStateQCWait

GetTagDWordStateQCWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 32-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich werden der Status und der Quality Code der Variablen zurückgegeben.

Syntax

DWORD GetTagDWordStateQCWait(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "DWORD".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagWordStateQCWait (Seite 1559)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagWordStateQCWait

GetTagFloatStateQCWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Gleitkommazahl 32-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich werden der Status und der Quality Code der Variablen zurückgegeben.

Syntax

float GetTagFloatStateQCWait(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "float".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagFloatStateQCWait (Seite 1551)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagFloatStateQCWait

GetTagMultiStateQCWait

Funktion

Die Werte, Stati und Quality Codes mehrerer Variablen werden ermittelt und an den entsprechenden Adressen im angegebenen Format abgelegt. Die Werte werden explizit aus dem AS gelesen.

Der Funktion müssen zwei DWORD Arrays übergeben werden, in dessen Member sich nach dem Aufruf der Funktion die Stati und Quality Codes der einzelnen Variablen befinden. Die Arrays müssen so groß gewählt werden, dass für diese Stati und Quality Codes genügend Speicherplatz zur Verfügung steht.

Syntax

BOOL GetTagMultiStateQCWait(DWORD* pdwState, DWORD* pdwQualityCode, const char* pFormat)

Parameter

pdwState

Feld, in dem nach dem Durchlauf der Funktion der Status der einzenen Variablen abgelegt wird.

pdwQualityCode

Feld, in dem nach dem Durchlauf der Funktion der Quality Code der einzelnen Variablen abgelegt wird.

pFormat

Formatbeschreibung für alle angeforderten Variablen und für jede Variable Name und Adresse des Wertes.

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Formatbeschreiber (Seite 1596)

Variablenstati (Seite 1601)

Beispiel GetTagMultiStateQCWait (Seite 1552)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagMultiStateQCWait

GetTagRawStateQCWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Rohdatentyp". Der Wert wird explizit aus dem AS gelesen. Zusätzlich werden der Status und der Quality Code der Variablen zurückgegeben.

Syntax

BOOL GetTagRawStateQCWait(Tag Tag_Name, BYTE pValue, DWORD size, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

pValue

Zeiger auf ein Byte-Feld, das den Wert der Rohdatenvariablen enthält.

size

Größe des Byte-Feldes in Byte.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagRawStateQCWait (Seite 1556)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagRawStateQCWait

GetTagSByteStateQCWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 8-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich werden der Status und der Quality Code der Variablen zurückgegeben.

Syntax

signed char GetTagSByteStateQCWait(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "signed char".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagSByteStateQCWait (Seite 1557)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagSByteStateQCWait

GetTagSDWordStateQCWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 32-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich werden der Status und der Quality Code der Variablen zurückgegeben.

Syntax

long GetTagSDWordStateQCWait(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "long".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagSByteStateQCWait (Seite 1557)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagSByteStateQCWait

GetTagSWordStateQCWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 16-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich werden der Status und der Quality Code der Variablen zurückgegeben.

Syntax

short GetTagSWordStateQCWait(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "short".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagSByteStateQCWait (Seite 1557)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagSByteStateQCWait

GetTagValueStateQCWait

Funktion

Ermöglicht die Übergabe eines Wertes in Form eines Variants. Ermittelt den Zeiger auf die Ergebnisstruktur, die den Wert enthält. Der Wert wird explizit aus dem AS gelesen. Zusätzlich werden der Status und der Quality Code der Variablen zurückgegeben.

Syntax

BOOL GetTagValueStateQCWait(LPDM_VARKEY lpdmVarKey,
LPDM_VAR_UPDATE_STRUCTEX lpdmresult, LPCMN_ERROR lpdmError);

Parameter

lpdmVarKey

Zeiger auf eine Struktur vom Datentyp "DM_VARKEY"

lpdmresult

Zeiger auf den Wert vom Datentyp "DM_VAR_UPDATE_STRUCTEX"

lpdmError

Zeiger auf die Struktur, welche die Fehlerbeschreibung enthält

Rückgabewert

TRUE

Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Strukturdefinition CMN_ERROR (Seite 1604)

Strukturdefinition DM_VAR_UPDATE_STRUCTEX (Seite 1607)

Strukturdefinition DM_VARKEY (Seite 1608)

Funktionsweise der GetTag Funktionen (Seite 1403)

Funktionsweise der GetTag Funktionen

Strukturdefinition CMN_ERROR

Strukturdefinition DM_VAR_UPDATE_STRUCTEX

Strukturdefinition DM_VARKEY

GetTagWordStateQCWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 16-Bit". Der Wert wird explizit aus dem AS gelesen. Zusätzlich werden der Status und der Quality Code der Variablen zurückgegeben.

Syntax

WORD GetTagWordStateQCWait(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "WORD".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagWordStateQCWait (Seite 1559)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagWordStateQCWait

GetTagBitStateQC

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Binäre Variable". Zusätzlich werden derStatus und der Quality Code der Variablen zurückgegeben.

Syntax

BOOL GetTagBitStateQC(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "BOOL".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagBitStateQC (Seite 1547)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagBitStateQC

GetTagByteStateQC

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 8-Bit". Zusätzlich werden derStatus und der Quality Code der Variablen zurückgegeben.

Syntax

BYTE GetTagByteStateQC(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "BYTE".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagWordStateQCWait (Seite 1559)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagWordStateQCWait

GetTagCharStateQC

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Textvariable 8-Bit" oder"Textvariable 16-Bit". Zusätzlich werden der Status und der Quality Code der Variablen zurückgegeben.

Syntax

char* GetTagCharStateQC(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Zeiger auf den Wert der Variablen im Datentyp "char".

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagCharStateQCWait (Seite 1549)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagCharStateQCWait

GetTagDoubleStateQC

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Gleitkommazahl 64-Bit". Zusätzlich werden derStatus und der Quality Code der Variablen zurückgegeben.

Syntax

double GetTagDoubleStateQC(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "double".

Siehe auch

Beispiel GetTagFloatStateQCWait (Seite 1551)

Variablenstati (Seite 1601)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagFloatStateQCWait

Variablenstati

Funktionsweise der GetTag Funktionen

GetTagDWordStateQC

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 32-Bit". Zusätzlich werden derStatus und der Quality Code der Variablen zurückgegeben.

Syntax

DWORD GetTagDWordStateQC(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "DWORD".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagWordStateQCWait (Seite 1559)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagWordStateQCWait

Funktionsweise der GetTag Funktionen

Variablenstati

GetTagFloatStateQC

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Gleitkommazahl 32-Bit". Zusätzlich werden derStatus und der Quality Code der Variablen zurückgegeben.

Syntax

```
float GetTagFloatStateQC(Tag Tag_Name, PDWORD lp_dwstate, PDWORD
pdwQualityCode);
```

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "float".

Siehe auch

Beispiel GetTagFloatStateQCWait (Seite 1551)

Variablenstati (Seite 1601)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagFloatStateQCWait

Variablenstati

Funktionsweise der GetTag Funktionen

GetTagRawStateQC

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Rohdatentyp". Zusätzlich werden derStatus und der Quality Code der Variablen zurückgegeben.

Syntax

BOOL GetTagRawStateQC(Tag Tag_Name, BYTE* pValue, DWORD size, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

pValue

Zeiger auf ein Byte-Feld, das den Wert der Rohdatenvariablen enthält.

size

Größe des Byte-Feldes in Byte.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagRawStateQCWait (Seite 1556)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagRawStateQCWait

Funktionsweise der GetTag Funktionen

Variablenstati

GetTagSByteStateQC

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 8-Bit". Zusätzlich werden derStatus und der Quality Code der Variablen zurückgegeben.

Syntax

signed char GetTagSByteStateQC(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "signed char".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagSByteStateQCWait (Seite 1557)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagSByteStateQCWait

Variablenstati

Funktionsweise der GetTag Funktionen

GetTagSDWordStateQC

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 32-Bit". Zusätzlich werden derStatus und der Quality Code der Variablen zurückgegeben.

Syntax

long GetTagSDWordStateQC(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "long".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagSByteStateQCWait (Seite 1557)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagSByteStateQCWait

Funktionsweise der GetTag Funktionen

Variablenstati

GetTagSWordStateQC

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 16-Bit". Zusätzlich werden derStatus und der Quality Code der Variablen zurückgegeben.

Syntax

short GetTagSWordStateQC(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "short".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagSByteStateQCWait (Seite 1557)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagSByteStateQCWait

Funktionsweise der GetTag Funktionen

Variablenstati

GetTagValueStateQC

Funktion

Ermöglicht die Übergabe eines Wertes in Form eines Variants. Ermittelt den Zeiger auf die Ergebnisstruktur, die den Wert enthält. Zusätzlich werden der Status und der Quality Code der Variablen zurückgegeben.

Syntax

BOOL GetTagValueStateQC(LPDM_VARKEY lpdmVarKey,
LPDM_VAR_UPDATE_STRUCTEX lpdmresult, LPCMN_ERROR lpdmError);

Parameter

lpdmVarKey

Zeiger auf eine Struktur vom Datentyp "DM_VARKEY"

lpdmresult

Zeiger auf den Wert vom Datentyp "DM_VAR_UPDATE_STRUCTEX"

lpdmError

Zeiger auf die Struktur, welche die Fehlerbeschreibung enthält

Rückgabewert

TRUE

Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Strukturdefinition CMN_ERROR (Seite 1604)

Strukturdefinition DM_VAR_UPDATE_STRUCTEX (Seite 1607)

Strukturdefinition DM_VARKEY (Seite 1608)

Funktionsweise der GetTag Funktionen (Seite 1403)

Funktionsweise der GetTag Funktionen

Strukturdefinition DM_VARKEY

Strukturdefinition DM_VAR_UPDATE_STRUCTEX

Strukturdefinition CMN_ERROR

GetTagWordStateQC

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 16-Bit". Zusätzlich werden derStatus und der Quality Code der Variablen zurückgegeben.

Syntax

WORD GetTagWordStateQC(Tag Tag_Name, PDWORD lp_dwstate, PDWORD pdwQualityCode);

Parameter

Tag_Name

Name der Variablen.

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

pdwQualityCode

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Quality Code der Variablen abgelegt wird.

Rückgabewert

Wert der Variablen im Datentyp "WORD".

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagWordStateQCWait (Seite 1559)

Funktionsweise der GetTag Funktionen (Seite 1403)

Variablenstati

Funktionsweise der GetTag Funktionen

Beispiel GetTagWordStateQCWait

wait

GetTagBitWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Binäre Variable". Der Wert wird explizit aus dem AS gelesen.

Syntax

```
BOOL GetTagBitWait(Tag Tag_Name);
```

Parameter

Tag_Name

Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "BOOL"

Siehe auch

Beispiel GetTagBit (Seite 1546)

Funktionsweise der GetTag Funktionen (Seite 1403)

Funktionsweise der GetTag Funktionen

Beispiel GetTagBit

GetTagByteWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 8-Bit". Der Wert wird explizit aus dem AS gelesen.

Syntax

BYTE GetTagByteWait(Tag Tag_Name);

Parameter

Tag_Name

Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "BYTE"

Siehe auch

Beispiel GetTagWord (Seite 1559)

Funktionsweise der GetTag Funktionen (Seite 1403)

Funktionsweise der GetTag Funktionen

Beispiel GetTagWord

GetTagCharWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Textvariable 8-Bit" oder "Textvariable 16-Bit". Der Wert wird explizit aus dem AS gelesen.

Syntax

char* GetTagCharWait(Tag Tag_Name);

Parameter

Tag_Name

Name der Variablen

Rückgabewert

Zeiger auf eine Zeichenkette, die den Wert der Variablen enthält.

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Siehe auch

Beispiel GetTagChar (Seite 1548)

Funktionsweise der GetTag Funktionen (Seite 1403)

Funktionsweise der GetTag Funktionen

Beispiel GetTagChar

GetTagDoubleWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Gleitkommazahl 64-Bit". Der Wert wird explizit aus dem AS gelesen.

Syntax

double GetTagDoubleWait(Tag Tag_Name);

Parameter

Tag_Name

Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "double"

Siehe auch

Beispiel GetTagFloat (Seite 1550)

Funktionsweise der GetTag Funktionen (Seite 1403)

Funktionsweise der GetTag Funktionen

Beispiel GetTagFloat

GetTagDWordWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 32-Bit". Der Wert wird explizit aus dem AS gelesen.

Syntax

DWORD GetTagDWordWait(Tag Tag_Name);

Parameter

Tag_Name

Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "DWORD"

Siehe auch

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagWord (Seite 1559)

Beispiel GetTagWord

Funktionsweise der GetTag Funktionen

GetTagFloatWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Gleitkommazahl 32-Bit". Der Wert wird explizit aus dem AS gelesen.

Syntax

float GetTagFloatWait(Tag Tag_Name);

Parameter

Tag_Name

Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "float"

Siehe auch

Beispiel GetTagFloat (Seite 1550)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagFloat

Funktionsweise der GetTag Funktionen

GetTagMultiWait

Funktion

Die Werte mehrerer Variablen werden ermittelt und bei den entsprechenden Adressen im angegebenen Format abgelegt. Der Wert wird explizit aus dem AS gelesen. Der Speicher für den Variablenwert wird durch die Funktion mit SysMalloc angelegt.

Syntax

BOOL GetTagMultiWait(const char* pFormat,...)

Parameter

pFormat

Formatbeschreibung für alle angeforderten Variablen und für jede Variable Name und Adresse des Wertes

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Formatbeschreiber (Seite 1596)

Beispiel GetTagMultiWait (Seite 1554)

Funktionsweise der GetTag Funktionen (Seite 1403)

Funktionsweise der GetTag Funktionen

Beispiel GetTagMultiWait

Formatbeschreiber

GetTagRawWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Rohdatentyp". Der Wert wird explizit aus dem AS gelesen.

Syntax

BOOL GetTagRawWait(Tag Tag_Name , BYTE pValue, DWORD size);

Parameter

Tag_Name

Name der Variablen

pValue

Zeiger auf ein Byte-Feld, das den Wert der Rohdatenvariablen enthält

size

Größe des Byte-Feldes in Byte

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagRaw (Seite 1555)

Funktionsweise der GetTag Funktionen

Beispiel GetTagRaw

GetTagSByteWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 8-Bit". Der Wert wird explizit aus dem AS gelesen.

Syntax

signed char GetTagSByteWait(Tag Tag_Name);

Parameter

Tag_Name

Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "signed char"

Siehe auch

Beispiel GetTagSByte (Seite 1557)

Funktionsweise der GetTag Funktionen (Seite 1403)

Funktionsweise der GetTag Funktionen

Beispiel GetTagSByte

GetTagSDWordWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 32-Bit". Der Wert wird explizit aus dem AS gelesen.

Syntax

long GetTagSDWordWait(Tag Tag_Name);

Parameter

Tag_Name

Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "long"

Siehe auch

Beispiel GetTagSByte (Seite 1557)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagSByte

Funktionsweise der GetTag Funktionen

GetTagSWordWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 16-Bit". Der Wert wird explizit aus dem AS gelesen.

Syntax

short GetTagSWordWait(Tag Tag_Name);

Parameter

Tag_Name

Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "short"

Siehe auch

Beispiel GetTagSByte (Seite 1557)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagSByte

Funktionsweise der GetTag Funktionen

GetTagValueWait

Funktion

Ermöglicht die Übergabe eines Wertes in Form eines Variants. Ermittelt den Zeiger auf die Ergebnisstruktur, die den Wert enthält. Der Wert wird direkt aus der AS gelesen.

Syntax

BOOL GetTagValueWait(LPDM_VARKEY lpdmVarKey, LPDM_VAR_UPDATE_STRUCT lpdmresult, LPCMN_ERROR lpdmError);

Parameter

lpdmVarKey

Zeiger auf eine Struktur vom Datentyp "DM_VARKEY"

lpdmresult

Zeiger auf den Wert vom Datentyp "DM_VAR_UPDATE_STRUCT"

lpdmError

Zeiger auf die Struktur, die die Fehlerbeschreibung enthält

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Strukturdefinition CMN_ERROR (Seite 1604)

Strukturdefinition DM_VAR_UPDATE_STRUCT (Seite 1606)

Strukturdefinition DM_VARKEY (Seite 1608)

Funktionsweise der GetTag Funktionen (Seite 1403)

Funktionsweise der GetTag Funktionen

Strukturdefinition DM_VARKEY

Strukturdefinition DM_VAR_UPDATE_STRUCTEX

Strukturdefinition CMN_ERROR

GetTagWordWait

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 16-Bit". Der Wert wird explizit aus dem AS gelesen.

Syntax

WORD GetTagWordWait(Tag Tag_Name);

Parameter

Tag_Name

Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "WORD"

Siehe auch

Beispiel GetTagWord (Seite 1559)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagWord

Funktionsweise der GetTag Funktionen

GetTagBit

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Binäre Variable".

Syntax

BOOL GetTagBit(Tag Tag_Name);

Parameter

Tag_Name
Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "BOOL"

Siehe auch

Beispiel GetTagBit (Seite 1546)
Funktionsweise der GetTag Funktionen (Seite 1403)
Beispiel GetTagBit
Funktionsweise der GetTag Funktionen

GetTagByte

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 8-Bit".

Syntax

BYTE GetTagByte(Tag Tag_Name);

Parameter

Tag_Name
Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "BYTE"

Siehe auch

Beispiel GetTagWord (Seite 1559)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagWord

Funktionsweise der GetTag Funktionen

GetTagChar

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Textvariable 8-Bit" oder "Textvariable 16-Bit".

Syntax

char* GetTagChar(Tag Tag_Name);

Parameter

Tag_Name

Name der Variablen

Rückgabewert

Zeiger auf eine Zeichenkette, die den Wert der Variablen enthält.

Der Rückgabewert muss auf seine Gültigkeit geprüft werden, um eine Nullpointer-Exception zu vermeiden, z. B. bei der Funktion "GetText()":

```
pszValue = GetText(lpszPictureName,"Text1");
if(pszValue != NULL)
{
  .......
}
```

Siehe auch

Beispiel GetTagChar (Seite 1548)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagChar

Funktionsweise der GetTag Funktionen

GetTagDouble

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Gleitkommazahl 64-Bit".

Syntax

double GetTagDouble(Tag Tag_Name);

Parameter

Tag_Name
Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "double"

Siehe auch

Beispiel GetTagFloat (Seite 1550)
Funktionsweise der GetTag Funktionen (Seite 1403)
Beispiel GetTagFloat
Funktionsweise der GetTag Funktionen

GetTagDWord

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 32-Bit".

Syntax

DWORD GetTagDWord(Tag Tag_Name);

Parameter

Tag_Name
Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "DWORD"

Siehe auch

Beispiel GetTagWord (Seite 1559)
Funktionsweise der GetTag Funktionen (Seite 1403)
Beispiel GetTagWord
Funktionsweise der GetTag Funktionen

GetTagFloat

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Gleitkommazahl 32-Bit".

Syntax

```
float GetTagFloat(Tag Tag_Name);
```

Parameter

Tag_Name
Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "float".

Siehe auch

Beispiel GetTagFloat (Seite 1550)
Funktionsweise der GetTag Funktionen (Seite 1403)
Beispiel GetTagFloat
Funktionsweise der GetTag Funktionen

GetTagRaw

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "Rohdatentyp".

Syntax

```
BOOL GetTagRaw(Tag Tag_Name, BYTE* pValue, DWORD size);
```

Parameter

Tag_Name

Name der Variablen

pValue

Zeiger auf ein Byte-Feld, das den Wert der Rohdatenvariablen enthält

size

Größe des Byte-Feldes in Byte

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel GetTagRaw (Seite 1555)

Funktionsweise der GetTag Funktionen (Seite 1403)

Funktionsweise der GetTag Funktionen

Beispiel GetTagRaw

GetTagSByte

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 8-Bit".

Syntax

signed char GetTagSByte(Tag Tag_Name);

Parameter

Tag_Name

Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "signed char"

Siehe auch

Beispiel GetTagSByte (Seite 1557)
Funktionsweise der GetTag Funktionen (Seite 1403)
Beispiel GetTagSByte
Funktionsweise der GetTag Funktionen

GetTagSDWord

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 32-Bit".

Syntax

long GetTagSDWord(Tag Tag_Name);

Parameter

Tag_Name
Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "long"

Siehe auch

Beispiel GetTagSByte (Seite 1557)
Funktionsweise der GetTag Funktionen (Seite 1403)
Beispiel GetTagSByte
Funktionsweise der GetTag Funktionen

GetTagSWord

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 16-Bit".

Syntax

short GetTagSWord(Tag Tag_Name);

Parameter

Tag_Name

Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "short"

Siehe auch

Beispiel GetTagSByte (Seite 1557)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagSByte

Funktionsweise der GetTag Funktionen

GetTagValue

Funktion

Ermöglicht die Übergabe eines Wertes in Form eines Variants. Ermittelt den Zeiger auf die Ergebnisstruktur, die den Wert enthält.

Syntax

BOOL GetTagValue(LPDM_VARKEY lpdmVarKey, LPDM_VAR_UPDATE_STRUCT lpdmresult, LPCMN_ERROR lpdmError);

Parameter

lpdmVarKey

Zeiger auf eine Struktur vom Datentyp "DM_VARKEY"

lpdmresult

Zeiger auf den Wert vom Datentyp "DM_VAR_UPDATE_STRUCT"

lpdmError

Zeiger auf die Struktur, welche die Fehlerbeschreibung enthält

Rückgabewert

TRUE

Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Strukturdefinition CMN_ERROR (Seite 1604)

Strukturdefinition DM_VAR_UPDATE_STRUCT (Seite 1606)

Strukturdefinition DM_VARKEY (Seite 1608)

Funktionsweise der GetTag Funktionen (Seite 1403)

Funktionsweise der GetTag Funktionen

Strukturdefinition DM_VARKEY

Strukturdefinition DM_VAR_UPDATE_STRUCTEX

Strukturdefinition CMN_ERROR

GetTagWord

Funktion

Ermittelt den Wert einer Variablen vom Datentyp "vorzeichenlos 16-Bit".

Syntax

```
WORD GetTagWord(Tag Tag_Name);
```

Parameter

Tag_Name

Name der Variablen

Rückgabewert

Wert der Variablen im Datentyp "WORD"

Siehe auch

Beispiel GetTagWord (Seite 1559)

Funktionsweise der GetTag Funktionen (Seite 1403)

Beispiel GetTagWord

Funktionsweise der GetTag Funktionen

set

Funktionsweise der SetTag Funktionen

SetTagXXX

Die Funktion SetTagXXX erteilt den Auftrag, einen Wert zu schreiben, und kehrt sofort zum Aufrufer zurück. Dabei wird nicht gewartet, bis der Wert tatsächlich geschrieben wurde.

Der Aufruf zeichnet sich somit durch Folgendes aus:

- Der Aufruf ist schnell.
- Der Aufrufer weiß nicht, wann der Wert tatsächlich geschrieben ist.
- Die Funktion liefert keine Informationen über den Status des Schreibauftrages.

SetTagXXXWait

Die Funktion SetTagXXXWait erteilt den Auftrag, einen Wert zu schreiben und kehrt erst zum Aufrufer zurück, wenn der Wert tatsächlich geschrieben wurde.

Der Aufruf zeichnet sich somit durch Folgendes aus:

- Der Aufruf dauert im Vergleich zu SetTagXXX länger. Die Dauer ist unter anderem vom Kanal und dem AS abhängig.
- Nach dem Aufruf ist der Wert geschrieben.
- Die Funktion liefert keine Informationen über den Status des Schreibauftrages.

SetTagXXXState

Die Funktion SetTagXXXState besitzt die gleichen Merkmale wie SetTagXXX; zusätzlich liefert die Funktion Informationen über den Status des Schreibauftrages.

Da der Status intern immer mitgeliefert wird, besteht kein Performanceunterschied zu SetTagXXX.

SetTagXXXStateWait

Die Funktion SetTagXXXStateWait besitzt die gleichen Merkmale wie SetTagXXXWait, zusätzlich liefert die Funktion Informationen über den Status des Schreibauftrages.

Da der Status intern immer mitgeliefert wird, besteht kein Performanceunterschied zu SetTagXXXWait.

Der Unterschied zwischen den Funktionen SetTagXXXStateWait und SetTagXXXState entspricht dem Unterschied zwischen SetTagXXXWait und SetTagXXX.

Zu beachten ist, dass manche Stati erst gebildet werden können, wenn der Schreibvorgang abgeschlossen ist.

SetTagMultiWait

Die Funktion SetTagMultiWait besitzt die gleichen Merkmale wie SetTagXXXWait. Sie bietet jedoch die Möglichkeit, mehrere Schreibaufträge in einem Auftrag erteilen zu können.

state

wait

SetTagBitStateWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "Binäre Variable". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat. Zusätzlich wird der Status der Variablen zurückgegeben.

Funktionsweise der SetTag FunktionenBEISPIELE_INTERNE_FUNKTIONEN_TAG_STATEWAIT_23_130

Syntax

BOOL SetTagBitStateWait(Tag Tag_Name, short value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "short"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu ist der Variablenstatus auszuwerten.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagBitStateWait (Seite 1576)

Variablenstati

Beispiel SetTagBitStateWait

SetTagByteStateWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenlos 8-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat. Zusätzlich wird der Status der Variablen zurückgegeben.

Funktionsweise der SetTag FunktionenBEISPIELE_INTERNE_FUNKTIONEN_TAG_STATEWAIT_23_130

Syntax

BOOL SetTagByteStateWait(Tag Tag_Name, BYTE value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "BYTE"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu ist der Variablenstatus auszuwerten.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagWordStateWait (Seite 1583)

Variablenstati

Beispiel SetTagWordStateWait

SetTagCharStateWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "Textvariable 8-Bit" oder "Textvariable 16-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat. Zusätzlich wird der Status der Variablen zurückgegeben.

Funktionsweise der SetTag FunktionenBEISPIELE_INTERNE_FUNKTIONEN_TAG_STATEWAIT_23_130

Syntax

BOOL SetTagCharStateWait(Tag Tag_Name, LPSTR value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "LPSTR"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu ist der Variablenstatus auszuwerten.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagCharStateWait (Seite 1577)

Variablenstati

Beispiel SetTagCharStateWait

SetTagDoubleStateWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "Gleitkommazahl 64-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat. Zusätzlich wird der Status der Variablen zurückgegeben.

Funktionsweise der SetTag FunktionenBEISPIELE_INTERNE_FUNKTIONEN_TAG_STATEWAIT_23_130

Syntax

BOOL SetTagDoubleStateWait(Tag Tag_Name, double value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "double"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu ist der Variablenstatus auszuwerten.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagFloatStateWait (Seite 1578)

Variablenstati

Beispiel SetTagFloatStateWait

SetTagDWordStateWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenlos 32-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat. Zusätzlich wird der Status der Variablen zurückgegeben.

Funktionsweise der SetTag FunktionenBEISPIELE_INTERNE_FUNKTIONEN_TAG_STATEWAIT_23_130

Syntax

BOOL SetTagDWordStateWait(Tag Tag_Name, DWORD value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "DWORD"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu ist der Variablenstatus auszuwerten.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagWordStateWait (Seite 1583)

Variablenstati

Beispiel SetTagWordStateWait

SetTagFloatStateWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "Gleitkommazahl 32-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat. Zusätzlich wird der Status der Variablen zurückgegeben.

Funktionsweise der SetTag FunktionenBEISPIELE_INTERNE_FUNKTIONEN_TAG_STATEWAIT_23_130

Syntax

BOOL SetTagFloatStateWait(Tag Tag_Name, float value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "float"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu ist der Variablenstatus auszuwerten.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagFloatStateWait (Seite 1578)

Variablenstati

Beispiel SetTagFloatStateWait

SetTagMultiStateWait

Funktion

Setzt die Werte mehrerer Variablen. Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat.

Der Funktion muss ein DWORD Array übergeben werden, in dessen Member sich nach dem Aufruf der Funktion die Stati der einzelnen Variablen befinden. Das Array muss so groß gewählt werden, dass für diese Stati genügend Speicherplatz zur Verfügung steht.

Funktionsweise der SetTag FunktionenBEISPIELE_INTERNE_FUNKTIONEN_TAG_STATEWAIT_23_130

Syntax

BOOL SetTagMultiStateWait(DWORD* pdwState, const char* pFormat,...)

Parameter

pdwState

Feld, in dem die Variablenstati gespeichert werden

pFormat

Formatbeschreibung für alle angeforderten Variablen und für jede Variable Name und Wert

FormatbeschreiberBEISPIELE_INTERNE_FUNKTIONEN_TAG_STATEWAIT_23_130

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagMultiStateWait (Seite 1579)

Variablenstati

Beispiel SetTagMultiStateWait

SetTagRawStateWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "Rohdatentyp". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat. Zusätzlich wird der Status der Variablen zurückgegeben.

Funktionsweise der SetTag FunktionenBEISPIELE_INTERNE_FUNKTIONEN_TAG_STATEWAIT_23_130

Syntax

BOOL SetTagRawStateWait(Tag Tag_Name, BYTE pValue, DWORD size, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

pValue

Zeiger auf ein Byte-Feld, das den Wert der Rohdatenvariablen enthält

size

Größe des Byte-Feldes in Byte

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu ist der Variablenstatus auszuwerten.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagRawStateWait (Seite 1581)

Variablenstati

Beispiel SetTagRawStateWait

SetTagSByteStateWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 8-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat. Zusätzlich wird der Status der Variablen zurückgegeben.

Funktionsweise der SetTag FunktionenBEISPIELE_INTERNE_FUNKTIONEN_TAG_STATEWAIT_23_130

Syntax

BOOL SetTagSByteStateWait(Tag Tag_Name, signed char value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "signed char"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu ist der Variablenstatus auszuwerten.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagSByteStateWait (Seite 1582)

Variablenstati

Beispiel SetTagSByteStateWait

SetTagSDWordStateWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 32-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat. Zusätzlich wird der Status der Variablen zurückgegeben.

Funktionsweise der SetTag FunktionenBEISPIELE_INTERNE_FUNKTIONEN_TAG_STATEWAIT_23_130

Syntax

BOOL SetTagSDWordStateWait(Tag Tag_Name, long value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "long"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu ist der Variablenstatus auszuwerten.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagSByteStateWait (Seite 1582)

Beispiel SetTagSByteStateWait

Variablenstati

SetTagSWordStateWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 16-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat. Zusätzlich wird der Status der Variablen zurückgegeben.

Funktionsweise der SetTag FunktionenBEISPIELE_INTERNE_FUNKTIONEN_TAG_STATEWAIT_23_130

Syntax

BOOL SetTagSWordStateWait(Tag Tag_Name, short value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "short"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu ist der Variablenstatus auszuwerten.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagSByteStateWait (Seite 1582)

Variablenstati

Beispiel SetTagSByteStateWait

SetTagWordStateWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenlos 16-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat. Zusätzlich wird der Status der Variablen zurückgegeben.

Funktionsweise der SetTag FunktionenBEISPIELE_INTERNE_FUNKTIONEN_TAG_STATEWAIT_23_130

Synax

BOOL SetTagWordStateWait(Tag Tag_Name, WORD value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "WORD"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu ist der Variablenstatus auszuwerten.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagWordStateWait (Seite 1583)

Beispiel SetTagWordStateWait

Variablenstati

SetTagBitState

Funktion

Setzt den Wert einer Variablen vom Datentyp "Binäre Variable". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL SetTagBitState(Tag Tag_Name, short int value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "short int"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu muss der Variablenstatus ausgewertet werden.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagBitStateWait (Seite 1576)

Funktionsweise der SetTag Funktionen (Seite 1470)

Variablenstati

Beispiel SetTagBitStateWait

Funktionsweise der SetTag Funktionen

SetTagByteState

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenlos 8-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL SetTagByteState(Tag Tag_Name, BYTE value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "BYTE"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu muss der Variablenstatus ausgewertet werden.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagWordStateWait (Seite 1583)

Funktionsweise der SetTag Funktionen (Seite 1470)

Funktionsweise der SetTag Funktionen

Variablenstati

Beispiel SetTagWordStateWait

SetTagCharState

Funktion

Setzt den Wert einer Variablen vom Datentyp "Textvariable 8-Bit" oder "Textvariable 16-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL SetTagCharState(Tag Tag_Name, LPSTR value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "LPSTR"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu muss der Variablenstatus ausgewertet werden.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagCharStateWait (Seite 1577)

Funktionsweise der SetTag Funktionen (Seite 1470)

Variablenstati

Funktionsweise der SetTag Funktionen

Beispiel SetTagCharStateWait

SetTagDoubleState

Funktion

Setzt den Wert einer Variablen vom Datentyp "Gleitkommazahl 64-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL SetTagDoubleState(Tag Tag_Name, double value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "double"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu muss der Variablenstatus ausgewertet werden.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagFloatStateWait (Seite 1578)

Funktionsweise der SetTag Funktionen (Seite 1470)

Variablenstati

Funktionsweise der SetTag Funktionen

Beispiel SetTagFloatStateWait

SetTagDWordState

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenlos 32-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL SetTagDWordState(Tag Tag_Name, DWORD value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "DWORD"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu muss der Variablenstatus ausgewertet werden.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagWordStateWait (Seite 1583)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagWordStateWait

Variablenstati

Funktionsweise der SetTag Funktionen

SetTagFloatState

Funktion

Setzt den Wert einer Variablen vom Datentyp "Gleitkommazahl 32-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL SetTagFloatState(Tag Tag_Name, float value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "float"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu muss der Variablenstatus ausgewertet werden.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagFloatStateWait (Seite 1578)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagFloatStateWait

Variablenstati

Funktionsweise der SetTag Funktionen

SetTagRawState

Funktion

Setzt den Wert einer Variablen vom Datentyp "Rohdatentyp". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL SetTagRawState(Tag Tag_Name, BYTE* pValue, DWORD size, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

pValue

Zeiger auf ein Byte-Feld, das den Wert der Rohdatenvariablen enthält

size

Größe des Byte-Feldes in Byte

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel GetTagRaw (Seite 1555)

Funktionsweise der SetTag Funktionen (Seite 1470)

Funktionsweise der SetTag Funktionen

Variablenstati

Beispiel GetTagRaw

SetTagSByteState

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 8-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL SetTagSByteState(Tag Tag_Name, signed char value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "signed char"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu muss der Variablenstatus ausgewertet werden.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagSByteStateWait (Seite 1582)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagSByteStateWait

Funktionsweise der SetTag Funktionen

Variablenstati

SetTagSDWordState

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 32-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL SetTagSDWordState(Tag Tag_Name, long value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "long"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu muss der Variablenstatus ausgewertet werden.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagSByteStateWait (Seite 1582)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagSByteStateWait

Funktionsweise der SetTag Funktionen

Variablenstati

SetTagSWordState

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 16-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL SetTagSWordState(Tag Tag_Name, short value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "short"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu muss der Variablenstatus ausgewertet werden.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagSByteStateWait (Seite 1582)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagSByteStateWait

Funktionsweise der SetTag Funktionen

Variablenstati

SetTagWordState

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenlos 16-Bit". Zusätzlich wird der Status der Variablen zurückgegeben.

Syntax

BOOL SetTagWordState(Tag Tag_Name, WORD value, PDWORD lp_dwstate);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "short"

lp_dwstate

Zeiger auf ein DWORD, in dem nach dem Durchlauf der Funktion der Status der Variablen abgelegt wird.

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte. Dazu muss der Variablenstatus ausgewertet werden.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Variablenstati (Seite 1601)

Beispiel SetTagWordStateWait (Seite 1583)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagWordStateWait

Variablenstati

Funktionsweise der SetTag Funktionen

wait

SetTagBitWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "Binäre Variable". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat.

Syntax

BOOL SetTagBitWait(Tag Tag_Name, short value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "short"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagBit (Seite 1576)

Funktionsweise der SetTag Funktionen (Seite 1470)

Funktionsweise der SetTag Funktionen

Beispiel SetTagBit

SetTagByteWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenlos 8-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat.

Syntax

BOOL SetTagByteWait(Tag Tag_Name, BYTE value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "BYTE"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagWord (Seite 1583)

Funktionsweise der SetTag Funktionen (Seite 1470)

Funktionsweise der SetTag Funktionen

Beispiel SetTagWord

SetTagCharWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "Textvariable 8-Bit" oder "Textvariable 16-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat.

Syntax

BOOL SetTagCharWait(Tag Tag_Name, LPSTR value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "LPSTR"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagChar (Seite 1577)

Funktionsweise der SetTag Funktionen

Beispiel SetTagChar

SetTagDoubleWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "Gleitkommazahl 64-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat.

Syntax

BOOL SetTagDoubleWait(Tag Tag_Name, double value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "double"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagFloat (Seite 1578)

Funktionsweise der SetTag Funktionen (Seite 1470)

Funktionsweise der SetTag Funktionen

Beispiel SetTagFloat

SetTagDWordWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenlos 32-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat.

Syntax

BOOL SetTagDWordWait(Tag Tag_Name, DWORD value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "DWORD"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagWord (Seite 1583)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagWord

Funktionsweise der SetTag Funktionen

SetTagFloatWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "Gleitkommazahl 32-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat.

Syntax

BOOL SetTagFloatWait(Tag Tag_Name, float value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "float"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagFloat (Seite 1578)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagFloat

Funktionsweise der SetTag Funktionen

SetTagMultiWait

Funktion

Die Werte mehrerer Variablen werden im angegebenen Format gesetzt. Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat.

Syntax

BOOL SetTagMultiWait(const char* pFormat,...)

Parameter

pFormat

Formatbeschreibung für alle angeforderten Variablen und für jede Variable Name und Wert

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Formatbeschreiber (Seite 1596)

Beispiel SetTagMultiWait (Seite 1580)

Funktionsweise der SetTag Funktionen (Seite 1470)

Funktionsweise der SetTag Funktionen

Formatbeschreiber

Beispiel SetTagMultiWait

SetTagRawWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "Rohdatentyp". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat.

Syntax

BOOL SetTagRawWait(Tag Tag_Name, BYTE pValue, DWORD size);

Parameter

Tag_Name

Name der Variablen

pValue

Zeiger auf ein Byte-Feld, das den Wert der Rohdatenvariablen enthält

size

Größe des Byte-Feldes in Byte

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagRaw (Seite 1580)

Funktionsweise der SetTag Funktionen (Seite 1470)

Funktionsweise der SetTag Funktionen

Beispiel SetTagRaw

SetTagSByteWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 8-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat.

Syntax

BOOL SetTagSByteWait(Tag Tag_Name, signed char value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "signed char"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagSByte (Seite 1582)

Funktionsweise der SetTag Funktionen (Seite 1470)

Funktionsweise der SetTag Funktionen

Beispiel SetTagSByte

SetTagSDWordWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 32-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat.

Syntax

BOOL SetTagSDWordWait(Tag Tag_Name, long value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "long"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagSByte (Seite 1582)

Funktionsweise der SetTag Funktionen

Beispiel SetTagSByte

SetTagSWordWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 16-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat.

Syntax

BOOL SetTagSWordWait(Tag Tag_Name, short value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "short"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagSByte (Seite 1582)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagSByte

Funktionsweise der SetTag Funktionen

SetTagValueWait

Funktion

Ermöglicht die Übergabe eines Wertes in Form eines Variants und setzt den Zeiger auf den Wert vom Datentyp "Variant". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat.

Syntax

BOOL SetTagValueWait(LPDM_VARKEY lpdmVarKey, LPVARIANT lpdmValue, PDWORD dwState, LPCMN_ERROR lpdmError);

Parameter

lpdmVarKey

Zeiger auf eine Struktur vom Datentyp "DM_VARKEY"

lpdmValue

Zeiger auf den Wert vom Datentyp "Variant". Die Beschreibung des Datentyps VARIANT finden Sie in der einschlägigen Fachliteratur.

dwState

Status der Variablen, der nach Durchlaufen der Funktion zurückgeliefert wird

lpdmError

Zeiger auf die Struktur, welche die Fehlerbeschreibung enthält

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Strukturdefinition CMN_ERROR (Seite 1604)

Variablenstati (Seite 1601)

Strukturdefinition DM_VAR_UPDATE_STRUCT (Seite 1606)

Strukturdefinition DM_VARKEY (Seite 1608)

Funktionsweise der SetTag Funktionen (Seite 1470)

Funktionsweise der SetTag Funktionen

Strukturdefinition DM_VARKEY

Strukturdefinition DM_VAR_UPDATE_STRUCTEX

Strukturdefinition CMN_ERROR

SetTagWordWait

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenlos 16-Bit". Die Funktion wird erst beendet, nachdem das AS die Übernahme des Wertes zurückgemeldet hat.

Syntax

BOOL SetTagWordWait(Tag Tag_Name, WORD value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "WORD"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagWord (Seite 1583)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagWord

Funktionsweise der SetTag Funktionen

SetTagBit

Funktion

Setzt den Wert einer Variablen vom Datentyp "Binäre Variable".

Syntax

BOOL SetTagBit(Tag Tag_Name, short int value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "short int"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagBit (Seite 1576)

Beispiel SetTagBit

Funktionsweise der SetTag Funktionen

SetTagByte

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenlos 8-Bit".

Syntax

BOOL SetTagByte(Tag Tag_Name, BYTE value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "BYTE"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagWord (Seite 1583)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagWord

Funktionsweise der SetTag Funktionen

SetTagChar

Funktion

Setzt den Wert einer Variablen vom Datentyp "Textvariable 8-Bit" oder "Textvariable 16-Bit".

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "LPSTR"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagChar (Seite 1577)

Funktionsweise der SetTag Funktionen (Seite 1470)

Funktionsweise der SetTag Funktionen

Beispiel SetTagChar

SetTagDouble

Funktion

Setzt den Wert einer Variablen vom Datentyp "Gleitkommazahl 64-Bit".

Syntax

BOOL SetTagDouble(Tag Tag_Name, double value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "double"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagFloat (Seite 1578)

Funktionsweise der SetTag Funktionen (Seite 1470)

Funktionsweise der SetTag Funktionen

Beispiel SetTagFloat

SetTagDWord

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenlos 32-Bit".

Syntax

BOOL SetTagDWord(Tag Tag_Name, DWORD value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "DWORD"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagWord (Seite 1583)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagWord

Funktionsweise der SetTag Funktionen

SetTagFloat

Funktion

Setzt den Wert einer Variablen vom Datentyp "Gleitkommazahl 32-Bit".

Syntax

BOOL SetTagFloat (Tag Tag_Name, float value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "float"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagFloat (Seite 1578)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagFloat

Funktionsweise der SetTag Funktionen

SetTagRaw

Funktion

Setzt den Wert einer Variablen vom Datentyp "Rohdatentyp".

Syntax

BOOL SetTagRaw (Tag Tag_Name, BYTE* pValue, DWORD size);

Parameter

Tag_Name

Name der Variablen

pValue

Zeiger auf ein Byte-Feld, das den Wert der Rohdatenvariablen enthält

size

Größe des Byte-Feldes in Byte

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagRaw (Seite 1580)

Funktionsweise der SetTag Funktionen (Seite 1470)

Funktionsweise der SetTag Funktionen

Beispiel SetTagRaw

SetTagSByte

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 8-Bit".

Syntax

BOOL SetTagSByte (Tag Tag_Name, signed char value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "signed char"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagSByte (Seite 1582)
Funktionsweise der SetTag Funktionen (Seite 1470)
Funktionsweise der SetTag Funktionen
Beispiel SetTagSByte

SetTagSDWord

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 32-Bit".

Syntax

BOOL SetTagSDWord (Tag Tag_Name, long value);

Parameter

Tag_Name
Name der Variablen

value
Wert der Variablen im Datentyp "long"

Rückgabewert

TRUE
Die Funktion selbst wurde fehlerfrei durchlaufen.
Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE
Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagSByte (Seite 1582)
Funktionsweise der SetTag Funktionen (Seite 1470)
Beispiel SetTagSByte
Funktionsweise der SetTag Funktionen

SetTagSWord

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenbehaftet 16-Bit".

Syntax

BOOL SetTagSWord (Tag Tag_Name, short value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "short"

size

Größe des Byte-Feldes in Byte

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagSByte (Seite 1582)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagSByte

Funktionsweise der SetTag Funktionen

SetTagValue

Funktion

Ermöglicht die Übergabe eines Wertes in Form eines Variants und setzt den Zeiger auf den Wert vom Datentyp "Variant".

Syntax

BOOL SetTagValue (LPDM_VARKEY lpdmVarKey, LPVARIANT lpdmValue, PDWORD dwState, LPCMN_ERROR lpdmError);

Parameter

lpdmVarKey

Zeiger auf eine Struktur vom Datentyp "DM_VARKEY"

lpdmValue

Zeiger auf den Wert vom Datentyp "Variant". Die Beschreibung des Datentyps VARIANT finden Sie in der einschlägigen Fachliteratur.

dwState

Status der Variablen, der nach Durchlaufen der Funktion zurückgeliefert wird

lpdmError

Zeiger auf die Struktur, welche die Fehlerbeschreibung enthält

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Funktionsweise der SetTag Funktionen (Seite 1470)

Strukturdefinition CMN_ERROR (Seite 1604)

Variablenstati (Seite 1601)

Strukturdefinition DM_VAR_UPDATE_STRUCT (Seite 1606)

Strukturdefinition DM_VARKEY (Seite 1608)

Funktionsweise der SetTag Funktionen

Strukturdefinition CMN_ERROR

Strukturdefinition DM_VAR_UPDATE_STRUCTEX

Strukturdefinition DM_VARKEY

SetTagWord

Funktion

Setzt den Wert einer Variablen vom Datentyp "vorzeichenlos 16-Bit".

Syntax

BOOL SetTagWord (Tag Tag_Name, WORD value);

Parameter

Tag_Name

Name der Variablen

value

Wert der Variablen im Datentyp "WORD"

Rückgabewert

TRUE

Die Funktion selbst wurde fehlerfrei durchlaufen.

Es wird allerdings nicht geprüft, ob auch die Variable fehlerfrei geschrieben werden konnte.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel SetTagWord (Seite 1583)

Funktionsweise der SetTag Funktionen (Seite 1470)

Beispiel SetTagWord

Funktionsweise der SetTag Funktionen

2.15.3.6 WinCC

WinCC - Kurzbeschreibung

Mit den Funktionen der Gruppe WinCC können Sie im Runtime verschiedene Einstellungen vornehmen.

Mit den Funktionen der Untergruppe System können Sie das WinCC Runtime beeinflussen.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

system

DeactivateRTProject

Funktion

Deaktiviert das aktivierte Projekt.

Hinweis

Wird Runtime auf einem Server oder Client beendet, dann ist davon jeweils nur dieser Rechner betroffen.

Ein aktiviertes Projekt, bei dem der WinCC Explorer nicht gestartet ist, muss mit der internen Funktion "ExitWinCC" geschlossen werden.

Wenn das aktivierte Projekt mit der internen Funktion "DeactivateRTProject" verlassen wurde, dann bleibt das WinCC Projekt im Hintergrund geöffnet. Um dieses Projekt zu beenden, muss der WinCC Explorer geöffnet und anschließend über das Menü "Datei" > "Beenden" geschlossen werden.

Syntax

BOOL DeactivateRTProject ();

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel DeactivateRTProject (Seite 1525)

Beispiel DeactivateRTProject

ExitWinCC

Funktion

Deaktiviert das Runtime und beendet WinCC auf dem Rechner, auf dem die Funktion ausgeführt wird.

Hinweis

Wird Runtime auf einem Server oder Client beendet, dann ist davon jeweils nur dieser Rechner betroffen.

Ein aktiviertes Projekt, bei dem der WinCC Explorer nicht gestartet ist, muss mit der internen Funktion "ExitWinCC" geschlossen werden.

Wenn das aktivierte Projekt mit der internen Funktion "DeactivateRTProject" verlassen wurde, dann bleibt das WinCC Projekt im Hintergrund geöffnet. Um dieses Projekt zu beenden, muss der WinCC Explorer geöffnet und anschließend über das Menü "Datei" > "Beenden" geschlossen werden.

Syntax

```
BOOL ExitWinCC ();
```

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Beispiel ExitWinCC (Seite 1526)

Beispiel ExitWinCC

GetLanguage

Funktion

Ermittelt die aktuelle Runtime Sprache.

Syntax

DWORD GetLanguage ();

Rückgabewert

Die aktuelle Runtimesprache mit der dazugehörigen Sprachkennung wird zurückgegeben.

Hinweis

Eine ausführliche Tabelle "Language code" finden Sie in der Dokumentation "Grundlagen von VBSkript" unter dem Indexeintrag "Language code".

Siehe auch

Beispiel GetLanguage (Seite 1534)

Beispiel GetLanguage

InquireLanguage

Funktion

Ermittelt alle Sprachen, die in der Textbibliothek für die Laufzeit projektiert sind.

Mit dwCount geben Sie an, wo die Anzahl der ermittelten Sprachkennungen abgelegt werden soll.

Syntax

DWORD* InquireLanguage (DWORD* dwCount);

Parameter

dwCount

Zeiger auf die Anzahl der ermittelten Sprachkennungen

Rückgabewert

Die projektierten Sprachen mit den dazugehörigen Sprachkennungen werden zurückgegeben.

Hinweis

Eine ausführliche Tabelle "Language code" finden Sie in der Dokumentation "Grundlagen von VBSkript" unter dem Indexeintrag "Language code".

Siehe auch

Beispiel InquireLanguage (Seite 1563)

Beispiel InquireLanguage

SetLanguage

Funktion

Ändert die Spracheinstellung im Runtime.

Syntax

BOOL SetLanguage (DWORD dwLocaleID);

Parameter

dwLocaleID

Sprachkennung der einzustellenden Sprache

Rückgabewert

TRUE

Die Funktion wurde fehlerfrei durchlaufen.

FALSE

Es ist ein Fehler aufgetreten.

Siehe auch

Sprachkennungen (Seite 1599)

Beispiel SetLanguage (Seite 1571)

Beispiel SetLanguage

Sprachkennungen

FillDiagnoseInTags

Funktion

Aktiviert oder deaktiviert das Speichern von Diagnoseinformationen in Variablen.

Da das Abfüllen der Variablen eine zusätzliche Belastung des Systems darstellt, sollte dies nur kurzzeitig zur Diagnoseinformationen eingeschaltet werden.

Syntax

void FillDiagnoseInTags (BOOL bfill);

Parameter

bFill

Speichern von Diagnoseinformationen in Variablen ein/aus

TRUE	Versorgung der Diagnosevariablen einschalten
FALSE	Versorgung der Diagnosevariablen ausschalten

Diagnosevariablen von GlobalScript

@SCRIPT_COUNT_TAGS

Diese Variable enthält die aktuelle Anzahl der über Script angeforderten Variablen.

@SCRIPT_COUNT_REQUEST_IN_QUEUES

Diese Variable enthält die aktuelle Anzahl an Aufträgen.

@SCRIPT_COUNT_ACTIONS_IN_QUEUES

Diese Variable enthält die aktuelle Anzahl an Aktionen.

GetServerTagPrefix

Funktion

Um in einem verteilten System von einem WinCC-Client auf Variablen des zugehörigen Servers zugreifen zu können, sind die Variablennamen um das Serverpräfix zu ergänzen.

Erfolgt der Zugriff auf die Variablen mit den Funktionen GetTagxx oder SetTagxx, dann sorgt die Aktionssteuerung für die notwendige Ergänzung.

Werden für den Zugriff Funktionen des WinCC API verwendet, so müssen die Variablennamen durch den Anwender ergänzt werden. Die Funktion GetServerTagPrefix liefert hierfür diese Präfixe.

Es wird jeweils ein Zeiger vom Typ "char" auf ServerPrefix, TagPrefix und WindowPrefix zurückgeliefert.

Durch den Anwender darf der Speicher weder verändert (auch kein strcat) noch freigegeben werden.

Syntax

```
void GetServerTagPrefix (char** ppszServerPrefix, char** ppszTagPrefix, char** ppszWindowPrefix);
```

Parameter

ppszServerPrefix

Zeiger auf einen Zeiger, der auf das Serverpräfix verweist

ppszTagPrefix

Zeiger auf einen Zeiger, der auf das Tagpräfix verweist

ppszWindowPrefix

Zeiger auf einen Zeiger, der auf das Windowpräfix verweist

Siehe auch

Beispiel GetServerTagPrefix (Seite 1545)

Beispiel GetServerTagPrefix

TraceText

Funktion

Der in <Parameter> angegebene Wert wird in APDiag mitprotokolliert, wenn der angegebene Diagnoselevel erfüllt ist.

Syntax

void TraceText (DWORD dwTraceLevel, char* pszFormat, <Parameter>);

Parameter

dwTraceLevel

Diagnoselevel

pszFormat

Ausgabeformat (entsprechend printf-Funktion)

<Parameter>

Zu protokollierender Wert

Hinweis

Im Parametrierungsdialog zu dieser Funktion wird die Auswahl von Variablen, Grafikobjekten und Bildern angeboten.

TraceTime

Funktion

Der in <Parameter> angegebene Wert wird in APDiag mitprotokolliert, wenn der angegebene Diagnoselevel erfüllt ist.

Zusätzlich wird die Zeit seit AP Start Diagnose in Millisekunden ausgegeben, um Performance-Messungen zu ermöglichen.

Syntax

void TraceTime (DWORD dwTraceLevel, char* pszFormat, <Parameter>);

Parameter

dwTraceLevel

Diagnoselevel

pszFormat

Ausgabeformat (entsprechend printf-Funktion)

<Parameter>

Zu protokollierender Wert

Hinweis

Im Parametrierungsdialog zu dieser Funktion wird die Auswahl von Variablen, Grafikobjekten und Bildern angeboten.

2.15.4 Beispiele

2.15.4.1 Beispiele - A bis G

Beispiel AcknowledgeMessage

```
{
//Acknowledge the AlarmLogging message which is selected
AcknowledgeMessage(GetTagWord("U08i_MsgNr"));
}
```

Die zu quittierende Meldenummer angeben. Hier wird diese aus einer Variablen gelesen.

Beispiel AXC_OnBtnMsgFirst

```
{
// jump to the first message in the WinCC Alarm Control
AXC_OnBtnMsgFirst("gs_alarm_00","Control1");
}
```

Parameter der Funktion AXC_OnBtnMsgFirst:

"gs_alarm_00" ist der Name des Bildes, in dem das WinCC Alarm Control projektiert wurde.

Control1 ist der Objektname des WinCC Alarm Controls.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

Beispiel AXC_OnBtnMsgLast

```
{
// jump to the last message in the WinCC Alarm Control
AXC_OnBtnMsgLast("gs_alarm_00","Control1");
}
```

Parameter der Funktion AXC_OnBtnMsgLast:

"gs_alarm_00" ist der Name des Bildes, in dem das WinCC Alarm Control projektiert wurde.

Control1 ist der Objektname des WinCC Alarm Controls.

Beispiel AXC_OnBtnScroll

```
{
// activate/deactivate the scroll function
AXC_OnBtnScroll("gs_alarm_00","Control1");
}
```

Parameter der Funktion AXC_OnBtnScroll:

"gs_alarm_00" ist der Name des Bildes, in dem das WinCC Alarm Control projektiert wurde.

Control1 ist der Objektname des WinCC Alarm Controls.

Beispiel AXC_OnBtnSinglAckn

```
{
// acknowledge the active message
AXC_OnBtnSinglAckn("gs_alarm_00","Control1");
}
```

Parameter der Funktion AXC_OnBtnSinglAckn:

"gs_alarm_00" ist der Name des Bildes, in dem das WinCC Alarm Control projektiert wurde.

Control1 ist der Objektname des WinCC Alarm Controls.

Beispiel AXC_SetFilter

```
{
BOOL ret;
MSG_FILTER_STRUCT Filter;
CMN_ERROR Error;

//Reset the filter struct
memset( &Filter, 0, sizeof( MSG_FILTER_STRUCT ) );

//Set the filter name
strcpy( Filter.szFilterName, "Control1");

// Selektionselemente wählen
Filter.dwFilter = MSG_FILTER_NR_FROM | MSG_FILTER_NR_TO;

// Meldungsnummer von
Filter.dwMsgNr[0] = 2;
// Meldungsnummer bis
Filter.dwMsgNr[1] = 2;

ret = AXC_SetFilter("gs_alarm_00","Control1",&Filter,&Error);
}
```

1. Den Filter benennen.
2. Die Art des Filters wählen.
3. Die Filterkriterien angeben.
4. Den Filter setzen.

Hinweis

Der Filtertyp und die Filterkriterien sind anzupassen, alle weiteren Filtertypen sind in der Filterstruktur beschrieben.

Beispiel DeactivateRTProject

```
{
//deactivate the runtime
DeactivateRTProject ();
}
```

Diese Funktion deaktiviert das WinCC Runtime.

Beispiel ExitWinCC

```
{
//exit wincc
ExitWinCC ();
}
```

Diese Funktion beendet WinCC.

2.15.4.2 Beispiele - GetAlarmHigh bis GetPropChar

Beispiel GetAlarmHigh

```
{
double dAlarmHigh;
//Get the Alarm High Limit
dAlarmHigh = GetAlarmHigh(lpszPictureName,"Balken1");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetAlarmHigh:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Balken1" ist der Name des Objekts.

1. Die obere Alarmgrenze auslesen und in dAlarmHigh zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetBackColor

```
{
long int bk_color;

//Get the backgroundcolor
bk_color = GetBackColor(lpszPictureName,"StatischerText1");

//User defined code where the
//user can do something with the returnvalue
...
```

```
}
```

Parameter der Funktion GetBackColor:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"StatischerText1" ist der Name des Objekts.

1. Die aktuelle Hintergrundfarbe auslesen und in bk_color zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetBorderStyle

```
{
long int lstyle;

//Get the current border style
lstyle = GetBorderStyle(lpszPictureName,"Rechteck1");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetBorderStyle:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Rechteck1" ist der Name des Objekts.

1. Die aktuelle Linienart des Objekts auslesen und in lstyle zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetFilling

```
{
BOOL bfilling;

//Get the actual state of dynamic filling
bfilling = GetFilling(lpszPictureName,"Rechteck1");

if(bfilling)
{
  // User defined code if the
```

```
  // dynamic filling is activated
  ...
}
Else
{
  // User defined code if the
  // dynamic filling is deactivated
  ...
}
}
```

Parameter der Funktion GetFilling:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Rechteck1" ist der Name des Objekts.

1. Auslesen, ob das dynamische Füllen aktiviert ist oder nicht, und in bfilling zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code, abhängig vom Rückgabewert der Funktion.

Beispiel GetFillingIndex

```
{
long int filling_index;

//Get the actual filling index of the object
filling_index = GetFillingIndex(lpszPictureName,"Rechteck1");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetFillingIndex:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Rechteck1" ist der Name des Objekts.

1. Den aktuellen Füllstand des Objekts auslesen und in filling_index zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetFillStyle

```
{
long int lstyle;

//Get the current fill style
lstyle = GetFillStyle(lpszPictureName,"Rechteck1");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetFillStyle:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Rechteck1" ist der Name des Objekts.

1. Das aktuelle Füllmuster des Objekts auslesen und in lstyle zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetFlashBackColor

```
{
BOOL bflash_col;

//Get if the flashing is on or off
bflash_col = GetFlashBackColor(lpszPictureName,"Gruppe1");

if(bflash_col)
{
  // User defined code if the
  // flashing is activated
  ...
}
Else
{
  // User defined code if the
  // flashing is deactivated
  ...
}
}
```

Parameter der Funktion GetFlashBackColor:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Gruppe1" ist der Name des Objekts.

1. Auslesen, ob das Blinken der Hintergrundfarbe aktiviert ist oder nicht, und in bflash_col zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code, abhängig vom Rückgabewert der Funktion.

Beispiel GetFlashBackColorOn

```
{
long int flashcol_on;

//Get the BackFlashColor
flashcol_on = GetBackFlashColorOn(lpszPictureName,"Gruppe1");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetBackFlashColorOn:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Gruppe1" ist der Name des Objekts.

1. Die aktuelle Hintergrundblinkfarbe für den Zustand "On" des Objekts auslesen und in flashcol_on zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetFlashRateFlashPic

```
{
long lFlashRate;

//Get the flashrate
lFlashRate = GetFlashRateFlashPic(lpszPictureName,"Zustandsanzeige1");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetFlashRateFlashPic:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Zustandsanzeige1" ist der Name des Objekts.

1. Die aktuelle Blinkfrequenz des Objekts auslesen und in lFlashRate zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetFocus

```
{
char* pszValue = NULL;
char szValue[_MAX_PATH+1];


//Get the Object which has the focus
pszValue = Get_Focus();

//Copy the string
if(pszValue != NULL)
{
  strncpy(szValue,pszValue,_MAX_PATH);
}
//User defined code where the
//user can do something with the returnvalue
...
}
```

1. Auslesen, auf welchem Objekt der Fokus liegt, und in pszValue zwischenspeichern.
2. Wenn ein gültiger Wert zurückgegeben wurde, den Rückgabewert der Funktion in der lokalen Zeichenfolge szValue speichern. Es werden maximal _MAX_PATH Zeichen gespeichert.
3. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetFontBold

```
{
BOOL bbold;

//Get if the text is bold
bbold = GetFontBold(lpszPictureName,"StatischerText1");

if(bbold)
{
  // User defined code if the
  // font is bold
  ...
```

```
}
Else
{
  // User defined code if the
  // font is not bold
  ...
}
}
```

Parameter der Funktion GetFontBold:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"StatischerText1" ist der Name des Objekts.

1. Auslesen, ob der Text fett ist oder nicht, und in bbold zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code, abhängig vom Rückgabewert der Funktion.

Beispiel GetFontSize

```
{
long int fontsize;

//Get the actual Font size
fontsize = GetFontSize(lpszPictureName,"StatischerText1");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetFontSize:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"StatischerText1" ist der Name des Objekts.

1. Den aktuellen Schriftgrad auslesen und in fontsize zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetHeight

```
{
long lHeight;
```

```
//Get the height of the object
lHeight = GetHeight(lpszPictureName,"WinCCLogo");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetHeight:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"WinCCLogo" ist der Name des Objekts.

1. Die aktuelle Höhe des Objekts auslesen und in lHeight zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetHiddenInput

```
{
BOOL bHiddenInput;

//Get the state of hidden input
bHiddenInput = GetHiddenInput(lpszPictureName,"EAFeld1");

if(bHiddenInput)
{
  // User defined code if the
  // hidden input is activated
  ...
}
Else
{
  // User defined code if the
  // hidden input is activated
  ...
}
}
```

Parameter der Funktion GetHiddenInput:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"EAFeld1" ist der Name des Objekts.

1. Auslesen, ob der Text fett ist oder nicht, und in bHiddenInput zwischenspeichern.

2. Ausführen von benutzerdefiniertem Code, abhängig vom Rückgabewert der Funktion.

Beispiel GetLanguage

```
{
DWORD rt_language;

//Get the current language
rt_language = GetLanguage ();

//User defined code where the
//user can do something with the returnvalue
...
}
```

1. Die aktuelle Runtimesprache auslesen und in rt_language zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetLeft

```
{
long lPos;

//Get the x-position of the object
lPos = GetLeft(lpszPictureName,"WinCCLogo");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetLeft:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"WinCCLogo" ist der Name des Objekts.

1. Die aktuelle X-Position des Objekts auslesen und in lPos zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetLink

```
{
LINKINFO linkinfo;

//Get the linked Tag
GetLink(lpszPictureName,"Balken1","Process",&linkinfo);

// linkinfo.szLinkName is the tag name
// linkinfo.dwCycle is the update cycle
// linkinfo.LinkType is the type of the connection


//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetLink:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Balken1" ist der Name des Objekts.

"Process" ist die Eigenschaft, die mit einer Variable verbunden ist.

"&linkinfo" ist die Adresse der Struktur linkinfo.

1. Füllt die übergebene Struktur linkinfo mit den Informationen der Variablenanbindung.
2. Ausführen von benutzerdefiniertem Code, abhängig vom Rückgabewert der Funktion.

Beispiel GetLinkedVariable

```
{
char* pszVarName = NULL;
char szVarName[_MAX_PATH+1];

//Get the TagName
pszVarName = GetLinkedVariable("gs_stand_graph_00","StatischerText6","Visible");

//Copy the string
if (strcmp (pszVarName,"")!= 0)
{
  strncpy(szVarName,pszVarName,_MAX_PATH);
}
else printf("Das Attribut 'visible' ist nicht dynamisiert\r\n");
}
//User defined code where the
//user can do something with the returnvalue
...
```

```
}
```

Parameter der Funktion GetLinkedVariable:

"gs_stand_graph_00" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"StatischerText6" ist der Name des Objekts.

"Visible" ist die Eigenschaft, die mit einer Variable verbunden ist.

1. Den Rückgabewert der Funktion GetLinkedVariable in pszVarName zwischenspeichern.
2. Falls ein gültiger Wert zurückgegeben wurde, den Rückgabewert in szVarName speichern. Es werden maximal _MAX_PATH Zeichen gespeichert.
3. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetLocalPicture

```
{
char* pszPicName = NULL;
char szPicName[_MAX_PATH+1];

//Get the Local Picture
pszPicName = GetLocalPicture(lpszPictureName);

//Copy the string
if (pszPicName != NULL)
{
  strncpy(szPicName,pszPicName,_MAX_PATH);
}
//User defined code where the
//user can do something with the returnvalue
...
}
```

1. Den Rückgabewert der Funktion GetLocalPicture in pszPicName zwischenspeichern.
2. Falls ein gültiger Wert zurückgegeben wurde, den Rückgabewert in szPicName speichern. Es werden maximal _MAX_PATH Zeichen gespeichert.
3. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetMarker

```
{
BOOL bmarker;
```

```
//Get the state of the Marker
bmarker = GetMarker(lpszPictureName,"Balken1");

if(bmarker)
{
  // User defined code if the
  // marker is activated
  ...
}
Else
{
  // User defined code if the
  // marker is deactivated
  ...
}
}
```

Parameter der Funktion GetMarker:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Balken1" ist der Name des Objekts.

1. Auslesen, ob der Marker eingeblendet ist oder nicht, und in bmarker zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code, abhängig vom Rückgabewert der Funktion.

Beispiel GetOutputValueDouble

```
{
double doutput;

//Get the output value of the EA Field 1
doutput = GetOutputValueDouble(lpszPictureName,"EAFeld1");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetOutputValueDouble:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"EAFeld1" ist der Name des Objekts.

1. Auslesen des Ausgabewertes und in doutput zwischenspeichern.

2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetParentPicture

```
{
char* pszPicName = NULL;
char szPicName[_MAX_PATH+1];

//Get the parent picture
pszPicName = GetParentPicture(lpszPictureName);

//Copy the string
if (pszPicName != NULL)
{
  strncpy(szPicName,pszPicName,_MAX_PATH);
}
//User defined code where the
//user can do something with the returnvalue
...
}
```

1. Den Rückgabewert der Funktion GetParentPicture in pszPicName zwischenspeichern.
2. Falls ein gültiger Wert zurückgegeben wurde, den Rückgabewert in szPicName speichern. Es werden maximal _MAX_PATH Zeichen gespeichert.
3. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetPictureDown

```
{
char* pszPicName = NULL;
char szPicName[_MAX_PATH+1];

//Get the current picture name
pszPicName = GetPictureDown(lpszPictureName,"Rundbutton1");

if(pszPicName != NULL)
{
//Copy the string
strncpy(szPicName,pszPicName,_MAX_PATH);
}

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetPictureDown:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Rundbutton1" ist der Name des Objekts.

1. Den Bildnamen des im Rundbutton1 angezeigten Bildes auslesen und in pszPicName zwischenspeichern.

2. Wenn ein gültiger Wert zurückgegeben wurde, den Rückgabewert der Funktion in der lokalen Zeichenfolge szPicName speichern. Es werden maximal _MAX_PATH Zeichen gespeichert.

3. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetPictureName

```
{
char* pszPictureName = NULL;
char szPictureName[_MAX_PATH + 1];

//Get the current PictureName
pszPictureName = GetPictureName(lpszPictureName,"GraphikObjekt1");

if(pszPictureName != NULL)
{
//copy the string
strncpy(szPictureName,pszPictureName,_MAX_PATH);
}
//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetPictureName:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"GraphikObjekt1" ist der Name des Objekts.

1. Den Bildnamen des im GraphikObjekt1 angezeigten Bildes auslesen und in pszPictureName zwischenspeichern.

2. Wenn ein gültiger Wert zurückgegeben wurde, den Rückgabewert der Funktion in der lokalen Zeichenfolge szPictureName speichern. Es werden maximal _MAX_PATH Zeichen gespeichert.

3. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetPictureUp

```
{
char* pszPicName = NULL;
char szPicName[_MAX_PATH+1];

//Get the current picture name
pszPicName = GetPictureUp(lpszPictureName,"Rundbutton1");

if(pszPicName != NULL)
{
//Copy the string
strncpy(szPicName,pszPicName,_MAX_PATH);
}

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetPictureUp:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Rundbutton1" ist der Name des Objekts.

1. Den Bildnamen des im Rundbutton1 angezeigten Bildes auslesen und in pszPicName zwischenspeichern.

2. Wenn ein gültiger Wert zurückgegeben wurde, den Rückgabewert der Funktion in der lokalen Zeichenfolge szPicName speichern. Es werden maximal _MAX_PATH Zeichen gespeichert.

3. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetPosition

```
{
long int lpos;

//Get the actual position of the Slider
lpos = GetPosition(lpszPictureName,"Control1");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetPosition:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Control1" ist der Name des Objekts.

1. Die aktuelle Position des Schiebers auslesen und in lpos zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetPropBOOL

```
{
BOOL bProp;

//Get the property Visible
bProp = GetPropBOOL("gs_graph_eafield","EAFeld1","Visible");

if(bProp)
{
  // User defined code if the
  // objekt is visible
  ...
}
else
{
  // User defined code if the
  // objekt is not visible
  ...
}
}
```

Parameter der Funktion GetVisible:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"EAFeld1" ist der Name des Objekts.

"Visible" ist die Eigenschaft des Objekts.

1. Auslesen, ob das Objekt sichtbar ist oder nicht, und in bProp zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code, abhängig vom Rückgabewert der Funktion.

Beispiel GetPropChar

```
{
char* pszProp = NULL;
char szProp[14];
```

```
//Get the property Tooltiptext
pszProp = GetPropChar("lpszPictureName","EAFeld1","Tooltiptext");

if(pszProp != NULL)
{
//Copy the string
strncpy(szProp,pszProp,13);
}
//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetPropChar:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"EAFeld1" ist der Name des Objekts.

"Tooltiptext" ist die Eigenschaft des Objekts.

1. Den Tooltipptext des Objekts auslesen und in pszProp zwischenspeichern.

2. Wenn ein gültiger Wert zurückgegeben wurde, den Rückgabewert der Funktion in der lokalen Zeichenfolge szProp speichern. Es werden maximal 13 Zeichen gespeichert.

3. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

2.15.4.3 Beispiele - GetRangeMax bis GetWidth

Beispiel GetRangeMax

```
{
long int lrange;

//Get the upper scale Limit
lrange = GetRangeMax(lpszPictureName,"Control1");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetRangeMax:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Control1" ist der Name des Objekts.

1. Die aktuelle Obergrenze des Objekts auslesen und in lrange zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetRangeMin

```
{
long int lrange;

//Get the lower scale Limit
lrange = GetRangeMin(lpszPictureName,"Control1");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetRangeMin:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Control1" ist der Name des Objekts.

1. Die aktuelle Untergrenze des Objekts auslesen und in lrange zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetScaling

```
{
BOOL bscaling;

//Get the Scaling state
bscaling = GetScaling(lpszPictureName,"Balken1");

if (bscaling)
{
  // User defined code if the
  // bar object has an additional scale
  ...
}
Else
{
  // User defined code if the
  // bar object has no additional scale
  ...
}
```

```
}
```

Parameter der Funktion GetScaling:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Balken1" ist der Name des Objekts.

1. Auslesen, ob die Skala des Balkens angezeigt wird oder nicht, und in bscaling zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code, abhängig vom Rückgabewert der Funktion.

Beispiel GetServerTagPrefix

```
{
char* pszServerPrefix;
char* pszTagPrefix;
char* pszWindowPrefix;
int nServerPrefixLen = 0;
int nTagPrefixLen = 0;
int nTagLen = 0;
char myTagName[MAX_DM_VAR_NAME+1];

//Initialize the return value
memset(myTagName,0,MAX_DM_VAR_NAME + 1);

//Get the serverprefix the tagprefix and the windowprefix
GetServerTagPrefix(&pszServerPrefix, &pszTagPrefix, &pszWindowPrefix);

//If a serverprefix exists
if (pszServerPrefix)
{
  //Get the length of the string
  nServerPrefixLen = strlen(pszServerPrefix);
}
Else
{
  printf("No server prefix was returned.");
  return;
}

//If a tagprefix exists
if (pszTagPrefix)
{
  //Get the length of the string
  nTagPrefixLen = strlen(pszTagPrefix);
}
```

```
//Get the length of the tag
nTagLen = strlen("TagName");

//Check if the lenght of the
//ServerPrefix+TagPrefix+VarName + the double points < MAX_DM_VAR_NAME)
if (nServerPrefixLen + nTagPrefixLen + nTagLen+2 < MAX_DM_VAR_NAME)
{
  sprintf(myTagName,"%s::%s%s",pszServerPrefix,pszTagPrefix,"TagName");
  //User defined code where the
  //user can do something with the returnvalue
  ...
}
Else
{
  printf("The resulting string is too long.");
  return;
}
}
```

1. Die Variable myTagName initialisieren.
2. Das Serverpräfix, das Variablenpräfix und das Windowpräfix auslesen.
3. Falls kein Serverpräfix zurückgegeben wurde, wird ein Text ausgegeben und die Funktion verlassen.
4. Falls ein Serverpräfix zurückgegeben wurde, dessen Länge ermitteln und in nServerPrefixLen zwischenspeichern.
5. Falls ein Variablenpräfix zurückgegeben wurde, dessen Länge ermitteln und in nTagPrefixLen zwischenspeichern.
6. Die Länge des Variablennamens ermitteln und in nVarLen zwischenspeichern.
7. Wenn die für Variablennamen zulässige Länge überschritten wird, dann wird ein Text ausgegeben und die Funktion verlassen.
8. Wenn die für Variablennamen zulässige Länge nicht überschritten wird, dann wird der für eine Client Umgebung benötigte Variablenname zusammengesetzt.
9. Ausführen von benutzerdefiniertem Code zur Verarbeitung der Rückgabewerte.

Beispiel GetServerTagPrefix

```
{
char* pszServerPrefix;
char* pszTagPrefix;
char* pszWindowPrefix;

//Get the serverprefix and the tagprefix
GetServerTagPrefix(&pszServerPrefix, &pszTagPrefix, &pszWindowPrefix);
//User defined code where the
//user can do something with the returnvalue
...
```

```
}
```

Parameter der Funktion GetServerTagPrefix:

"pszServerPrefix" ist die Variable, in die das Serverpräfix geschrieben wird.

"pszTagPrefix" ist die Variable, in die das Variablenpräfix geschrieben wird.

"pszWindowPrefix" ist die Variable, in die das Windowpräfix geschrieben wird.

1. Das Serverpräfix, das Variablenpräfix und das Windowpräfix auslesen.
2. In pszServerPrefix steht das zurückgegebene Serverpräfix.
3. In pszTagPrefix steht das zurückgegebene Variablenpräfix.
4. In pszWindowPrefix steht das zurückgegebene Windowpräfix.
5. Ausführen von benutzerdefiniertem Code zur Verarbeitung der Rückgabewerte.

Beispiel GetTagBit

```
{
BOOL bstate;

//Get the current state of the tag
bstate = GetTagBit("gs_tag_bit");

if(bstate)
{
  // User defined code if the
  // value of the tag is true
  ...
}
else
{
  // User defined code if the
  // value of the tag is false
  ...
}
}
```

Parameter der Funktion GetTagBit:

"gs_tag_bit" ist der Name der Variablen.

1. Den Wert der Variablen auslesen und in bstate zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code, abhängig vom Rückgabewert der Funktion.

Beispiel GetTagBitStateQC

```
{
DWORD dwState;
DWORD dwQC;
BOOL bValue;

dwState = 0xFFFFFFFF;

//Get the tag value
//dwstate is the tag state
bValue = GetTagBitStateQCWait("gs_tag_bit",&dwState,&dwQC);

//Create a string which includes the tag value
if (bValue)
{
  // User defined code if the
  // value of the tag is true
  ...
}
else
{
  // User defined code if the
  // value of the tag is false
  ...
}
}
```

Parameter der Funktion GetTagBitStateQC:

"gs_tag_bit" ist der Name der Variablen.

"&dwState" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

"&dwQC" ist die Adresse der Variablen, in die der Quality Code abgelegt werden soll.

1. Den Wert der Variablen auslesen und in bValue zwischenspeichern. Die Funktion legt den Variablenstatus in dwState und den Quality Code in dwQC ab.
2. Ausführen von benutzerdefiniertem Code, abhängig vom Rückgabewert der Funktion.

Beispiel GetTagBitStateWait

```
{
DWORD dwstate;
```

```
BOOL bValue;

dwstate = 0xFFFFFFFF;

//Get the tag value
//dwstate is the tag state
bValue = GetTagBitStateWait("gs_tag_bit",&dwstate);

//Create a string which includes the tag value
if (bValue)
{
  // User defined code if the
  // value of the tag is true
  ...
}
else
{
  // User defined code if the
  // value of the tag is false
  ...
}
}
```

Parameter der Funktion GetTagBitStateWait:

"gs_tag_bit" ist der Name der Variablen.

"&dwstate" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

1. Den Wert der Variablen auslesen und in bstate zwischenspeichern. Die Funktion legt den Variablenstatus in dwstate ab.
2. Ausführen von benutzerdefiniertem Code, abhängig vom Rückgabewert der Funktion.

Beispiel GetTagChar

```
{
char* pszValue = NULL;
char szValue[13];

//Get the current value of the tag
pszValue = GetTagChar("gs_tag_char");

if(pszValue != NULL)
{
//Copy the string
strncpy(szValue,pszValue,12);
}
//User defined code where the
//user can do something with the returnvalue
...
```

```
}
```

Parameter der Funktion GetTagChar:

"gs_tag_char" ist der Name der Variablen.

1. Den Wert der Variablen auslesen und in pszValue zwischenspeichern.

2. Wenn ein gültiger Wert zurückgegeben wurde, den Rückgabewert der Funktion in der lokalen Zeichenfolge szValue speichern. Es werden maximal 12 Zeichen gespeichert.

3. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetTagCharStateQCWait

```
{
DWORD dwState;
DWORD dwQC;
char* pszRetValue = NULL;
char szRetValue[13];

dwState = 0xFFFFFFFF;

//Get the tag value
pszRetValue = GetTagCharStateQCWait("gs_tag_char",&dwState, &dwQC);

if (pszRetValue != NULL)
{
//Copy the string
strncpy(szRetValue,pszRetValue,12);
}
//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagCharStateQCWait:

"gs_tag_char" ist der Name der Variablen.

"&dwState" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

"&dwQC" ist die Adresse der Variablen, in die der Quality Code abgelegt werden soll.

1. Den Wert der Variablen auslesen und in pszRetValue zwischenspeichern. Die Funktion legt den Variablenstatus in dwState und den Quality Code in dwQC ab.

2. Wenn ein gültiger Wert zurückgegeben wurde, den Rückgabewert der Funktion in der lokalen Zeichenfolge szRetValue speichern. Es werden maximal 12 Zeichen gespeichert.

3. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetTagCharStateWait

```
{
DWORD dwstate;
char szValue[11];
char* pszRetValue = NULL;
char szRetValue[13];

dwstate = 0xFFFFFFFF;
//Get the tag value
//dwstate is the tag state
pszRetValue = GetTagCharStateWait("gs_tag_char",&dwstate);

if (pszRetValue != NULL)
{
//Copy the string
strncpy(szRetValue,pszRetValue,12);
}
//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagCharStateWait:

"gs_tag_char" ist der Name der Variablen.

"&dwstate" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

1. Den Wert der Variablen auslesen und in pszRetValue zwischenspeichern. Die Funktion legt den Variablenstatus in dwstate ab.

2. Wenn ein gültiger Wert zurückgegeben wurde, den Rückgabewert der Funktion in der lokalen Zeichenfolge szRetValue speichern. Es werden maximal 12 Zeichen gespeichert.

3. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetTagFloat

```
{
float fValue;

//Get the current value of the tag
fValue = GetTagFloat("gs_tag_float");

//User defined code where the
//user can do something with the returnvalue
```

```
...
}
```

Parameter der Funktion GetTagFloat:

"gs_tag_float" ist der Name der Variablen.

1. Den Wert der Variablen auslesen und in fValue zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetTagFloatStateQCWait

```
{
DWORD dwState;
DWORD dwQC;
float fValue;

dwState = 0xFFFFFFFF;

//Get the tag value
fValue = GetTagFloatStateQCWait("gs_tag_float",&dwState, &dwQC);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagFloatStateQCWait:

"gs_tag_float" ist der Name der Variablen.

"&dwState" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

"&dwQC" ist die Adresse der Variablen, in die der Quality Code abgelegt werden soll.

1. Den Wert der Variablen auslesen und in fValue zwischenspeichern. Die Funktion legt den Variablenstatus in dwState und den Quality Code in dwQC ab.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetTagFloatStateWait

```
{
DWORD dwstate;
float fValue;
```

```
dwstate = 0xFFFFFFFF;
//Get the tag value
//dwstate is the tag state
fValue = GetTagFloatStateWait("gs_tag_float",&dwstate);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagFloatStateWait:

"gs_tag_float" ist der Name der Variablen.

"&dwstate" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

1. Den Wert der Variablen auslesen und in fValue zwischenspeichern. Die Funktion legt den Variablenstatus in dwstate ab.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetTagMultiStateQCWait

```
{
#define DATA_SIZE 5
DWORD dwState[DATA_SIZE];
DWORD dwQC[DATA_SIZE];

//define all Datas
BOOL lValue1;
long lValue2 ;
char* szValue3;
double dblValue4 ;
WORD lValue5 ;


//Set the tags
GetTagMultiStateQCWait(dwState,dwQC,"%d%d%s%f%d",
  "gs_tag_bit",&lValue1,
  "gs_tag_SByte",&lValue2,
  "gs_tag_char",&szValue3,
  "gs_tag_float",&dblValue4,
  "gs_tag_word",&lValue5);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagMultiStateWait:

"dwState" ist das DWord-Array, in das die Variablenstati gespeichert werden.

"dwQC" ist das DWord-Array, in das die Quality Codes gespeichert werden.

"%d%d%s%f%d" sind die Typbeschreibungen der auszulesenden Variablen.

"gs_tag_bit" ist die Variable, die gelesen werden soll.

"&lValue1" ist die Adresse der Variablen, in die der Wert der Variablen gs_tag_bit abgelegt werden soll.

"gs_tag_SByte" ist die Variable, die gelesen werden soll.

"&lValue2" ist die Adresse der Variablen, in die der Wert der Variablen gs_tag_SByte abgelegt werden soll.

Die weiteren Parameter sind analog zu den zuvor beschriebenen zu behandeln.

1. Anlegen eines DWord-Arrays mit der benötigten Größe (Anzahl der Variablen).

2. Die Werte der Variablen einlesen und zwischenspeichern. Der Wert der Variablen gs_tag_bit wird in lValue1 zwischengespeichert. Der Wert der Variablen gs_tag_SByte wird in lValue2 zwischengespeichert usw.

3. Ausführen von benutzerdefiniertem Code zur Verarbeitung der Rückgabewerte.

Beispiel GetTagMultiStateWait

```
{
#define DATA_SIZE 5
DWORD dwData[DATA_SIZE];

//define all Datas
BOOL lValue1;
long lValue2 ;
char* szValue3;
double dblValue4 ;
WORD lValue5 ;


//Set the tags
GetTagMultiStateWait(dwData,"%d%d%s%f%d",
  "gs_tag_bit",&lValue1,
  "gs_tag_SByte",&lValue2,
  "gs_tag_char",&szValue3,
  "gs_tag_float",&dblValue4,
  "gs_tag_word",&lValue5);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagMultiStateWait:

"dwData" ist das DWord-Array, in das die Variablenstati gespeichert werden.

"%d%d%s%f%d" sind die Typbeschreibungen der auszulesenden Variablen.

"gs_tag_bit" ist die Variable, die gelesen werden soll.

"&lValue1" ist die Adresse der Variablen, in die der Wert der Variablen gs_tag_bit abgelegt werden soll.

"gs_tag_SByte" ist die Variable, die gelesen werden soll.

"&lValue2" ist die Adresse der Variablen, in die der Wert der Variablen gs_tag_SByte abgelegt werden soll.

Die weiteren Parameter sind analog zu den zuvor beschriebenen zu behandeln.

1. Anlegen eines DWord-Arrays mit der benötigten Größe (Anzahl der Variablen).

2. Die Werte der Variablen einlesen und zwischenspeichern. Der Wert der Variablen gs_tag_bit wird in lValue1 zwischengespeichert. Der Wert der Variablen gs_tag_SByte wird in lValue2 zwischengespeichert usw.

3. Ausführen von benutzerdefiniertem Code zur Verarbeitung der Rückgabewerte.

Beispiel GetTagMultiWait

```
DWORD dwVar1Value;
char* szVar2Value;
//Speicher für den Variablenwert wird
//durch die Funktion mit SysMalloc angelegt
double dbVar3Value;

BOOL ok;

ok=GetTagMultiWait("%d%s%f", "Ernie_word", &dwVar1Value,
  "Ernie_char", &szVar2Value,
  "Ernie_double", &dbVar3Value);

printf("Word %d, String %s, Double %f\r\n",
  dwVar1Value, szVar2Value, dbVar3Value);
```

Beispiel GetTagPrefix

```
{
char* pszTagPrefix = NULL;
char szTagPrefix[7];

//Get the current tag prefix
pszTagPrefix = GetTagPrefix(lpszPictureName,"Bildfenster1");
```

```
if(pszTagPrefix != NULL)
{
//Copy the string
strncpy(szTagPrefix,pszTagPrefix,6);
}
//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagPrefix:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Bildfenster1" ist der Name des Objekts.

1. Das aktuelle Variablenpräfix des Bildfenster1 auslesen und in pszTagPrefix zwischenspeichern.

2. Wenn ein gültiger Wert zurückgegeben wurde, den Rückgabewert der Funktion in der lokalen Zeichenfolge szTagPrefix speichern. Es werden maximal 6 Zeichen gespeichert.

3. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetTagRaw

```
{
#define DATA_SIZE 3

BYTE byData[DATA_SIZE];

//Get the current values of the tag
GetTagRaw("gs_tag_raw",byData,DATA_SIZE);

//Use the values received in the array byData
...
}
```

Parameter der Funktion GetTagRaw:

"gs_tag_raw" ist der Name der Variablen.

"byData" ist das Byte-Array, in das die Werte der Rohdatenvariablen gespeichert werden.

"DATA_SIZE" ist die Anzahl der Werte, die gelesen werden.

1. Die Werte der Variablen auslesen und in byData zwischenspeichern.

2. Ausführen von benutzerdefiniertem Code zur Verarbeitung der Rückgabewerte.

Beispiel GetTagRawStateQCWait

```
{
#define DATA_SIZE 3
DWORD dwState;
DWORD dwQC;
BYTE byData[DATA_SIZE];

dwState = 0xFFFFFFFF;

//Get the values of the tag
GetTagRawStateQCWait("gs_tag_raw",byData,DATA_SIZE,&dwState,&dwQC);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagRawStateQCWait:

"gs_tag_raw" ist der Name der Variablen.

"byData" ist das Byte-Array, in das die Werte der Rohdatenvariablen gespeichert werden.

"DATA_SIZE" ist die Anzahl der Werte, die gelesen werden.

"&dwState" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

"&dwQC" ist die Adresse der Variablen, in die der Quality Code abgelegt werden soll.

1. Die Werte der Variablen auslesen und in byData zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung der Rückgabewerte.

Beispiel GetTagRawStateWait

```
{
#define DATA_SIZE 3
DWORD dwstate;
BYTE byData[DATA_SIZE];
char szValue[11];

//Load dwState with default values
dwstate = 0xFFFFFFFF;

//Get the values of the tag
//dwstate is the tag state
GetTagRawStateWait("gs_tag_raw",byData,DATA_SIZE,&dwstate);
```

```
//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagRawStateWait:

"gs_tag_raw" ist der Name der Variablen.

"byData" ist das Byte-Array, in das die Werte der Rohdatenvariablen gespeichert werden.

"DATA_SIZE" ist die Anzahl der Werte, die gelesen werden.

"&dwstate" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

1. Die Werte der Variablen auslesen und in byData zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung der Rückgabewerte.

Beispiel GetTagSByte

```
{
long lValue;

//Get the current value of the tag
lValue = GetTagSByte("gs_tag_SByte");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagSByte:

"gs_tag_SByte" ist der Name der Variablen.

1. Den Wert der Variablen auslesen und in lValue zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetTagSByteStateQCWait

```
{
DWORD dwState;
DWORD dwQC;
```

```
long lValue;

dwState = 0xFFFFFFFF;

//Get the tag value
lValue = GetTagSByteStateQCWait("gs_tag_SByte",&dwState, &dwQC);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagSByteStateQCWait:

"gs_tag_SByte" ist der Name der Variablen.

"&dwState" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

"&dwQC" ist die Adresse der Variablen, in die der Quality Code abgelegt werden soll.

1. Den Wert der Variablen auslesen und in lValue zwischenspeichern. Die Funktion legt den Variablenstatus in dwState und den Quality Code in dwQC ab.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetTagSByteStateWait

```
{
DWORD dwstate;
long lValue;

dwstate = 0xFFFFFFFF;
//Get the tag value
//dwstate is the tag state
lValue = GetTagSByteStateWait("gs_tag_SByte",&dwstate);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagSByteStateWait:

"gs_tag_SByte" ist der Name der Variablen.

"&dwstate" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

1. Den Wert der Variablen auslesen und in lValue zwischenspeichern. Die Funktion legt den Variablenstatus in dwstate ab.

2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetTagWord

```
{
WORD wValue;

//Get the current value of the tag
wValue = GetTagWord("gs_tag_word");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagWord:

"gs_tag_word" ist der Name der Variablen.

1. Den Wert der Variablen auslesen und in wValue zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetTagWordStateQCWait

```
{
DWORD dwState;
DWORD dwQC;
WORD wValue;

dwState = 0xFFFFFFFF;

//Get the tag value
wValue = GetTagWordStateQCWait("gs_tag_word",&dwState, &dwQC);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagWordStateQCWait:

"gs_tag_word" ist der Name der Variablen.

"&dwState" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

"&dwQC" ist die Adresse der Variablen, in die der Quality Code abgelegt werden soll.

1. Den Wert der Variablen auslesen und in wValue zwischenspeichern. Die Funktion legt den Variablenstatus in dwState und den Quality Code in dwQC ab.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetTagWordStateWait

```
{
DWORD dwstate;
WORD wValue;

dwstate = 0xFFFFFFFF;
//Get the tag value
//dwstate is the tag state
wValue = GetTagWordStateWait("gs_tag_word",&dwstate);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTagWordStateWait:

"gs_tag_word" ist der Name der Variablen.

"&dwstate" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

1. Den Wert der Variablen auslesen und in wValue zwischenspeichern. Die Funktion legt den Variablenstatus in dwstate ab.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetText

```
{
char* pszValue = NULL;
char szValue[13];

//Get the Text which is actually set
pszValue = GetText(lpszPictureName,"StatischerText1");

if(pszValue != NULL)
{
  //Copy the string
  strncpy(szValue,pszValue,12);
}
//User defined code where the
```

```
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetText:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"StatischerText1" ist der Name des Objekts.

1. Den Text, der im Objekt StatischerText1 steht, auslesen und in pszValue zwischenspeichern.
2. Wenn ein gültiger Wert zurückgegeben wurde, den Rückgabewert der Funktion in der lokalen Zeichenfolge szValue speichern. Es werden maximal 12 Zeichen gespeichert.
3. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetTop

```
{
long lPos;

//Get the y-Position of the Object
lPos = GetTop(lpszPictureName,"WinCCLogo");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetTop:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"WinCCLogo" ist der Name des Objekts.

1. Die aktuelle Y-Position des Objekts auslesen und in lPos zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

Beispiel GetVisible

```
{
BOOL bVisible;

//Get the visibility
```

```
bVisible = GetVisible(lpszPictureName,"GraphikObjekt1");

if(bVisible)
{
  // User defined code if the
  // objekt is visible
  ...
}
else
{
  // User defined code if the
  // objekt is not visible
  ...
}
}
```

Parameter der Funktion GetVisible:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"GraphikObjekt1" ist der Name des Objekts.

1. Auslesen, ob das Objekt sichtbar ist oder nicht, und in bVisible zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code, abhängig vom Rückgabewert der Funktion.

Beispiel GetWidth

```
{
long lWidth;

//Get the width of the object
lWidth = GetWidth(lpszPictureName,"WinCCLogo");

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion GetWidth:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"WinCCLogo" ist der Name des Objekts.

1. Die aktuelle Breite des Objekts auslesen und in lWidth zwischenspeichern.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Rückgabewerts.

2.15.4.4 Beispiele - H bis S

Beispiel InquireLanguage

```
{
DWORD count;
DWORD* language;
int i;

//Count the installed languages
language = InquireLanguage(&count);

printf("#################### INQUIRE LANGUAGE ###################");
//Print out the count of languages
printf ( "\r\nCount Languages=%d\r\n", count );

//print out which languages are installed
for (i=1;i<=count; i++)
{
printf ("\r\n%d.language=%x", i,*language++);
}
}
```

1. Die für die Laufzeit projektierten Sprachen ermitteln. In language werden die Sprachkennungen zwischengespeichert. In count wird die Anzahl der Sprachen zwischengespeichert.
2. Es wird die Anzahl der ermittelten Sprachen ausgegeben.
3. Es werden alle ermittelten Sprachkennungen ausgegeben.

Beispiel ProgramExecute

```
{
//start the program calc.exe
ProgramExecute("C:\\Winnt\\system32\\calc.exe");
}
```

Als Parameter ist die Datei mit ihrem Pfad anzugeben.

Beispiel ResetFilter

```
{
BOOL ret;
MSG_FILTER_STRUCT Filter;
```

```
CMN_ERROR Error;

//delete the whole Filter struct
memset( &Filter, 0, sizeof( MSG_FILTER_STRUCT ) );

//set an empty filter struct
AXC_SetFilter("gs_alarm_00","Control1",&Filter,&Error);
}
```

1. Die Filterstruktur löschen.
2. Die Filterstruktur mit leeren Werten beschreiben.

Beispiel RPTJobPreview

```
{
//Start the print preview of the specified print job
RPTJobPreview("Documentation Text Library");
}
```

Parameter der Funktion "RPTJobPreview":

"Documentation Text Library" ist der Name des Druckauftrags.

Beispiel RPTJobPrint

```
{
//Print the specified print job out
RPTJobPrint("@Text library (compact)");
}
```

Parameter der Funktion RPTJobPrint:

@Text library (compact) ist der Name des Druckauftrags.

Beispiel SysMalloc

```
char* main(...);
{
char* returnwert;
char text[17];
returnwert=SysMalloc(17);
strcpy(returnwert,&text[0];
return returnwert;
```

```
}
```

2.15.4.5 Beispiele - SetAlarmHigh bis SetPropChar

Beispiel SetAlarmHigh

```
{
//Set the upper limit for the warning
SetAlarmHigh(lpszPictureName,"Balken1",3.0);
}
```

Parameter der Funktion SetAlarmHigh:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Balken1" ist der Name des Objekts.

"3.0" ist der Wert, auf den die obere Alarmgrenze gesetzt wird.

Beispiel SetBackColor

```
{
//Set the back color blue
SetBackColor(lpszPictureName,"StatischerText1",CO_BLUE);
}
```

Parameter der Funktion SetBackColor:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"StatischerText1" ist der Name des Objekts.

"CO_BLUE" ist die Konstante für die Farbe Blau.

Hinweis

Statt der Konstanten für den Farbwert können Sie die Farbe auch in einem Hexadezimalwert angeben.

Beispiel SetBorderEndStyle

```
{
```

```
SetBorderEndStyle(lpszPictureName,"Line", (2|393216));
}
```

Setzt das linke Linienende als ausgefüllten Pfeil und das rechte Linienende als ausgefüllten Kreis. Das linke Linienende wird in den beiden unteren Bytes, dass rechte Linienende in den beiden oberen Bytes gespeichert. Die Parameterübergabe erfolgt mittels Zahlenwerten.

Beispiel für das Setzen der Linienenden mit symbolischen Bezeichnungen

```
{
SetBorderEndStyle(lpszPictureName,"Line", (LE_FULL_ARROW|(LE_FULL_CIRCLE <<16));
}
```

Setzt das linke Linienende als ausgefüllten Pfeil und das rechte Linienende als ausgefüllten Kreis. Das linke Linienende wird in den beiden unteren Bytes, dass rechte Linienende in den beiden oberen Bytes gespeichert. Um das rechte Linienende anzusprechen wird die symbolische Bezeichnung "LE_FULL_CIRCLE" um 2 Byte bzw. 16 Bit in die beiden oberen Bytes verschoben.

Beispiel SetBorderStyle

```
{
//Change the Border style
SetBorderStyle(lpszPictureName,"Rechteck1",3);
}
```

Parameter der Funktion SetBorderStyle:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Rechteck1" ist der Name des Objekts.

"3" ist die Linienart, die für das Objekt eingestellt wird.

Beispiel SetColorAlarmHigh

```
{
//Set the Color for the alarm high limit to red
SetColorAlarmHigh(lpszPictureName,"Balken1",CO_RED);
}
```

Parameter der Funktion SetColorAlarmHigh:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Balken1" ist der Name des Objekts.

"CO_RED" ist die Konstante für die Farbe Rot.

Hinweis

Statt der Konstanten für den Farbwert können Sie die Farbe auch in einem Hexadezimalwert angeben.

Beispiel SetCursorMode

```
{
//Set the Cursor Mode to Alpha cursor
SetCursorMode(lpszPictureName,"GraphikObjekt1",FALSE);
}
```

Parameter der Funktion SetCursorMode:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"GraphikObjekt1" ist der Name des Objekts.

"FALSE" bedeutet: Der Cursormodus "Alpha-Cursor" wird eingestellt.

Beispiel SetFilling

```
{
//Set the dynamic filling true
SetFilling(lpszPictureName,"Rechteck1",TRUE);
}
```

Parameter der Funktion SetFilling:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Rechteck1" ist der Name des Objekts.

"TRUE" bedeutet: Das dynamische Füllen aktivieren.

Beispiel SetFillingIndex

```
{
//Set the Filling of the Rechteck1 to 10
SetFillingIndex(lpszPictureName,"Rechteck1",10);
}
```

Parameter der Funktion SetFillingIndex:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Rechteck1" ist der Name des Objekts.

"10" ist der Füllstand, der dem Objekt zugewiesen wird.

Beispiel SetFillStyle

```
{
//Change the fill style
SetFillStyle(lpszPictureName,"Rechteck1",196617);
}
```

Parameter der Funktion SetFillStyle:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Rechteck1" ist der Name des Objekts.

"196617" ist das Füllmuster (Backsteinmauer), das für das Objekt eingestellt wird.

Beispiel SetFlashBackColor

```
{
//Set the flashing to True
SetFlashBackColor(lpszPictureName,"Gruppe1",TRUE);
}
```

Parameter der Funktion SetFlashBackColor:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Gruppe1" ist der Name des Objekts.

"TRUE" bedeutet: Das Blinken der Hintergrundfarbe aktivieren.

Beispiel SetFlashBackColorOn

```
{
//Set the Flasch color for the state on to red
SetBackFlashColorOn(lpszPictureName,"Gruppe1",CO_RED);
}
```

Parameter der Funktion SetBackFlashColorOn:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Gruppe1" ist der Name des Objekts.

"CO_Red" ist die Konstante für die Farbe Rot.

Hinweis

Statt der Konstanten für den Farbwert können Sie die Farbe auch in einem Hexadezimalwert angeben.

Beispiel SetFlashRateFlashPic

```
{
//Set the flash rate to 0
SetFlashRateFlashPic(lpszPictureName,"Zustandsanzeige1",0);
}
```

Parameter der Funktion SetFlashRateFlashPic:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Zustandsanzeige1" ist der Name des Objekts.

"0" ist die Blinkfrequenz mit der das Objekt blinken soll.

Beispiel SetFocus

```
{
//Set the Focus on the Object Button 1
Set_Focus(lpszPictureName,"Button1");
}
```

Parameter der Funktion Set_Focus:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Button1" ist der Name des Objekts, auf das der Fokus gesetzt wird.

Beispiel SetFontBold

```
{
//Set the displayed Text bold
SetFontBold(lpszPictureName,"StatischerText1",TRUE);
}
```

Parameter der Funktion SetFontBold:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"StatischerText1" ist der Name des Objekts.

"TRUE" bedeutet: Den Text fett schreiben.

Beispiel SetFontSize

```
{
//Set Font Size to 12
SetFontSize(lpszPictureName,"StatischerText1",12);
}
```

Parameter der Funktion SetFontSize:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"StatischerText1" ist der Name des Objekts.

"12" ist der Schriftgrad, auf den der Text gesetzt wird.

Beispiel SetHeight

```
{
//Set the height of the object to 100
SetHeight(lpszPictureName,"WinCCLogo",100);
}
```

Parameter der Funktion SetHeight:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"WinCCLogo" ist der Name des Objekts.

"100" ist die Höhe, auf die das Objekt gesetzt wird.

Beispiel SetHiddenInput

```
{
//Set the hidden input true
SetHiddenInput(lpszPictureName,"EAFeld1",TRUE);
}
```

Parameter der Funktion SetHiddenInput:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"EAField1" ist der Name des Objekts.

"TRUE" bedeutet: Die verdeckte Eingabe aktivieren.

Beispiel SetLanguage

```
{
//German
SetLanguage(0x0407);
}
```

Die Runtime-Sprache wird auf Deutsch gesetzt.

Beispiel SetLeft

```
{
//Set the x-position to 0
SetLeft(lpszPictureName,"WinCCLogo",0);
}
```

Parameter der Funktion SetLeft:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"WinCCLogo" ist der Name des Objekts.

"0" ist die X-Position, auf die das Objekt gesetzt wird.

Beispiel SetLink

```
{
LINKINFO linkinfo;

//Set the link type
linkinfo.LinkType = 1;

//Set the update cycle
linkinfo.dwCycle = 0;

//set the Structmember
strcpy(linkinfo.szLinkName, "U08i_link_00");

//Set the connection to the tag
SetLink(lpszPictureName,"Balken1","Process",&linkinfo);
}
```

Parameter der Funktion SetLink:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Balken1" ist der Name des Objekts.

"Process" ist die Eigenschaft, die mit einer Variable verbunden ist.

"&linkinfo" ist die Adresse der Struktur linkinfo

1. Den Verbindungstyp an der Eigenschaft Process auf Direktverbindung setzen.
2. Den Aktualisierungszyklus auf "Bei Änderung" setzen.
3. Den Variablennamen auf U08i_link_00 setzen.

Beispiel SetMarker

```
{
//Set the marker visible
SetMarker(lpszPictureName,"Balken1",TRUE);
}
```

Parameter der Funktion SetMarker:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Balken1" ist der Name des Objekts.

"TRUE" bedeutet: Der Marker wird eingeblendet.

Beispiel SetOutputValueDouble

```
{
//Set the outputvalue of the EA Filed to 55.5
SetOutputValueDouble(lpszPictureName,"EAFeld1",55.5);
}
```

Parameter der Funktion SetOutputValueDouble:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"EAField1" ist der Name des Objekts.

"55.5" ist der Wert, der ausgegeben wird.

Beispiel SetPictureDown

```
{
//Set the picture name to activated.bmp
SetPictureDown(lpszPictureName,"Rundbutton1","activated.bmp");
}
```

Parameter der Funktion SetPictureDown:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Rundbutton1" ist der Name des Objekts.

"activated.bmp" ist der Bildname des Bildes, das im Rundbutton1 angezeigt werden soll.

Beispiel SetPictureName

```
{
//Set the picture name cool_man.bmp
SetPictureName(lpszPictureName,"GraphikObjekt1","cool_man.bmp");
}
```

Parameter der Funktion SetPictureName:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"GraphikObjekt1" ist der Name des Objekts.

"cool_man.bmp" ist der Bildname des Bildes, das im GraphikObjekt1 angezeigt werden soll.

Beispiel SetPictureUp

```
{
//Set the picture name to deactivated.bmp
SetPictureUp(lpszPictureName,"Rundbutton1","deactivated.bmp");
}
```

Parameter der Funktion SetPictureUp:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Rundbutton1" ist der Name des Objekts.

"deactivated.bmp" ist der Bildname des Bildes, das im Rundbutton1 angezeigt werden soll.

Beispiel SetPosition

```
{
//Set the Slider Position to 30
SetPosition(lpszPictureName,"Control1",30);
}
```

Parameter der Funktion SetPosition:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Control1" ist der Name des Objekts.

"30" ist die Position auf die der Schieber gesetzt werden soll.

Beispiel SetPropBOOL

```
{
//Set the visibility TRUE
SetPropBOOL("lpszPictureName","EAFeld1","Visible",TRUE);
}
```

Parameter der Funktion SetVisible:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"EAFeld1" ist der Name des Objekts.

"TRUE" bedeutet: Das Objekt soll sichtbar sein.

Beispiel SetPropChar

```
{
//Set the property Tooltiptext
SetPropChar("gs_graph_eafield","EAFeld1","ToolTipText","Tooltiptext1 ");
}
```

Parameter der Funktion SetPropChar:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"EAFeld1" ist der Name des Objekts.

"Tooltiptext" ist die Eigenschaft des Objekts.

"Tooltiptext 1" ist der Wert, auf den die Eigenschaft gesetzt werden soll.

2.15.4.6 Beispiele - SetRangeMax bis SetWidth

Beispiel SetRangeMax

```
{
//Set the Upper Scale Limit
SetRangeMax(lpszPictureName,"Control1",80);
}
```

Parameter der Funktion SetRangeMax:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Control1" ist der Name des Objekts.

"80" ist die Obergrenze, die dem Objekt zugewiesen werden soll.

Beispiel SetRangeMin

```
{
//Set the lower Scale Limit
SetRangeMin(lpszPictureName,"Control1",0);
}
```

Parameter der Funktion SetRangeMin:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Control1" ist der Name des Objekts.

"0" ist die Untergrenze, die dem Objekt zugewiesen werden soll.

Beispiel SetScaling

```
{
//Set the Scaling Visible
SetScaling(lpszPictureName,"Balken1",TRUE);

}
```

Parameter der Funktion SetScaling:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Balken1" ist der Name des Objekts.

"TRUE" bedeutet: Die Skalierung sichtbar schalten.

Beispiel SetTagBit

```
{
//Set the tag to true
SetTagBit("gs_tag_bit",TRUE);
}
```

Parameter der Funktion SetTagBit:

"gs_tag_bit" ist der Name der Variablen.

"TRUE" ist der Wert, mit dem die Variable beschrieben werden soll.

Beispiel SetTagBitStateWait

```
{
DWORD dwstate;

//Load dwState with default values
dwstate = 0xFFFFFFFF;

//Set the value of the tag to TRUE
//dwstate is the tag state
SetTagBitStateWait("gs_tag_bit",TRUE,&dwstate);

//User defined code where the
//user can do something with the returnvalue
```

```
...
}
```

Parameter der Funktion SetTagBitStateWait:

"gs_tag_bit" ist der Name der Variablen.

"TRUE" ist der Wert, mit dem die Variable beschrieben werden soll.

"&dwstate" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

1. Setzen der Variablen auf den angegebenen Wert.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Variablenstatus.

Beispiel SetTagChar

```
{
//Set the tag to Beispieltext
SetTagChar("gs_tag_char","Example Text");
}
```

Parameter der Funktion SetTagChar:

"gs_tag_char" ist der Name der Variablen.

"Example Text" ist der Wert, mit dem die Variable beschrieben werden soll.

Beispiel SetTagCharStateWait

```
{
DWORD dwstate;

//Load dwState with default values
dwstate = 0xFFFFFFFF;

//Set the tag to Example Text
//dwstate is the tag state
SetTagCharStateWait("gs_tag_char","Example Text",&dwstate);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion SetTagCharStateWait:

"gs_tag_char" ist der Name der Variablen.

"Example Text" ist der Wert, mit dem die Variable beschrieben werden soll.

"&dwstate" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

1. Setzen der Variablen auf den angegebenen Wert.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Variablenstatus.

Beispiel SetTagFloat

```
{
//Set the tag to 55.4711
SetTagFloat("gs_tag_float",55.4711);
}
```

Parameter der Funktion SetTagFloat:

"gs_tag_float" ist der Name der Variablen.

"55.4711" ist der Wert, mit dem die Variable beschrieben werden soll.

Beispiel SetTagFloatStateWait

```
{
DWORD dwstate;
char szValue[9];

//Load dwState with default values
dwstate = 0xFFFFFFFF;

//Set the tag to 55.4711
//dwstate is the tag state
SetTagFloatStateWait("gs_tag_float",55.4711,&dwstate);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion SetTagFloatStateWait:

"gs_tag_float" ist der Name der Variablen.

"55.4711" ist der Wert, mit dem die Variable beschrieben werden soll.

"&dwstate" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

1. Setzen der Variablen auf den angegebenen Wert.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Variablenstatus.

Beispiel SetTagMultiStateWait

```
{
#define DATA_SIZE 5
DWORD dwData[DATA_SIZE];

//define all tags
BOOL lValue1;
long lValue2;
char szValue3[_MAX_PATH];
float lValue4;
char lValue5;

// Fill the tags with the values
// you want to set into the WinCC tags
...

//Set the WinCC tags
SetTagMultiStateWait(dwData,"%d%d%s%f%d","gs_tag_bit",lValue1,
 "gs_tag_SByte",lValue2,
 "gs_tag_char",szValue3,
 "gs_tag_float",lValue4,
 "gs_tag_word",lValue5);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion SetTagMultiStateWait:

"dwData" ist das DWord-Array, in das die Variablenstati gespeichert werden.

"%d%d%s%f%d" sind die Typbeschreibungen der zu beschreibenden Variablen.

"gs_tag_bit" ist die WinCC Variable, die beschrieben werden soll.

"lValue1" ist die Variable, auf deren Wert die WinCC Variable gs_tag_bit gesetzt werden soll.

"gs_tag_SByte" ist die WinCC Variable, die beschrieben werden soll.

"&lValue2" ist die Variable, auf deren Wert die WinCC Variable gs_tag_SByte gesetzt werden soll.

Die weiteren Parameter sind analog zu den zuvor beschriebenen zu behandeln.

1. Anlegen eines DWord-Arrays mit der benötigten Größe (Anzahl der Variablen).
2. Anlegen von Variablen, mit deren Werten die WinCC Variablen beschrieben werden sollen.

3. Die WinCC Variablen mit den Werten der zuvor angelegten und abgefüllten Variablen beschreiben.

4. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Variablenstatus.

Beispiel SetTagMultiWait

```
BOOL ok;

ok=SetTagMultiWait("%d%s%f", "Ernie_word", 16,
  "Ernie_char", "Hallo Welt",
  "Ernie_double", 55.4711);
```

Beispiel SetTagPrefix

```
{
//Set the TagPrefix to Struct1.
SetTagPrefix(lpszPictureName,"Bildfenster1","Struct1.");

//Set the picture name again to update the tag prefix
SetPictureName(lpszPictureName,"Bildfenster1","gs_graph_eafield");
}
```

Parameter der Funktion SetTagPrefix:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"Bildfenster1" ist der Name des Objekts.

"Struct1." ist das Variablenpräfix, das am Bildfenster1 gesetzt werden soll.

1. Das Variablenpräfix des Objekts "Bildfenster1" auf "Struct1." setzen.

2. Den Namen des Bildes, das im Bildfenster angezeigt wird, erneut setzen, damit das Setzen des Variablenpräfixes wirksam wird.

Beispiel SetTagRaw

```
{
#define DATA_SIZE 3

BYTE byData[DATA_SIZE];

// Fill the Byte array with the values
// you want to set into the raw data tag
...
```

```
//Set the tag to the default values
SetTagRaw("gs_tag_raw",byData,DATA_SIZE);
}
```

Parameter der Funktion SetTagRaw:

"gs_tag_raw" ist der Name der Variablen.

"byData" ist das Byte-Array, mit dessen Werten die Rohdatenvariable beschrieben wird.

"DATA_SIZE" ist die Anzahl der Werte, die geschrieben werden.

1. Anlegen eines BYTE-Arrays mit der benötigten Größe (Größe der Rohdatenvariable).
2. Das BYTE-Array mit den zu schreibenden Werten abfüllen.
3. Die Rohdatenvariable mit den Werten des BYTE-Arrays beschreiben.

Beispiel SetTagRawStateWait

```
{
#define DATA_SIZE 3

BYTE byData[DATA_SIZE];
DWORD dwstate;
char szValue[9];

//Load dwState with default values
dwstate = 0xFFFFFFFF;

// Fill the Byte array with the values
// you want to set into the raw data tag
...

//Set the tag to the default values
//dwstate is the tag state
SetTagRawStateWait("gs_tag_raw",byData,DATA_SIZE,&dwstate);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion SetTagRawStateWait:

"gs_tag_raw" ist der Name der Variablen.

"byData" ist das Byte-Array, mit dessen Werten die Rohdatenvariable beschrieben wird.

"DATA_SIZE" ist die Anzahl der Werte, die geschrieben werden.

"&dwstate" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

1. Anlegen eines BYTE-Arrays mit der benötigten Größe (Größe der Rohdatenvariablen).
2. Das BYTE-Array mit den zu schreibenden Werten abfüllen.
3. Die Rohdatenvariable mit den Werten des BYTE-Arrays beschreiben.
4. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Variablenstatus.

Beispiel SetTagSByte

```
{
//Set the tag to 50
SetTagSByte("gs_tag_SByte",50);
}
```

Parameter der Funktion SetTagSByte:

"gs_tag_SByte" ist der Name der Variablen.

"50" ist der Wert, mit dem die Variable beschrieben werden soll.

Beispiel SetTagSByteStateWait

```
{
DWORD dwstate;
char szValue[9];

//Load dwState with default values
dwstate = 0xFFFFFFFF;

//Set the tag to 50
//dwstate is the tag state
SetTagSByteStateWait("gs_tag_SByte",50,&dwstate);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion SetTagSByteStateWait:

"gs_tag_SByte" ist der Name der Variablen.

"50" ist der Wert, mit dem die Variable beschrieben werden soll.

"&dwstate" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

1. Setzen der Variablen auf den angegebenen Wert.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Variablenstatus.

Beispiel SetTagWord

```
{
//Set the tag to 50
SetTagWord("gs_tag_word",50);
}
```

Parameter der Funktion SetTagWord:

"gs_tag_word" ist der Name der Variablen.

"50" ist der Wert, mit dem die Variable beschrieben werden soll.

Beispiel SetTagWordStateWait

```
{
DWORD dwstate;
char szValue[9];

//Load dwState with default values
dwstate = 0xFFFFFFFF;

//Set the tag to 50
//dwstate is the tag state
SetTagWordStateWait("gs_tag_word",50,&dwstate);

//User defined code where the
//user can do something with the returnvalue
...
}
```

Parameter der Funktion SetTagWordStateWait:

"gs_tag_word" ist der Name der Variablen.

"50" ist der Wert, mit dem die Variable beschrieben werden soll.

"&dwstate" ist die Adresse der Variablen, in die der Variablenstatus abgelegt werden soll.

1. Setzen der Variablen auf den angegebenen Wert.
2. Ausführen von benutzerdefiniertem Code zur Verarbeitung des Variablenstatus.

Beispiel SetText

```
{
//Set the text Beispieltext on the StaticTextfield
SetText(lpszPictureName,"StatischerText1","ExampleText");
}
```

Parameter der Funktion SetText:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"StatischerText1" ist der Name des Objekts.

"ExampleText" ist der Text, der ausgegeben werden soll.

Beispiel SetTop

```
{
//Set the y-position to 0
SetTop(lpszPictureName,"WinCCLogo",140);
}
```

Parameter der Funktion SetTop:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"WinCCLogo" ist der Name des Objekts.

"140" ist die Y-Position, auf die das Objekt gesetzt wird.

Beispiel SetVisible

```
{
//Set the Object visible
SetVisible(lpszPictureName,"GraphikObjekt1",TRUE);
}
```

Parameter der Funktion SetVisible:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"GraphikObjekt1" ist der Name des Objekts.

"TRUE" bedeutet: Das Objekt soll sichtbar sein.

Beispiel SetWidth

```
{
//Set the width of the object to 400
SetWidth(lpszPictureName,"WinCCLogo",400);
}
```

Parameter der Funktion SetWidth:

"lpszPictureName" ist der Name des Bildes, in dem das Objekt projektiert wurde.

"WinCCLogo" ist der Name des Objekts.

"400" ist die Breite, auf die das Objekt gesetzt wird.

2.15.4.7 Beispiele - T bis Z

Beispiel TlgGetNumberOfColumns

```
{
char text[5];
long int columns

//get number of Columns
columns = GetNumberOfColumns("TableControl_01");

//convert long int to char
sprintf(text,"%d",columns);

//set text on TextField5
SetText(lpszPictureName,"StatischerText5",text);
}
```

Parameter der Funktion TlgGetNumberOfColumns:

"TableControl_01" ist der Name des WinCC Table Contols.

1. Den Rückgabewert der Funktion TlgGetNumberOfColumns in columns zwischenspeichern.
2. Den Rückgabewert im String text zwischenspeichern.

3. Den Rückgabewert in einem statischen Textfeld ausgeben.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

Beispiel TlgGetNumberOfColumns

```
{
char text[5];
long int columns

//get number of Columns
columns = GetNumberOfColumns("TableControl_01");

//convert long int to char
sprintf(text,"%d",columns);

//set text on TextField5
SetText(lpszPictureName,"StatischerText5",text);
}
```

Parameter der Funktion TlgGetNumberOfColumns:

"TableControl_01" ist der Name des WinCC Table Contols.

1. Den Rückgabewert der Funktion TlgGetNumberOfColumns in columns zwischenspeichern.
2. Den Rückgabewert im String text zwischenspeichern.
3. Den Rückgabewert in einem statischen Textfeld ausgeben.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

Beispiel TlgGetNumberOfRows

```
{
char text[5];
long int rows;

//get number of rows
rows = TlgGetNumberOfRows("TableControl_01");

//convert long int to char
sprintf(text,"%d",rows);

//set text on TextField5
SetText(lpszPictureName,"StatischerText5",text);
}
```

Parameter der Funktion TlgGetNumberOfRows:

TableControl_01 ist der Name des WinCC Table Contols.

1. Den Rückgabewert der Funktion TlgGetNumberOfRows in rows zwischenspeichern.
2. Den Rückgabewert im String text zwischenspeichern.
3. Den Rückgabewert in einem statischen Textfeld ausgeben.

Beispiel TlgGetRulerTimeTrend

```
{
SYSTEMTIME systime;
WORD wHour;
WORD wMin;
WORD wSec;

char szTime[10];

//Get the current systemtime
systime = TlgGetRulerTimeTrend("TrendControl_01",0);

//Get the hour
wHour = systime.wHour;
//Get the minute
wMin = systime.wMinute;
//Get the second
wSec = systime.wSecond;

//
sprintf(szTime,"%d:%d:%d",wHour,wMin,wSec);

//output the variable name
```

```
SetText(lpszPictureName,"StatischerText7",szTime);
}
```

1. Die aktuelle Systemzeit auslesen.
2. Aus der Struktur SYSTEMTIME die Stunde, Minute und Sekunde auslesen.
3. Einen String erzeugen, der die Zeit enthält.
4. Die aktuelle Zeit ausgeben.

Beispiel TlgGetRulerVariableNameTrend

```
{
char* pszVarName = NULL;
char szVarName[20];

//Get the ruler variable name
pszVarName = TlgGetRulerVariableNameTrend("TrendControl_01",0);

if (pszVarName != NULL)
{
// Copy the string
strncpy(szVarName,pszVarName,19);
}
//output the variable name
SetText(lpszPictureName,"StatischerText6",szVarName);
}
```

Parameter der Funktion TlgGetRulerVariableNameTrend:

"TrendControl_01" ist der Name des WinCC Trend Controls.

"0" ist die Nummer der Kurve.

1. Den Rückgabewert der Funktion TlgGetRulerVariableNameTrend in pszVarName zwischenspeichern.
2. Falls ein Wert zurückgegeben wurde, den Rückgabewert in szVarName kopieren.
3. Den Rückgabewert in einem statischen Textfeld ausgeben.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

Beispiel TlgTrendWindowPressOpenDlgButton

```
{
//Opens the Property Dialog
TlgTrendWindowPressOpenDlgButton("TrendControl_01");
}
```

Parameter der Funktion TlgTrendWindowPressOpenDlgButton:

"TrendControl_01" ist der Fenstertitel des WinCC Trend Controls.

Beispiel TlgTrendWindowPressStartStopButton

```
{
//start/stop the actualisation
TlgTrendWindowPressStartStopButton("TrendControl_01");
}
```

Parameter der Funktion TlgTrendWindowPressStartStopButton:

"TrendControl_01" ist der Fenstertitel des WinCC Trend Controls.

Hinweis

Zu den Funktionsbeschreibungen werden verschiedene Beispiele angeboten. Bei Funktionen mit einer ähnlichen Syntax wird im Beispiel eine ausgewählte Funktion als Vorlage verwendet. Dieses Beispiel müssen Sie gegebenenfalls anpassen.

Beispiel TlgTrendWindowPressZoomInButton

```
{
//zoom in
TlgTrendWindowPressZoomInButton("TrendControl_01");
}
```

Parameter der Funktion TlgTrendWindowPressZoomInButton:

"TrendControl_01" ist der Fenstertitel des WinCC Trend Controls.

Beispiel TlgTrendWindowPressZoomOutButton

```
{
// zoom out
TlgTrendWindowPressZoomOutButton("TrendControl_01");
}
```

Parameter der Funktion TlgTrendWindowPressZoomOutButton:

"TrendControl_01" ist der Fenstertitel des WinCC Trend Controls.

2.15.4.8 Beispiele WinCC Controls

So fügen Sie Elemente zu einem WinCC OnlineTableControl hinzu

Einleitung

Im folgenden Beispiel fügen Sie Wertspalten mit Eigenschaften in ein leeres WinCC OnlineTableControl ein und verbinden die Spalten mit Archivvariablen.

Voraussetzung

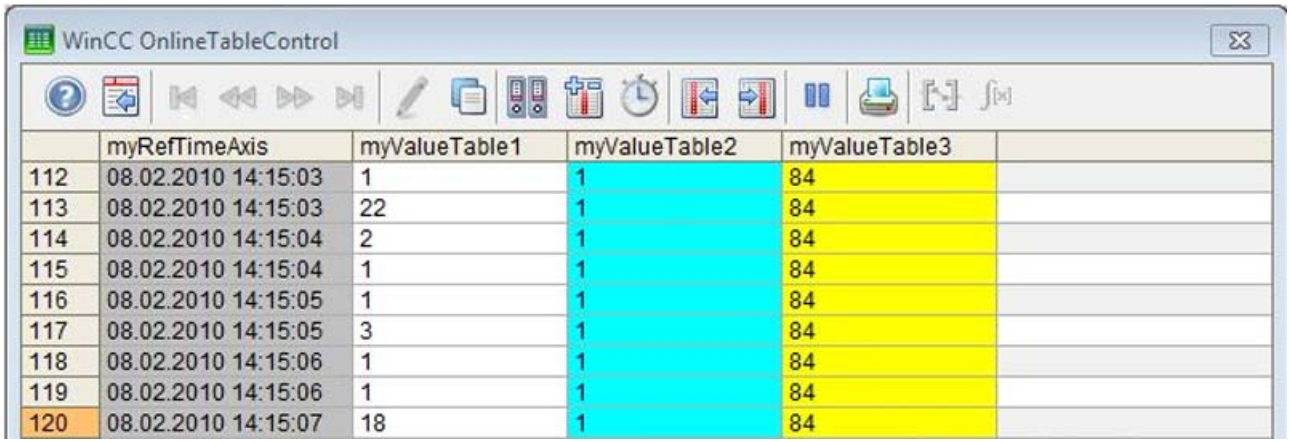
- Im Editor "Tag Logging" ist ein Archiv mit drei Archivvariablen angelegt.
- Im Graphics Designer ist ein "WinCC OnlineTableControl" mit dem Namen "Control2" in das Prozessbild eingefügt.
- Im Graphics Designer ist ein Button eingefügt. Sie haben für den Button z. B. das Ereignis "Mausklick" projektiert mit einer C-Aktion und folgendem Skript.

Beispiel

```
//enable BackColor
SetPropBOOL(lpszPictureName, "Control2", "UseColumnBackColor",TRUE);

//add new new TimeColumn and assign column length
SetPropChar(lpszPictureName, "Control2", "TimeColumnAdd", "myRefTimeColumn");
SetPropWord(lpszPictureName, "Control2", "TimeColumnLength", 20);

//add new ValueColumn and assign propertys
SetPropChar(lpszPictureName, "Control2", "ValueColumnAdd", "myValueTable1");
SetPropWord(lpszPictureName, "Control2", "ValueColumnProvider", 1);
SetPropChar(lpszPictureName, "Control2", "ValueColumnTagName", "Process value archive\
\PDL_ZT_1");
SetPropWord(lpszPictureName, "Control2", "ValueColumnBackColor", RGB(255,255,255));
SetPropChar(lpszPictureName, "Control2", "ValueColumnTimeColumn", "myRefTimeColumn");
```

```
//add new ValueColumn and assign propertys
SetPropChar(lpszPictureName, "Control2", "ValueColumnAdd", "myValueTable2");
SetPropWord(lpszPictureName, "Control2", "ValueColumnProvider", 1);
SetPropChar(lpszPictureName, "Control2", "ValueColumnTagName", "Process value archive\
\PDL_ZT_2");
SetPropWord(lpszPictureName, "Control2", "ValueColumnBackColor", RGB(0,255,255));
SetPropChar(lpszPictureName, "Control2", "ValueColumnTimeColumn", "myRefTimeColumn");

//add new ValueColumn and assign propertys
SetPropChar(lpszPictureName, "Control2", "ValueColumnAdd", "myValueTable3");
SetPropWord(lpszPictureName, "Control2", "ValueColumnProvider", 1);
SetPropChar(lpszPictureName, "Control2", "ValueColumnTagName", "Process value archive\
\PDL_ZT_3");
SetPropWord(lpszPictureName, "Control2", "ValueColumnBackColor", RGB(255,255,0));
SetPropChar(lpszPictureName, "Control2", "ValueColumnTimeColumn", "myRefTimeColumn");
```

Ergebnis



	myRefTimeAxis	myValueTable1	myValueTable2	myValueTable3
112	08.02.2010 14:15:03	1	1	84
113	08.02.2010 14:15:03	22	1	84
114	08.02.2010 14:15:04	2	1	84
115	08.02.2010 14:15:04	1	1	84
116	08.02.2010 14:15:05	1	1	84
117	08.02.2010 14:15:05	3	1	84
118	08.02.2010 14:15:06	1	1	84
119	08.02.2010 14:15:06	1	1	84
120	08.02.2010 14:15:07	18	1	84

So fügen Sie Elemente zu einem WinCC OnlineTrendControl hinzu

Einleitung

Im folgenden Beispiel fügen Sie die Elemente Kurvenfenster, Wertachse, Zeitachse und Kurven in ein leeres WinCC OnlineTrendControl ein.

Voraussetzung

- Im Editor "Tag Logging" ist ein Archiv mit drei Archivvariablen angelegt.
- Im Graphics Designer ist ein "WinCC OnlineTrendControl" mit dem Namen "Control2" in das Prozessbild eingefügt.
- Im Graphics Designer ist ein Button eingefügt. Sie haben für den Button z. B. das Ereignis "Mausklick" projektiert mit einer C-Aktion und folgendem Skript.

Beispiel

```
//create reference to new window, time and value axis
SetPropChar(lpszPictureName, "Control2", "TrendWindowAdd", "myWindow");
SetPropChar(lpszPictureName, "Control2", "TimeAxisAdd", "myTimeAxis");
SetPropChar(lpszPictureName, "Control2", "ValueAxisAdd", "myValueAxis");

//assign time and value axis to the window
SetPropChar(lpszPictureName, "Control2", "TimeAxisTrendWindow", "myWindow");
SetPropChar(lpszPictureName, "Control2", "ValueAxisTrendWindow", "myWindow");

//add new trend and assign propertys
SetPropChar(lpszPictureName, "Control2", "TrendAdd", "myTrend1");
SetPropWord(lpszPictureName, "Control2", "TrendProvider", 1);
SetPropChar(lpszPictureName, "Control2", "TrendTagName", "Process value archive\
\PDL_ZT_1");
SetPropWord(lpszPictureName, "Control2", "TrendColor",  RGB(255,0,0));
SetPropChar(lpszPictureName, "Control2", "TrendTrendWindow", "myWindow");
SetPropChar(lpszPictureName, "Control2", "TrendTimeAxis", "myTimeAxis");
SetPropChar(lpszPictureName, "Control2", "TrendValueAxis", "myValueAxis");

//add new trend and assign propertys
SetPropChar(lpszPictureName, "Control2", "TrendAdd", "myTrend2");
SetPropWord(lpszPictureName, "Control2", "TrendProvider", 1);
SetPropChar(lpszPictureName, "Control2", "TrendTagName", "Process value archive\
\PDL_ZT_2");
SetPropWord(lpszPictureName, "Control2", "TrendColor", RGB(0,255,0));
SetPropChar(lpszPictureName, "Control2", "TrendTrendWindow", "myWindow");
SetPropChar(lpszPictureName, "Control2", "TrendTimeAxis", "myTimeAxis");
SetPropChar(lpszPictureName, "Control2", "TrendValueAxis", "myValueAxis");

//add new trend and assign propertys
SetPropChar(lpszPictureName, "Control2", "TrendAdd", "myTrend3");
SetPropWord(lpszPictureName, "Control2", "TrendProvider", 1);
SetPropChar(lpszPictureName, "Control2", "TrendTagName", "Process value archive\
\PDL_ZT_3");
SetPropWord(lpszPictureName, "Control2", "TrendColor", RGB(0,0,255));
SetPropChar(lpszPictureName, "Control2", "TrendTrendWindow", "myWindow");
SetPropChar(lpszPictureName, "Control2", "TrendTimeAxis", "myTimeAxis");
SetPropChar(lpszPictureName, "Control2", "TrendValueAxis", "myValueAxis");
```

2.15.5 Auflistungen

2.15.5.1 Balkenrichtung

Balkenrichtung	Zahlenwert
oben	0
unten	1
links	2
rechts	3

2.15.5.2 Balkenskalierung

Zahlenwert	Balkenskalierung
0	Linear (gleiche Gewichtung)
1	Logarithmisch (niedrige Werte betont)
2	Negativ Logarithmisch (hohe Werte betont)
3	Automatisch (Linear)
4	Tangens (hohe und niedrige Werte betont)
5	Quadratisch (hohe Werte betont)
6	Kubisch (hohe Werte stark betont)

2.15.5.3 Blinkfrequenzen

Zahlenwert	Blinkfrequenz
0	0,5 Hz
1	2 Hz
2	8 Hz

Hinweis

Da es sich beim Blinken um eine softwaretechnische Realisierung handelt, ist die Blinkfrequenz system- und hardwareabhängig (Anzahl der Objekte, Prozessor, Speicher, Aktualisierungszeit usw.).

Die Angaben in der Tabelle sind deshalb nur als Orientierungswerte anzusehen.

2.15.5.4 E/A-Feld, Ausgabeformat

Wie Zahlenwerte bei der Ausgabe in ein E/A-Feld dargestellt werden, wird durch eine Formatangabe gesteuert.

Eine Formatangabe setzt sich aus einem oder mehreren Formatierungszeichen zusammen. Die gültigen Formatierungszeichen und ihre Bedeutung sind in der folgenden Tabelle zusammengestellt:

Zeichen	Bedeutung	Bemerkung
s	Positive Zahlen werden mit Vorzeichen ausgegeben	Steht immer an erster Stelle der Formatangabe Darf in der Formatangabe nur einmal vorkommen
0(NULL)	Führende und endende Nullen werden ausgegeben	Steht immer nach s Fehlt s, dann steht es immer an erster Stelle Darf in der Formatangabe nur einmal vorkommen
9	Gibt die Position einer Ziffer in der auszugebenden Zahl an	Kann in der Formatangabe beliebig oft vorkommen.
, (Komma)	Position des Dezimalzeichens	
e	Gibt die Zahl in Exponentialdarstellung aus	Steht immer an letzter Stelle der Formatangabe

Beispiele:

Zahl	Format	Darstellung
123,455	999,999	123,455
123,455	999,99	123,46
123,455	9999,9999	123,4550
123,455	s09999,9999	+0123,4550
123,455	9,99999e	1,23455e+002

Wird in der Formatangabe das Dezimalzeichen weggelassen, dann werden die Stellen nach dem Komma nicht angezeigt und die Zahl wird auf eine ganze Zahl gerundet.

Werden in der Formatangabe weniger Stellen nach dem Komma vorgesehen, als die Zahl tatsächlich hat, dann werden nur so viele Nachkommastellen ausgegeben, wie in der Formatangabe festgelegt sind.

Die Zahl wird entsprechend gerundet.

Hat die Zahl mehr Stellen vor dem Komma, als in der Formatangabe festgelegt ist, dann werden drei Sternchen (***) ausgegeben, zum Zeichen dafür, dass die Zahl in diesem Format nicht dargestellt werden kann.

2.15.5.5 E/A-Feld, Datentyp des Feldinhalts

Datentyp	Zahlenwert
binär	0
dezimal	1
string	2
hexadezimal	3

2.15.5.6 E/A-Feld, Feldtyp

Typ	Zahlenwert
Ausgabe	0
Eingabe	1
Aus- und Eingabe	2

2.15.5.7 Elementausrichtung in Check- und Radioboxen

Ausrichtung	Zahlenwert
links	0
rechts	-1

2.15.5.8 Farbtabelle

Die 16 Grundfarben sind:

Farbe	Farbwert(Hex)	symbolische Konstante
Rot	0x000000FF	CO_RED
Dunkelrot	0x00000080	CO_DKRED
Grün	0x0000FF00	CO_GREEN
Dunkelgrün	0x00008000	CO_DKGREEN
Blau	0x00FF0000	CO_BLUE
Dunkelblau	0x00800000	CO_DKBLUE

Farbe	Farbwert(Hex)	symbolische Konstante
Cyan	0x00FFFF00	CO_CYAN
Dunkelcyan	0x00808000	CO_DKCYAN
Gelb	0x0000FFFF	CO_YELLOW
Dunkelgelb	0x00008080	CO_DKYELLOW
Magenta	0x00FF00FF	CO_MAGENTA
Dunkelmagenta	0x00800080	CO_DKMAGENTA
Hellgrau	0x00C0C0C0	CO_LTGRAY
Grau	0x00808080	CO_DKGRAY
Schwarz	0x00000000	CO_BLACK
Weiß	0x00FFFFFF	CO_WHITE

Hinweis

Die symbolischen Konstanten sind durch #define extern vordefiniert und werden von WinCC zur Verfügung gestellt.

2.15.5.9 Formatbeschreiber

Bei den Formatbeschreibern wird folgender Typ erwartet:

%d = DWORD / Int

%f = double

%s = char*

Weiterhin ist es z. B. möglich, eine Textvariable mit %d zu lesen, wenn sicher gestellt ist, dass der Wert in einem DWORD abgebildet werden kann.

Bei der folgenden Versorgung ist sichergestellt, dass der Wert abgebildet werden kann:

Variable	Format	C-Variable
Bit	%d	DWORD / long int signed
Byte	%d	DWORD / long int signed
SByte	%d	DWORD / long int signed
Word	%d	DWORD / long int signed
SWord	%d	DWORD / long int signed
DWord	%d	DWORD / long int signed

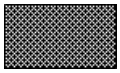
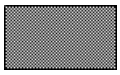
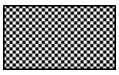
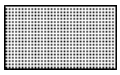
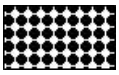
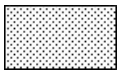
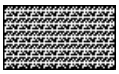
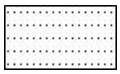
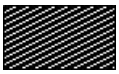
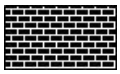
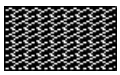
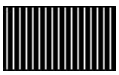
Variable	Format	C-Variable
SDWord	%d	DWORD / long int signed
Float	%f	double
Double	%f	double
Char	%s	char*

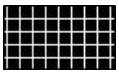
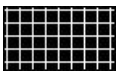
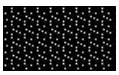
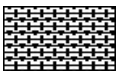
Hinweis

Wenn ein "DWORD", bei dem das 32. Bit gesetzt ist, gelesen werden soll, muss ein Formatbeschreiber für vorzeichenlose Ganzzahlen (%u) verwendet werden.

2.15.5.10 Füllmuster

Füllmuster	Wert
Transparent	65536
Massiv	0

Füllmuster	Wert	Füllmuster	Wert	Füllmuster	Wert
	1048576		196611		196627
	1048577		196612		196628
	1048578		196613		196629
	1048579		196614		196630
	1048832		196615		196631
	1048833		196616		196632
	1048834		196617		196633
	1048835		196618		196634
	131072		196619		196635

Füllmuster	Wert	Füllmuster	Wert	Füllmuster	Wert
	131073		196620		196636
	131074		196621		196637
	131075		196622		196638
	131076		196623		196639
	196608		196624		196640
	196609		196625		196641
	196610		196626		196642

Hinweis

Das Füllmuster "Massiv" füllt das Objekt mit der eingestellten Hintergrundfarbe aus.

2.15.5.11 Linienarten

Linienart	symbolische Bezeichnug	Wert
	LS_SOLID	0
	LS_DASH	1
	LS_DOT	2
	LS_DASHDOT	3
	LS_DASHDOT DOT	4
verborgen	LS_INVISIBLE	5

2.15.5.12 Linienenden

Linienende	symbolische Bezeichnung	Wert für die linken Linienenden	Wert für die rechten Linienenden
	LE_NO	0	0
	LE_HOLLOW_ARROW	1	65536
	LE_FULL_ARROW	2	131072
	LE_CFULL_ARROW	3	196608
	LE_LINE	4	262144
	LE_HOLLOW_CIRCLE	5	327680
	LE_FULL_CIRCLE	6	393216

Hinweis

Ab einer Linienbreite > 5 wird das Linienende "leerer Kreis" als ausgefüllter Kreis dargestellt.

2.15.5.13 Listenarten

Listenart	Zahlenwert
dezimal	0
binär	1
bit	2

2.15.5.14 Sprachkennungen

Von WinCC werden nur die SUBLANG_DEFAULT-Sprachen von Windows unterstützt.

symbolische Bezeichnung	Wert(hexadezimal)	Kürzel
LANG_ARABIC	0x0401	
LANG_AFRIKAANS	0x0436	
LANG_ALBANIAN	0x041C	
LANG_BASQUE	0x042D	
LANG_BULGARIAN	0x0402	
LANG_BYELORUSSIAN	0x0423	
LANG_CATALAN	0x0403	

symbolische Bezeichnung	Wert(hexadezimal)	Kürzel
LANG_CHINESE	0x0404	
LANG_CROATIAN	0x041A	
LANG_CZECH	0x0405	CSY
LANG_DANISH	0x0406	DAN
LANG_DUTCH	0x0413	NLD
LANG_ENGLISH	0x0409	ENU
LANG_ESTONIAN	0x0425	
LANG_FAEROESE	0x0438	
LANG_FARSI	0x0429	
LANG_FINNISH	0x040B	FIN
LANG_FRENCH	0x040C	FRA
LANG_GERMAN	0x0407	DEU
LANG_GREEK	0x0408	
LANG_HEBREW	0x040D	
LANG_HUNGARIAN	0x040E	HUN
LANG_ICELANDIC	0x040F	ISL
LANG_INDONESIAN	0x0421	
LANG_ITALIAN	0x0410	ITA
LANG_JAPANESE	0x0411	
LANG_KOREAN	0x0412	
LANG_LATVIAN	0x0426	
LANG_LITHUANIAN	0x0427	
LANG_NORWEGIAN	0x0414	NOR
LANG_POLISH	0x0415	PLK
LANG_PORTUGUESE	0x0416	PTB
LANG_ROMANIAN	0x0418	
LANG_RUSSIAN	0x0419	RUS
LANG_SLOVAK	0x041B	SKY
LANG_SLOVENIAN	0x0424	
LANG_SORBIAN	0x042E	
LANG_SPANISH	0x040A	ESP
LANG_SWEDISH	0x041D	SVE
LANG_THAI	0x041E	
LANG_TURKISH	0x041F	TRK
LANG_UKRAINIAN	0x0422	

2.15.5.15 Textausrichtung

Horizontal	Ausrichtung	Zahlenwert
	links	0
	zentriert	1
	rechts	2

Vertikal	Ausrichtung	Zahlenwert
	oben	0
	zentriert	1
	unten	2

2.15.5.16 Variablenstati

Wert (dezimal)	Wert (hexdezimal)	Bedeutung
0	0x0000	Kein Fehler
1	0x0001	Verbindung zum Partner nicht aufgebaut
2	0x0002	Protokollfehler
4	0x0004	Netzwerkbaugruppe defekt
8	0x0008	Projektierte Obergrenze überschritten
16	0x0010	Projektierte Untergrenze unterschritten
32	0x0020	Formatgrenze überschritten
64	0x0040	Formatgrenze unterschritten
128	0x0080	Wandlungsfehler
256	0x0100	Initialisierungswert der Variablen
512	0x0200	Ersatzwert der Variablen
1024	0x0400	Adressierungsfehler im Kanal
2048	0x0800	Variable nicht gefunden oder nicht vorhanden
4096	0x1000	Zugriff auf Variable nicht erlaubt
8192	0x2000	Timeout, keine Rückmeldung vom Kanal
16384	0x4000	Server nicht verfügbar

2.15.6 Strukturdefinitionen

2.15.6.1 Strukturdefinition CCAPErrorExecute

```
typedef struct {
DWORD dwCurrentThreadID; Thread ID of the current thread
DWORD dwErrorCode1;      Error code 1
DWORD dwErrorCode2;      Error code 2
BOOL bCycle;             cycle/acycle
char* szApplicationName; Name of the application
char* szFunctionName;    Name of the function
char* szTagName;         Name of the tag
LPVOID lpParam;          Pointer to the action stack
DWORD dwParamSize;       Size of the action stack
DWORD dwCycle;           Cycle of the variable
CMN_ERROR* pError;       Pointer to CMN_ERROR
} CCAPErrorExecute;
```

Members

Die Bedeutung der einzelnen Fehlerkennungen und die abhängig davon übergebenen Strukturelemente sind in der folgenden Tabelle angegeben:

dwErrorCode1	dwErrorCode2	bCycle	szApplicationName	szFunctionName	szTagName	lpParamSize	dwParamSize	dwCycle	pError	Beschreibung
1007001	0	x	x	x		x	x			Exception in der Aktion
1007001	1	x	x	x		x	x			Exception beim Zugriff auf das Rückgabe-ergebnis
1007001	4097	x	x	x		x	x			Stapelüberlauf beim Ausführen der Aktion
1007001	4098	x	x	x		x	x			Division durch 0 in der Aktion
1007001	4099	x	x	x		x	x			Zugriff auf ein nicht vorhandenes Symbol in der Aktion
1007001	4100	x	x	x		x	x			Access violation in der Aktion
1007004	0	x	x	x						Funktion ist nicht bekannt

1007005	1	x	x							Aktion enthält keinen P-Code
1007005	2	x	x							Funktionsname ist fehlerhaft
1007005	4	x	x	x		x	x			Typ des Rückgabewertes ist ungültig
1007005	32768ff	x	x	x		x	x			Fehler im Ciss Compiler beim Laden der Aktion
1007006	0	x	x	x	x	x	x	x		Variable ist nicht definiert
1007006	1	x	x	x	x	x	x	x		Variable timeout
1007006	2	x	x	x	x	x	x	x	x	Variable kann im gewünschten Format nicht geliefert werden
1007006	3	x	x	x	x	x	x	x	x	Variable liefert Statusverletzung, in CMN_ERROR.dwError1 steht der Status
1007007	1	x	x	x		x	x		x	Fehler bei PDLRTGetProp
1007007	2	x	x	x		x	x		x	Fehler bei PDLRTSetProp
1007007	3	x	x	x		x	x		x	Fehler bei DM-Aufruf

Fehlerstruktur

Die Funktion OnErrorExecute nutzt die Fehlerstruktur zur Auswertung bzw. zur Ausgabe von Fehlermeldungen, wenn dies in der Spalte pError mit einem "x" gekennzeichnet ist.

Siehe auch

Strukturdefinition CMN_ERROR (Seite 1604)

2.15.6.2 Strukturdefinition CCAPTime

```
typedef struct {
DWORD dwCurrentThreadID; ThreadID of the current Thread
DWORD dwCode;            Code
BOOL bCycle;             cycle/acycle
char* szApplicationName; Name of the Application
char* szFunctionName;    Name of the Function
LPVOID lpParam;          Pointer to the Action-Stack
DWORD dwParamSize;       size of the Action-Stack
double dblTime;
DWORD dwFlags;           flags
} CCAPTime;
```

Members

dwCode

Das Strukturelement dwCode gibt Auskunft über den Aufruf von OnTime:

dwCode = 113	Aufruf mit Zeitbestimmung für jede Aktion
dwCode = 114	Aufruf mit Zeitüberwachung für eine Aktion

dwFlags

Das Strukturelement dwFlags gibt Auskunft über die Art der Ausgabe:

dwFlags = TRUE	Die Ausgabe der Ergebnisse erfolgt in eine Datei
dwFlags = FALSE	Die Ausgabe der Ergebnisse erfolgt in das Diagnosefenster

2.15.6.3 Strukturdefinition CMN_ERROR

```
struct CMNERRORSTRUCT {
DWORD       dwError1,
DWORD       dwError2,
DWORD       dwError3,
DWORD       dwError4,
DWORD       dwError5;
TCHAR       szErrorText[MAX_ERROR_LEN];
}
CMN_ERROR
```

Beschreibung

In der erweiterten Fehlerstruktur befinden sich der Fehlercode und ein Fehlertext für den aufgetretenen Fehler. Jede Applikation kann die Fehlerstruktur zur Auswertung bzw. zur Ausgabe von Fehlermeldungen nutzen.

Members

dwError1 .. dwError5

Diese Einträge können durch die API-Funktionen beliebig verwendet werden.

In den API-Beschreibungen ist jeweils beschrieben, welche Werte die Einträge im Fehlerfall jeweils enthalten. Soweit nichts anderes angegeben ist, stehen die Fehlercodes in dwError1.

szErrorText

Puffer für textuelle Beschreibung der Fehlerursache

Der Inhalt wird aus den Resourcen ermittelt und ist deshalb sprachabhängig.

2.15.6.4 Strukturdefinition DM_TYPEREF

```
typedef struct {
DWORD dwType;
DWORD dwSize;
char szTypeName[MAX_DM_TYPE_NAME + 1];
}
DM_TYPEREF;
```

Members

dwType

Spezifiziert den Typ der Variablen

DM_VARTYPE_BIT	Binäre Variable
DM_VARTYPE_SBYTE	Vorzeichenbehafteter 8-Bit Wert
DM_VARTYPE_BYTE	Vorzeichenloser 8-Bit Wert
DM_VARTYPE_SWORD	Vorzeichenbehafteter 16-Bit Wert
DM_VARTYPE_WORD	Vorzichenloser 16-Bit Wert
DM_VARTYPE_SDWORD	Vorzeichenbehafteter 32-Bit Wert
DM_VARTYPE_DWORD	Vorzeichenloser 32-Bit Wert
DM_VARTYPE_FLOAT	Gleitkommazahl 32-Bit IEEE 754
DM_VARTYPE_DOUBLE	Gleitkommazahl 64-Bit IEEE 754
DM_VARTYPE_TEXT_8	Textvariable 8-Bit Zeichensatz
DM_VARTYPE_TEXT_16	Textvariable 16-Bit Zeichensatz
DM_VARTYPE_RAW	Rohdatentyp
DM_VARTYPE_STRUCT	Strukturvariable
DM_VARTYPE_TEXTREF	Text-Referenz-Variable

dwSize

Gibt die Länge des Datentyps in Byte an.

szTypeName

Enthält bei Strukturvariablen den Namen des Strukturtyps

2.15.6.5 Strukturdefinition DM_VAR_UPDATE_STRUCT

```
typedef struct {
DM_TYPEREF dmTypeRef;
DM_VARKEY dmVarKey;
VARIANT dmValue;
DWORD dwState;
}
DM_VAR_UPDATE_STRUCT;
```

Members

dmTypeRef

Enthält Angaben zum Variablentyp. Bei zyklischen Anforderungen wird in diese Struktur aus Performancegründen nichts eingetragen.

dmVarKey

Spezifiziert die zu bearbeitenden Variablen.

dmValue

Wert der Variablen.

Bei Zugriffen auf den Wert des VARIANTS muss zwischen dem Namen des VARIANTS und dem Namen des Members ein ".u." eingefügt werden.

```
Beispiel
// Variant versorgen
myVariant.vt = VT_I4;
myVariant.u.lVal = 233;
```

Die Beschreibung des Datentyps VARIANT finden Sie in der einschlägigen Fachliteratur. Der VARIANT dmValue muss vor Erstbenutzung mit VariantInit() initialisiert und nach Benutzung mit VariantClear(&dmValue) wieder freigegeben werden. Die Struktur DM_VAR_UPDATE_STRUCT darf aus diesem Grund nicht mit ZeroMemory() oder memset() gelöscht werden.

dwState

Kennzeichnet den Status der Variablen.

Siehe auch

Variablenstati (Seite 1601)

Strukturdefinition DM_VARKEY (Seite 1608)

Strukturdefinition DM_TYPEREF (Seite 1605)

2.15.6.6 Strukturdefinition DM_VAR_UPDATE_STRUCTEX

```
typedef struct {
DM_TYPEREF dmTypeRef;
DM_VARKEY dmVarKey;
VARIANT dmValue;
DWORD dwState;
DWORD dwQualityCode;
}
DM_VAR_UPDATE_STRUCTEX;
```

Members

dmTypeRef

Enthält Angaben zum Variablentyp. Bei zyklischen Anforderungen wird in diese Struktur aus Performancegründen nichts eingetragen.

dmVarKey

Spezifiziert die zu bearbeitenden Variablen.

dmValue

Wert der Variablen.

Bei Zugriffen auf den Wert des VARIANTS muss zwischen dem Namen des VARIANTS und dem Namen des Members ein ".u." eingefügt werden.

```
Beispiel
// Variant versorgen
myVariant.vt = VT_I4;
myVariant.u.lVal = 233;
```

Die Beschreibung des Datentyps VARIANT finden Sie in der einschlägigen Fachliteratur. Der VARIANT dmValue muss vor Erstbenutzung mit VariantInit() initialisiert und nach Benutzung mit VariantClear(&dmValue) wieder freigegeben werden. Die Struktur DM_VAR_UPDATE_STRUCTEX darf aus diesem Grund nicht mit ZeroMemory() oder memset() gelöscht werden.

dwState

Kennzeichnet den Status der Variablen.

dwQualityCode

Kennzeichnet den QualityCode der Variablen.

Siehe auch

Variablenstati (Seite 1601)

Strukturdefinition DM_VARKEY (Seite 1608)

Strukturdefinition DM_TYPEREF (Seite 1605)

2.15.6.7 Strukturdefinition DM_VARKEY

```
typedef struct {
DWORD dwKeyType;
DWORD dwID;
char szName[ MAX_DM_VAR_NAME + 1 ];
LPVOID lpvUserData;
}
DM_VARKEY;
```

Members

dwKeyType

Gibt an, ob die Variable über eine Schlüssel-ID oder über ihren Namen angesprochen werden soll:

DM_VARKEY_ID Spezifizierung über Schlüssel ID

DM_VARKEY_NAME Spezifizierung über Variablennamen

dwID

Enthält die Schlüssel-ID der Variablen, wenn dwKeyType entsprechend gesetzt ist

szName

Enthält den Namen der Variablen, wenn dwKeyType entsprechend gesetzt ist.

lpvUserData

Zeiger auf applikationsspezifische Daten

2.15.6.8 Strukturdefinition LINKINFO

```
typedef struct {
LINKTYPE LinkType;
DWORD dwCycle;
TCHAR szLinkName[256];
}
LINKINFO;
```

Members

LinkType

LinkType sind Enumerationskonstanten, die in der Datei "Trigger.h" definiert sind. Diese sind mit dem Befehl #include "Trigger.h" und der entsprechenden Enumerationskonstanten in Ihr Script einzubinden.

BUBRT_LT_NOLINK	0	keine Verknüpfung
BUBRT_LT_VARIABLE_DIRECT	1	direkte Variable
BUBRT_LT_VARIABLE_INDIRECT	2	indirekte Variable
BUBRT_LT_ACTION	3	C-Aktion
BUBRT_LT_ACTION_WIZARD	4	Dynamic Dialog
BUB_LT_DIRECT_CONNECTION	5	Direktverbindung
BUBRT_LT_ACTION_WIZARD_INPROC	6	Dynamic Dialog

Bei der Funktion SetLink dürfen nur die Enumerarionskonstanten BUBRT_LT_VARIABLE_DIREKT und BUBRT_LT_VARIABLE_INDIRECT verwendet werden. Bei der Funktion GetLink können alle aufgeführten Enumerationskonstanten zurückgeliefert werden.

dwCycle

Zykluszeit der Aktualisierung

dwCycle	Aktualisierungszyklus
255	Bildzyklus
235	Fensterzyklus
0	Bei Änderung
1	250ms
2	500ms
3	1s
4	2s
5	5s
6	10s
7	1min
8	5min
9	10min
10	1h
11-15	Anwenderzyklus 1-5

szLinkName

Variablenname

2.15.6.9 Strukturdefinition MSG_FILTER_STRUCT

```
typedef struct {
CHAR        szFilterName[MSG_MAX_TEXTLEN+1];
WORD        dwFilter;
SYSTEMTIME  st[2];
DWORD       dwMsgNr[2];
DWORD       dwMsgClass;
DWORD       dwMsgType[MSG_MAX_CLASS];
DWORD       dwMsgState;
WORD        wAGNr[2];
WORD        wAGSubNr[2];
DWORD       dwArchivMode;
char        szTB[MSG_MAX_TB][
MSG_MAX_TB_CONTENT+1]
DWORD       dwTB;
Double      dPValue[MSG_MAX_PVALUE][2];
DWORD       dwPValue[2];
DWORD       dwMsgCounter[2];
DWORD       dwQuickSelect;
}
MSG_FILTER_STRUCT;
```

Beschreibung

In dieser Struktur werden die Filterkriterien angegeben.

Members

dwFilter

Die Filterbedingungen werden über die folgenden Konstanten aus der Datei "m_global.h" festgelegt:

MSG_FILTER_DATE_FROM	Datum von
MSG_FILTER_DATE_TO	Datum bis
MSG_FILTER_TIME_FROM	Uhrzeit von
MSG_FILTER_TIME_TO	Uhrzeit bis
MSG_FILTER_NR_FROM	Meldungsnummer von
MSG_FILTER_NR_TO	Meldungsnummer bis
MSG_FILTER_CLASS	Meldungsklassen
MSG_FILTER_STATE	Meldungsstatus
MSG_FILTER_AG_FROM	AG Nummer von
MSG_FILTER_AG_TO	AG Nummer bis
MSG_FILTER_AGSUB_FROM	AG Subnummer von
MSG_FILTER_AGSUB_TO	AG Subnummer bis
MSG_FILTER_TEXT	Meldungstexte
MSG_FILTER_PVALUE	Prozesswerte

MSG_FILTER_COUNTER_FROM	Interner Meldungszähler von
MSG_FILTER_COUNTER_TO	Interner Meldungszähler bis
MSG_FILTER_PROCESSMSG	Prozessmeldungen
MSG_FILTER_SYSMSG	Systemmeldungen
MSG_FILTER_BEDMSG	Bedienmeldungen
MSG_FILTER_DATE	Datum von bis
MSG_FILTER_TIME	Uhrzeit von bis
MSG_FILTER_NR	Meldungsnummer von bis
MSG_FILTER_VISIBLEONLY	Sichtbare Meldungen anzeigen
MSG_FILTER_HIDDENONLY	Ausgeblendete Meldungen anzeigen

st

Datum / Uhrzeit von - bis

Dabei ist st[0] Beginnzeitpunkt (von), st[1] Endzeitpunkt (bis)

Belegen Sie diese Felder für die Filterkriterien: MSG_FILTER_DATE, MSG_FILTER_DATE_FROM, MSG_FILTER_DATE_TO, MSG_FILTER_TIME, MSG_FILTER_TIME_FROM, bzw. MSG_FILTER_TIME_TO

Wird eine aktuelle Zeit für die Übergabe eines SYSTEMTIME-Parameters benötigt, so ist die Funktion GetLocalTime zu verwenden und nicht GetSystemTime. Es besteht in der Regel ein beträchtlicher Zeitunterschied zwischen diesen beiden Funktionen.

dwMsgNr

Meldungsnummer von - bis

Dabei ist dwMsgNr[0] Anfangsnr. (von), dwMsgNr[1] Endnr. (bis)

Belegen Sie diese Felder für die Filterkriterien: MSG_FILTER_NR, MSG_FILTER_NR_FROM bzw. MSG_FILTER_NR_TO

dwMsgClass

Meldungsklassen bitcodiert

Belegen Sie dieses Feld für das Filterkriterium: MSG_FILTER_CLASS

dwMsgType

Meldungsart je Meldeklasse, bitcodiert

Belegen Sie dieses Feld für das Filterkriterium: MSG_FILTER_CLASS

dwMsgState

Meldungsstatus bitcodiert

Belegen Sie dieses Feld für das Filterkriterium: MSG_FILTER_STATE

wAGNr

AGNr von - bis

Belegen Sie diese Felder für die Filterkriterien: MSG_FILTER_AG_FROM bzw. MSG_FILTER_AG_TO

wAGSubNr

AGSubNr von - bis

Belegen Sie dieses Feld für die Filterkriterien: MSG_FILTER_AGSUB_FROM bzw. MSG_FILTER_AGSUB_TO

dwArchivMode

Archivierung/ Protokollierung

Muss mit 0 belegt werden.

szTB

Texte der Textblöcke

Belegen Sie diese Felder für das Filterkriterium: MSG_FILTER_TEXT

dwTB

Textblöcke aktiv (von - bis, bitcodiert)

Belegen Sie dieses Feld für das Filterkriterium: MSG_FILTER_TEXT

dPValue

Prozesswerte von - bis

Belegen Sie diese Felder für das Filterkriterium: MSG_FILTER_PVALUE

dwPValue

Prozesswerte aktiv (von - bis bitcodiert)

Belegen Sie dieses Feld für das Filterkriterium: MSG_FILTER_PVALUE

dwMsgCounter

Interner Meldungszähler von - bis

Belegen Sie diese Felder für die Filterkriterien: MSG_FILTER_COUNTER_FROM, MSG_FILTER_COUNTER_TO

dwQuickSelect

Schnellselektion für Stunde, Tag, Monat

Der Parameter ist für spätere Erweiterungen reserviert und muss mit 0 vorbesetzt werden.

Belegen Sie dieses Feld für das Filterkriterium: MSG_FILTER_QUICKSELECT

LOWORD Typ:

MSG_FILTER_QUICK_MONTH	Schnellselektion letzte n Monate
MSG_FILTER_QUICK_DAYS	Schnellselektion letzte n Tage
MSG_FILTER_QUICK_HOUR	Schnellselektion letzte n Stunden

HIWORD Anzahl: 1...n

Der Endzeitpunkt der Schnellselektion bezieht sich auf die aktuelle Systemzeit des lokalen Rechners. Der Startzeitpunkt wird n * (Monate, Tage, Stunden zurückgerechnet).

2.15.6.10 Strukturdefinition MSG_RTDATA_STRUCT

```
typedef struct {
DWORD                dwMsgState;
DWORD                dwMsgNr;
SYSTEMTIME           stMsgTime;
DWORD                dwTimeDiff;
DWORD                dwCounter;
DWORD                dwFlags;
WORD                 wPValueUsed;
WORD                 wTextValueUsed;
double               dPValue[MSG_MAX_PVALUE];
MSG_TEXTVAL_STRUCT   mtTextValue[MSG_MAX_PVALUE];
}
MSG_RTDATA_STRUCT;
```

Members

dwMsgState

Zustand einer Meldung

MSG_STATE_COME	0x00000001	Meldung gekommen
MSG_STATE_GO	0x00000002	Meldung gegangen
MSG_STATE_QUIT	0x00000003	Meldung quittiert
MSG_STATE_LOCK	0x00000004	Meldung gesperrt
MSG_STATE_UNLOCK	0x00000005	Meldung Freigabe
MSG_STATE_QUIT_SYSTEM	0x00000010	Meldung quittiert durch System
MSG_STATE_QUIT_EMERGENCY	0x00000011	Not - Quittierung
MSG_STATE_QUIT_HORN	0x00000012	Hupenquittierung
MSG_STATE_COMEGO	0x00000013	Meldung gekommen und gegangen, nur Anzeige in Meldeliste
MSG_STATE_UPDATE	0x00010000	Bit für Meldungsupdate
MSG_STATE_RESET	0x00020000	Bit für Meldungsreset
MSG_STATE_SUMTIME	0x00040000	Bit für Sommerzeit aktiv
MSG_STATE_INSTANCE	0x00080000	Bit für Instanzmeldung (n Meldungen einer Nr.)

dwMsgNr

Meldungsnummer

stMsgTime

Datum / Uhrzeit: Telegrammzeit in Abhängigkeit von der aufrufenden Funktion

dwTimeDiff

Zeitdauer kommend / Telegrammzeit in Sekunden

dwCounter

Interner Meldungszähler

dwFlags

Meldungsflags in der Datenbank

MSG_FLAG_SUMTIME	0x00000001	Sommerzeit aktiv
MSG_FLAG_COMMENT	0x00000002	Meldung hat Kommentar
MSG_FLAG_ARCHIV	0x00000004	Archivieren
MSG_FLAG_PROTOCOL	0x00000008	Protokollieren
MSG_FLAG_TEXTVALUES	0x00000010	Meldung hat Textbegleitwerte
MSG_FLAG_TIMEINVALID	0x00000020	Bit für ungültigen Datum/Uhrzeit-Stempel
MSG_FLAG_INSTANCE	0x00000040	Kennzeichnung von Instanzmeldungen (185269)

wPValueUsed

benutzte Prozesswerte, bitcodiert. Jedes Bit darf nur in einem der beiden Strukturelemente "wPValueUsed" oder "wTextValueUsed" gesetzt werden. Ein Begleitwert kann entweder Zahl oder Text sein.

wTextValueUsed

verwendete Textwerte, bitcodiert. Jedes Bit darf nur in einem der beiden Strukturelemente "wPValueUsed" oder "wTextValueUsed" gesetzt werden. Ein Begleitwert kann entweder Zahl oder Text sein.

VBA zur automatisierten Projektierung 3

3.1 Automatisierte Projektierung

Inhalt

Mit VBA können Sie die Projektierung im Graphics Designer automatisieren. Dies umfasst:

- Anpassung des Graphics Designer
- Bearbeitung von Bildern
- Bearbeitung von Objekten
- Dynamsierung mit VBA
- Zugriff auf Fremdapplikationen

Im Editor "Graphics Designer" steht Ihnen dazu ein VBA-Editor zur Verfügung.

Dieses Kapitel enthält

- eine kurze Einführung zum Einsatz von VBA in WinCC,
- Grundlagen zum Einsatz von VBA im Graphics Designer und
- die Referenz zum VBA-Objektmodell des Graphics Designer.

3.2 Einführung: VBA in WinCC einsetzen

3.2.1 Einführung: VBA in WinCC einsetzen

Einleitung

Im Graphics Designer steht Ihnen einen VBA-Editor zur Verfügung, mit dessen Hilfe Sie die Projektierung von Bildern automatisieren können. Der VBA-Editor ist identisch mit dem aus den Produkten der Microsoft Office Familie. Sie können Ihre VBA-Programmiererfahrung direkt nutzen.

Prinzip

Mit VBA erweitern Sie die Funktionalität des Graphics Designer und automatisieren die Projektierung. VBA können Sie im Graphics Designer unter anderem wie folgt einsetzen:

- Benutzerdefinierte Menüs und Symbolleisten anlegen
- Standard-, Smart- und Windows-Objekte anlegen und bearbeiten
- Eigenschaften von Bildern und Objekten dynamisieren
- Aktionsprojektierung bei Bildern und Objekten
- Auf Produkte zugreifen, die VBA unterstützen (z.B. Produkte der MS Office-Familie)

Die Beschreibung des VBA-Objektmodells für den Graphics Designer finden Sie in dieser Dokumentation unter "VBA-Referenz".

Siehe auch

VBA-Code im WinCC-Projekt organisieren (Seite 1617)

VBA-Referenz (Seite 1735)

VBA im Graphics Designer (Seite 1623)

VBA-Makros im Graphics Designer ausführen (Seite 1621)

Differenzierung: Einsatz von VBA (Seite 1616)

3.2.2 Differenzierung: Einsatz von VBA

Einleitung

VBA können Sie ausschließlich zur Projektierung und Funktionserweiterung im Graphics Designer einsetzen. Im Folgenden wird dargestellt, wo es z.B. bessere Möglichkeiten für effiziente Projektierung gibt oder wo VBA nicht eingesetzt werden kann.

VB- und C-Skripte

VB- und C-Skripte sind ausschließlich in Runtime aktiv und werden zur Dynamisierung von Bild- und Objekteigenschaften sowie in der Aktionsprojektierung eingesetzt.

Dynamic Wizards

Die Dynamic Wizards werden durch VBA nicht ersetzt. Mit VBA haben Sie jedoch die Möglichkeit, die Funktionalität der Dynamic Wizards komfortabel zu erweitern.

ODK

ODK sind Funktionsaufrufe, die den Zugriff auf alle Funktionalitäten von WinCC ermöglichen, sowohl im Configuration System als auch in Runtime. VBA zeichnet sich im Gegensatz zum ODK durch den einfachen objektorientierten Zugriff auf Objekte des Graphics Designer aus.

Siehe auch

VBA im Graphics Designer (Seite 1623)

VBA-Makros im Graphics Designer ausführen (Seite 1621)

VBA-Code im WinCC-Projekt organisieren (Seite 1617)

Einführung: VBA in WinCC einsetzen (Seite 1616)

3.2.3 VBA-Code im WinCC-Projekt organisieren

Einleitung

Den VBA-Code Ihres WinCC-Projektes organisieren Sie im VBA-Editor. Dort legen Sie fest, ob der VBA-Code nur in einem Bild, im ganzen Projekt oder in allen Projekten verfügbar sein soll. In Abhängigkeit davon, wo Sie den VBA-Code platzieren, spricht man von

- globalem VBA-Code,
- projektspezifischem VBA-Code oder
- bildspezifischem VBA-Code.

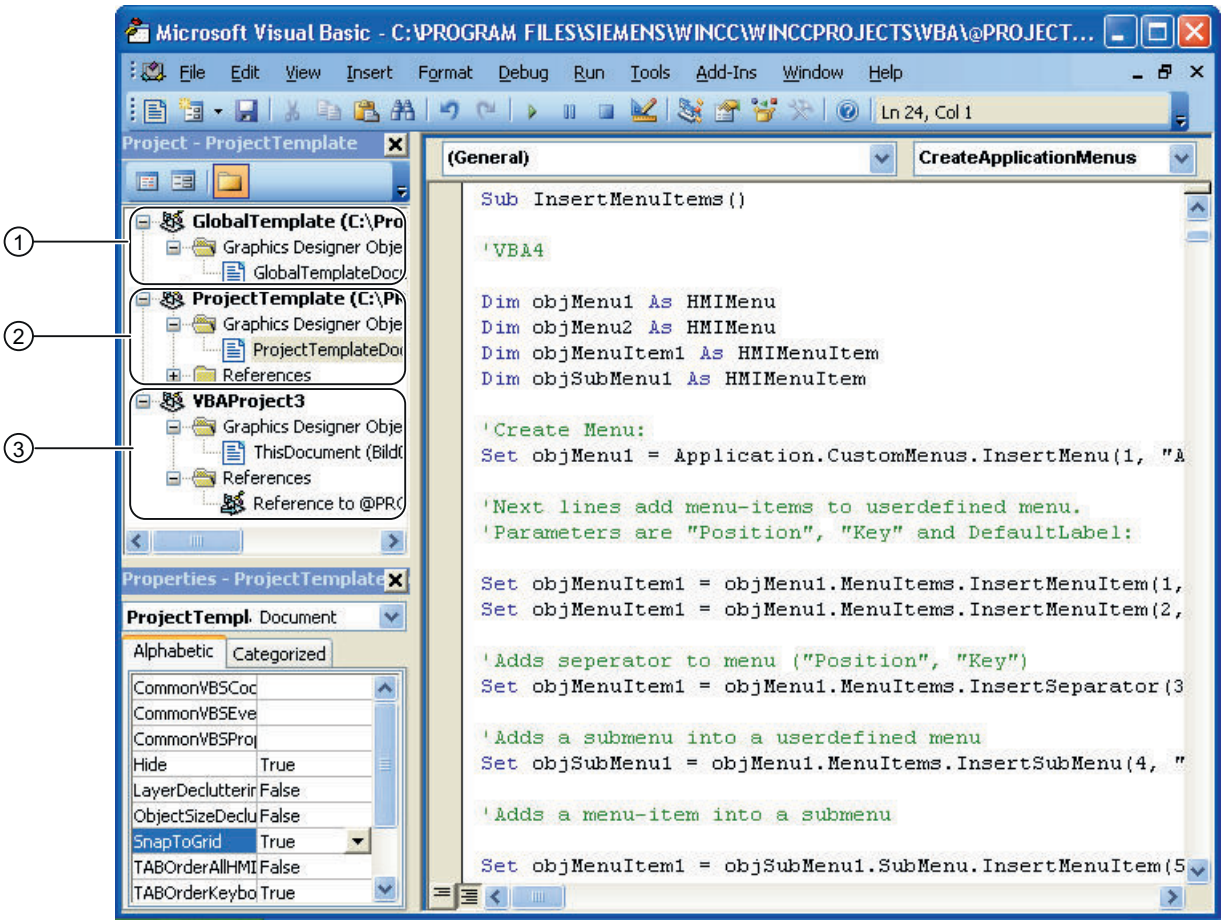
Hinweis

Ein Bild im Graphics Designer heißt im VBA-Objektmodell "Document".

Der VBA-Editor

Den VBA-Editor starten Sie im Graphics Designer mit <Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor". Wenn Sie noch kein Bild im Graphics Designer geöffnet haben, können Sie nur den globalen oder projektspezifischen VBA-Code editieren.

Im Project Explorer des VBA-Editors werden die globalen und projektspezifischen Daten sowie alle geöffneten Bilder angezeigt:



Globaler VBA-Code (1)

Bezeichnet VBA-Code, den Sie im VBA-Editor in das Dokument "GlobalTemplateDocument" schreiben. Dieser VBA-Code wird in der Datei "@GLOBAL.PDT" gespeichert, die sich im Installationsverzeichnis von WinCC befindet.

Platzieren Sie im "GlobalTemplateDocument" VBA-Code, der in allen WinCC-Projekten auf Ihrem Rechner verfügbar sein soll. Wenn Sie den VBA-Code auf einem anderen Rechner benötigen, verwenden Sie die Ex- und Importfunktionen des VBA-Editors.

Ein WinCC-Rechner benutzt ausschließlich die lokal im Installationverzeichnis (…\Siemens \WinCC\Templates) von WinCC abgelegte @GLOBAL.PDT.

Hinweis

Wenn Sie eine Update-Installation durchführen, wird Ihre globale Vorlage "@Global.pdt" in dem Backupfile "@Global.sav" gesichert. Das Backupfile wird im Verzeichnis …\Siemens \WinCC\Templates abgelegt. Ihr VBA-Code aus der alten globalen Vorlage wird nicht automatisch in die neue globale Vorlage übernommen.

Übernehmen des VBA-Codes aus der alten globalen Vorlage:

Um nach einer Update-Installation den VBA-Code aus der alten Vorlage zu übernehmen, gehen Sie folgendermaßen vor:

1. Wenn Sie bereits VBA-Code in die neue globale Vorlage eingetragen haben, öffnen Sie den VBA-Editor im Graphics Designer und kopieren Sie den VBA-Code.
2. Schließen Sie WinCC.
3. Öffnen Sie im Windows-Explorer das Verzeichnis …\Siemens\WinCC\Templates.
4. Löschen Sie die neue globale Vorlage "@Global.pdt".
5. Benennen Sie das Backupfile "@Global.sav" in "@Global.pdt" um.
6. Wenn Sie VBA-Code aus der neuen globalen Vorlage kopiert haben, öffnen Sie den VBA-Editor im Graphics-Designer und fügen Sie den kopierten VBA-Code ein.

Der VBA-Code aus Ihrer alten globalen Vorlage steht Ihnen wieder zur Verfügung.

Projektspezifischer VBA-Code (2)

Bezeichnet VBA-Code, den Sie im VBA-Editor in das Dokument "ProjectTemplateDocument" schreiben. Dieser VBA-Code wird in der Datei "@PROJECT.PDT" gespeichert, die sich im Stammverzeichnis jedes WinCC-Projektes befindet.

Die "@PROJECT.PDT"-Datei hat eine Referenz auf die "@GLOBAL.PDT"-Datei. Funktionen und Prozeduren, die Sie in der Datei "@GLOBAL.PDT" gespeichert haben, können Sie im "ProjectTemplateDocument" direkt aufrufen.

Platzieren Sie im "ProjectTemplateDocument" VBA-Code, den Sie in allen Bildern des geöffneten Projektes verwenden wollen. Wenn Sie den VBA-Code auf einem anderen Rechner benötigen, verwenden Sie die Ex- und Importfunktionen des VBA-Editors.

Die "@PROJECT.PDT"-Datei können Sie wie eine PDL-Datei öffnen und bearbeiten. Auf diese Weise können Sie die "@PROJECT.PDT"-Datei als Vorlagendatei verwenden: Sie können dort z.B. das Grundbild Ihrer Anlage anlegen, dass dann automatisch in jede neue PDL-Datei des Projektes übertragen wird. Bildeigenschaften wie Ebenen oder Zoom sowie VBA-Code werden nicht in die PDL-Datei übertragen.

Bildspezifischer VBA-Code (3)

Bezeichnet VBA-Code, den Sie im VBA-Editor in das Dokument "This Document" des entsprechenden Bildes schreiben. Dieser VBA-Code wird zusammen mit dem Bild als PDL-Datei gespeichert.

Die PDL-Datei hat eine Referenz auf die Datei "@PROJECT.PDT". Funktionen und Prozeduren, die Sie in der Datei "@PROJECT.PDT" gespeichert haben, können Sie aus der PDL-Datei direkt aufrufen. Sie haben jedoch keinen Zugriff auf Funktionen und Prozeduren, die in der Datei "@GLOBAL.PDT" gespeichert sind.

Hinweis

In jedem Dokument können Sie Modules, Class Modules und User Forms anlegen.

Sie können den VBA-Code eines Modules vor unberechtigtem Zugriff mit einem Passwort schützen. Wählen Sie dazu im VBA-Editor den Menübefehl "Tools" > "VBAObject Properties".

Besonderheiten bei der Ausführung von VBA-Makros

Bei der Ausführung von VBA-Makros gilt folgende Regelung: Es wird erst bild- und danach projektspezifischer VBA-Code ausgeführt. Wenn Sie also beispielsweise ein VBA-Makro aufrufen, das sowohl im Bild als auch im projektspezifischen VBA-Code enthalten ist, wird nur das VBA-Makro aus dem Bild ausgeführt. Auf diese Weise wird die doppelte Ausführung von VBA-Makros und Funktionen unterbunden, was sonst zu Fehlern führen kann.

Beim Event-Handling ist die Weiterleitung von Ereignissen standardmäßig aktiviert. Sie können die Weiterleitung unterbinden, wenn Sie nur im bildspezifischen VBA-Code auf ein Ereignis reagieren wollen.

Weitere Informationen zu diesem Thema finden Sie unter "Event-Handling".

Testen mit dem Debugger

Ihre VB-Skripte können Sie in Runtime mit dem Debugger des VBA-Editor testen. Weitere Informationen finden Sie in der Hilfe zum VBA-Editor.

Siehe auch

Event-Handling (Seite 1715)

VBA im Graphics Designer (Seite 1623)

VBA-Makros im Graphics Designer ausführen (Seite 1621)

So exportieren und importieren Sie VBA-Code (Seite 1620)

3.2.4 So exportieren und importieren Sie VBA-Code

Prinzip

Im VBA-Editor können Sie VBA-Code ex- und importieren, um ihn auf einen anderen Rechner zu übertragen. Referenzen auf Prozeduren und Funktionen, die Sie innerhalb des Projektes aufrufen, bleiben damit erhalten.

Hinweis

Referenzen auf externe Bibliotheken müssen Sie auf dem Zielrechner manuell nachtragen, wenn Sie VBA-Code importieren.

Vorgehensweise

VBA-Code exportieren

1. Wählen Sie im Project Explorer des VBA-Editor das Modul aus, dessen VBA-Code Sie exportieren wollen.
2. Wählen Sie den Menübefehl "File" > "Export File".

3. Wählen Sie den Pfad aus und geben Sie den Dateinamen ein.
4. Klicken Sie auf "Speichern".

Der VBA-Code wird in eine Datei exportiert. Der Dateityp ist abhängig vom Modul, aus dem der VBA-Code exportiert wurde.

VBA-Code importieren

1. Wählen Sie im Project Explorer des VBA-Editors das Dokument aus, in das Sie den VBA-Code importieren wollen.
2. Wählen Sie den Menübefehl "File" > "Import File".
3. Wählen Sie die Datei aus und klicken Sie auf "Öffnen", um den VBA-Code als "ThisDocument" in den Ordner "Class Modules" zu importieren.
4. Öffnen Sie im Ordner "Class Modules" das Dokument "ThisDocument" und kopieren Sie den VBA-Code in das Dokument des gewünschten Projektes.

Siehe auch

VBA-Code im WinCC-Projekt organisieren (Seite 1617)

3.2.5 VBA-Makros im Graphics Designer ausführen

Einleitung

Um VBA-Makros im Graphics Designer auszuführen, stehen Ihnen drei Möglichkeiten zur Verfügung:

- Event-Handling
- Benutzerdefiniertes Menü oder Symbolleiste
- VBA-Editor

Event-Handling

Im Graphics Designer, dem aktiven Bild und der Bausteinbibliothek können vordefinierte Ereignisse (z.B. das Öffnen eines Bildes) eintreten, auf die Sie mit VBA-Event-Handlern reagieren können. Diese Ereignisse treten ausschließlich während der Projektierung im Graphics Designer auf und haben mit den Ereignissen der Aktionsprojektierung nichts zu tun.

In diesem Beispiel soll beim Öffnen eines Bildes eine kurze Meldung ausgegeben werden. Dazu wird das "Opened"-Ereignis verwendet:

```
Private Sub Document_Opened(CancelForwarding As Boolean)
MsgBox ("Bild wurde geöffnet!")
End Sub
```

Weitere Informationen zum Thema "Event-Handling" finden Sie unter "Event-Handling" und "Ereignisse".

Benutzerdefiniertes Menü oder Symbolleiste

Mit VBA haben Sie die Möglichkeit, im Graphics Designer benutzerdefinierte Menüs und Symbolleisten anzulegen. Jedem benutzerdefinierten Menüeintrag oder Symbo können Sie ein VBA-Makro zuordnen, das ausgeführt wird, wenn Sie den Menüeintrag oder das Symbol anklicken. Auf diese Weise können Sie die Funktionalität des Graphics Designer nach Ihren Bedürfnissen erweitern.

Weitere Informationen zum Anlegen benutzerdefinierter Menüs und Symbolleisten finden Sie unter "Eigene Menüs und Symbolleisten anlegen".

VBA-Editor

Sie können im VBA-Editor ein VBA-Makro mit <F5> starten. Mit <F8> können Sie ein VBA-Makro schrittweise ausführen.

Siehe auch

VBA-Referenz (Seite 1735)

Event-Handling (Seite 1715)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

VBA im Graphics Designer (Seite 1623)

VBA-Code im WinCC-Projekt organisieren (Seite 1617)

Einführung: VBA in WinCC einsetzen (Seite 1616)

3.3 VBA im Graphics Designer

3.3.1 VBA im Graphics Designer

Einleitung

Sie verwenden VBA im Graphics Designer, um häufig wiederkehrende Arbeitsschritte während der Projektierung zu automatisieren. Sie können benutzerdefinierte Menüs und Symbolleisten anlegen, um Ihre erstellen VBA-Makros komfortabel ausführen zu können.

Grundsätzlich können Sie im Graphics Designer alle Projektierungen, die Sie sonst mit der Maus ausführen, durch VBA-Makros ersetzen. Dies betrifft insbesondere die Oberfläche (Ebenen und Zoom) und das Bearbeiten von Objekten in Bildern einschließlich der Dynamisierung.

Graphics Designer mit VBA anpassen

Der Graphics Designer wird in VBA durch das Application-Objekt repräsentiert. Mit VBA können Sie im Graphics Designer in mehreren Sprachen projektieren, benutzerdefinierte Menüs und Symbolleisten anlegen und auf die Bausteinbibliothek zugreifen.

Bilder mit VBA bearbeiten

Ein Bild im Graphics Designer wird durch das Document-Objekt repräsentiert.

Mit VBA können Sie auf die Eigenschaften des Bildes zugreifen und Einstellungen für Ebenen und die Zoomfaktoren bearbeiten. Darüberhinaus können Sie bildspezifische Menüs und Symbolleisten anlegen. Diese sind jedoch nur sichtbar, solange das Bild aktiv ist.

Objekte mit VBA bearbeiten

Ein Objekt im Bild wird durch das HMIObject-Objekt repräsentiert. Mit VBA können Sie Objekte anlegen und löschen und auf die Objekteigenschaften zugreifen. Mit VBA können Sie z.B. in kurzer Zeit eine große Anzahl von Objekten mit identischen Eigenschaften für Ihr Anlagenbild anlegen.

Dynamisierungen anlegen mit VBA

Mit VBA können Sie Eigenschaften und Ereignisse von Bildern und Objekten dynamisieren.

Event-Handling

Mit VBA können Sie auf Ereignisse reagieren, die im Graphics Designer oder im Bild eintreten, wenn Sie z.B. ein neues Objekt in ein Bild einfügen. Das Event-Handling verwenden Sie, um VBA-Makros in bestimmten Programmsituationen auszuführen.

Zugriff auf Fremdapplikationen

Sie können mit VBA auf Programme zugreifen, die VBA unterstützen , z.B. die Produkte der Mircosoft Office-Familie. Damit haben Sie z.B. die Möglichkeit, Werte aus einer Excel-Tabelle auszulesen und diese dann Objekteigenschaften zuzuweisen.

Hinweis

Zugriff auf Applikationen, die mit .net kompiliert sind

Die mit .net kompilierten Applikationen müssen noch einmal kompiliert werden, um Zugriff auf VBA im Graphics Designer zu haben.

Siehe auch

Bilder mit VBA bearbeiten (Seite 1656)

SymbolLibrary-Objekt (Seite 2035)

HMIObject-Objekt (Seite 1955)

Document-Objekt (Seite 1920)

Application-Objekt (Seite 1888)

Zugriff auf Fremdapplikationen mit VBA (Seite 1718)

Event-Handling (Seite 1715)

Dynamisierungen anlegen mit VBA (Seite 1691)

Objekte mit VBA bearbeiten (Seite 1662)

Graphics Designer mit VBA anpassen (Seite 1625)

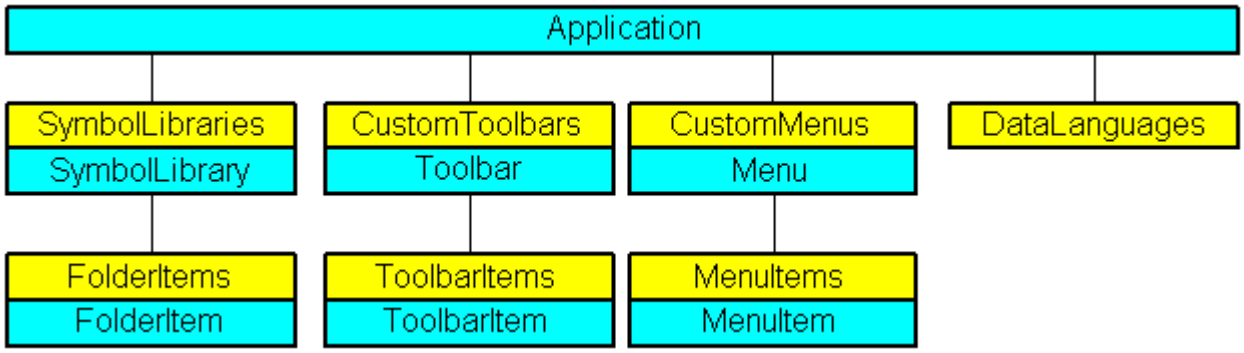
Einführung: VBA in WinCC einsetzen (Seite 1616)

3.3.2 Graphics Designer mit VBA anpassen

3.3.2.1 Graphics Designer mit VBA anpassen

Einleitung

In VBA stellt das Application-Objekt den Graphics Designer dar:



Zugriff auf die Bausteinbibliothek

Mit VBA haben Sie vollen Zugriff auf die Bausteinbibliothek. Sie können die Bausteinbibliothek mit VBA erweitern, in dem sie z.B. Ordner anlegen und löschen sowie Objekte kopieren und in ein Bild einfügen.

Benutzerdefinierte Menüs und Symbolleisten

Um im Graphics Designer VBA-Makros auszuführen, können Sie benutzerdefinierte Menüs und Symbolleisten anlegen. Damit erweitern Sie die Funktionalität des Graphics Designer nach Ihren individuellen Bedürfnissen.

Sprachabhängige Projektierung

Mit VBA können Sie im Graphics Designer mehrsprachig projektieren. Sie haben damit Zugriff auf die sprachabhängigen Objekteigenschaften und können die benutzerdefinierten Menüs und Symbolleisten in mehreren Sprachen anlegen.

Siehe auch

Bilder mit VBA bearbeiten (Seite 1656)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

Sprachabhängige Projektierung mit VBA (Seite 1626)

VBA im Graphics Designer (Seite 1623)

3.3.2.2 Sprachabhängige Projektierung mit VBA

Einleitung

Mit VBA können Sie im Graphics Designer für mehrere Sprachen projektieren. Sie haben einerseits Zugriff auf die sprachabhängigen Eigenschaften von Objekten im Graphics Designer und können andererseits die benutzerdefinierten Menüs und Symbolleisten in mehreren Sprachen zur Verfügung stellen. In VBA werden fremdsprachige Texte in einer Auflistung vom Typ "LanguageTexts" gespeichert. Die Einstellungen zu sprachabhängigen Schriftarten werden in einer Auflistung vom Typ "LanguageFonts" abgelegt.

Weitere Informationen zum sprachabhängigen Projektieren finden Sie auch in der WinCC-Dokumentation "Aufbau mehrsprachiger Projekte".

Oberflächen-Sprache

Die Oberflächen-Sprache können Sie nur in WinCC, jedoch nicht mit VBA umschalten. Wenn Sie die Oberflächen-Sprache in WinCC umschalten, wird das Ereignis "DesktopLanguageChanged" ausgelöst. Sie können die benutzerdefinierten Menüs und Symbolleisten anwendergerecht anpassen, indem Sie z.B sprachabhängige Symbol-Icons austauschen.

Folgende Objekte und die dazugehörigen, sprachabhängigen Eigenschaften reagieren auf die Umschaltung der Oberflächen-Sprache:

- FolderItem-Objekt
- Menu- und MenuItem-Objekt
- ToolbarItem-Objekt
- Weitere Informationen zur Oberflächen-Sprache finden Sie in der WinCC-Dokumentation "Aufbau mehrsprachiger Projekte" unter "Sprachbegriffe in WinCC".

Projektiersprache

Die Projektiersprache können Sie mit VBA mit der Eigenschaft "CurrentDataLanguage" umschalten.

In diesem Beispiel wird die Projektiersprache auf "Englisch" umgeschaltet:

```
Sub ChangeCurrentDataLanguage()
'VBA1
Application.CurrentDataLanguage = 1033
MsgBox "The Data language has been changed to english"
Application.CurrentDataLanguage = 1031
MsgBox "The Data language has been changed to german"
End Sub
```

Alle sprachabhängigen Eigenschaften wie z.B. ToolTipText sind von der Umschaltung betroffen.

Projektieren für mehrere Sprachen in VBA

Sie haben zwei Möglichkeiten, um mit VBA für mehrere Sprachen zu projektieren.

- Sprachumschaltung: Texteigenschaften von Objekten.
- Auflistung Language Texts: Texteigenschaften von benutzerdefinierten Menüs und Symbolleisten sowie Objekten.

Sprachumschaltung

Sie können die sprachabhängigen Eigenschaften (z.B. "Text") von Objekten mit VBA ändern. Dazu weisen Sie der entsprechenden Eigenschaft den Text zu und wechseln danach die Projektiersprache, um den fremdsprachigen Text zuzuweisen.

Auflistung LanguageTexts

Sie können die mehrsprachigen Texte des jeweiligen Objektes direkt in der dazugehörigen Auflistung vom Typ "LanguageTexts" speichern. Dazu geben Sie die Sprachenkennung der Sprache sowie den dazugehörigen Text ein.

Die Liste mit den Sprachkennungen finden Sie in der WinCC-Dokumentation (Index > Language Code).

In diesem Beispiel wird dem Button "myButton" eine deutsche und englische Beschriftung zugewiesen:

```
Sub AddLanguagesToButton()
'VBA2
Dim objLabelText As HMILanguageText
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
'
'Set defaultlabel:
objButton.Text = "Default-Text"
'
'Add english label:
Set objLabelText = objButton.LDTexts.Add(1033, "English Text")
'Add german label:
Set objLabelText = objButton.LDTexts.Add(1031, "German Text")
End Sub
```

Siehe auch

LanguageTexts-Objekt (Auflistung) (Seite 1966)

LanguageFonts-Objekt (Auflistung) (Seite 1963)

So weisen Sie Menüs und Symbolleisten Hilfetexte zu (Seite 1641)

So legen Sie Menüs mehrsprachig an (Seite 1635)

VBA-Referenz (Seite 1735)

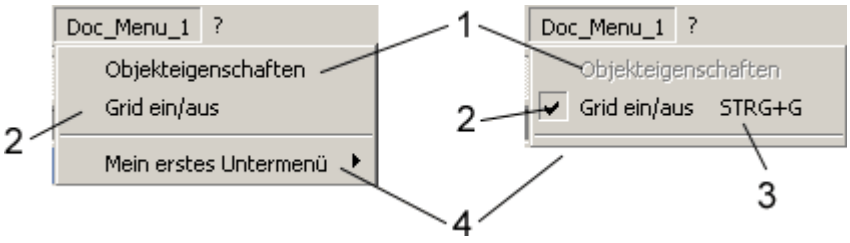
3.3.2.3 Eigene Menüs und Symbolleisten anlegen

Menüs und Symbolleisten konfigurieren

Einleitung

Sie können benutzerdefinierte Menüs und Symbolleisten mit "Leben" füllen, damit sie auf Programmsituationen im Graphics Designer reagieren. Wenn z.B. ein Symbol nicht verfügbar sein soll, weil kein Objekt ausgewählt ist, können Sie es ausgrauen. Ein Häkchen vor einem Menüeintrag kann beispielsweise anzeigen, ob eine Auswahl aktiviert ist.

Die folgende Abbildung zeigt Ihnen am Beispiel eines benutzerdefinierten Menüs die Konfigurationsmöglichkeiten:



Aktiv (ja/nein) (1)

Aktiviert den Eintrag oder graut ihn aus. Die Eigenschaft "Enabled" können Sie für benutzerdefinierte Menüs, Menüeinträge und Symbole verwenden:

```
'VBA13
Application.ActiveDocument.CustomMenus(1).MenuItems(1).Enabled = False
```

Mit Häkchen markiert (ja/nein) (2)

Markiert den Menüeintrag mit einem Häkchen. Die Eigenschaft "Checked" können Sie nur für benutzerdefinierte Menüeinträge verwenden:

```
'VBA14
Application.ActiveDocument.CustomMenus(1).MenuItems(2).Checked = True
```

Shortcut (3)

Definiert eine Tastaturkombination für einen Menüeintrag oder ein Symbol. Die Eigenschaft "Shortcut" können Sie nur für benutzerdefinierte Menüeinträge und Symbole verwenden:

```
'VBA15
Application.ActiveDocument.CustomMenus(1).MenuItems(3).Shortcut = "Strg+G"
```

Sichtbar (ja/nein) (4)

Blendet den Eintrag aus oder ein. Die Eigenschaft "Visible" können Sie für benutzerdefinierte Menüs, Menüeinträge und Symbolleisten sowie deren Symbole verwenden:

```
'VBA16
Application.ActiveDocument.CustomMenus(1).MenuItems(4).Visible = False
```

Eigene Menüs und Symbolleisten anlegen

Benutzerdefinierte Menüs und Symbolleisten im Graphics Designer

Sie können im Graphics Designer benutzerdefinierte Menüs und Symbolleisten verwenden, um VBA-Makros auszuführen. Es wird dabei zwischen anwendungsspezifischen und bildspezifischen Menüs und Symbolleisten unterschieden, die folgende Eigenschaften haben:

- Anwendungsspezifisches Menü/Symbolleiste: Ist immer sichtbar, wenn der Graphics Designer geöffnet ist. Verwenden Sie anwendungsspezifische Menüs und Symbolleisten dann, wenn die dort auszuführenden VBA-Makros immer zugänglich sein müssen.
- Bildspezifisches Menü/Symbolleiste: Ist an ein bestimmtes Bild gebunden und bleibt solange sichtbar, wie das Bild aktiv ist. Verwenden Sie bildspezifische Menüs und Symbolleisten, wenn die dort verwendeten VBA-Makros nur für das Bild relevant sind.

Platzierung von benutzerdefinierten Menüs und Symbolleisten

Bei den benutzerdefinierten Menüs entscheidet der Parameter "Position" über die endgültige Platzierung in der Menüleiste. Jedoch werden

- anwendungsspezifische Menüs stets rechts vom Menü "Fenster" im Graphics Designer platziert, während

bildspezifische Menüs stets links vom Menü "?" im Graphics Designer platziert werden.

Anwendungsspezifische Symbolleisten hingegen werden nicht "bevorzugt" behandelt: Hier entscheidet die Reihenfolge, in der Sie die Symbolleisten einfügen über die Platzierung. Symbolleisten werden unterhalb der Symbolleiste des Graphics Designer platziert:

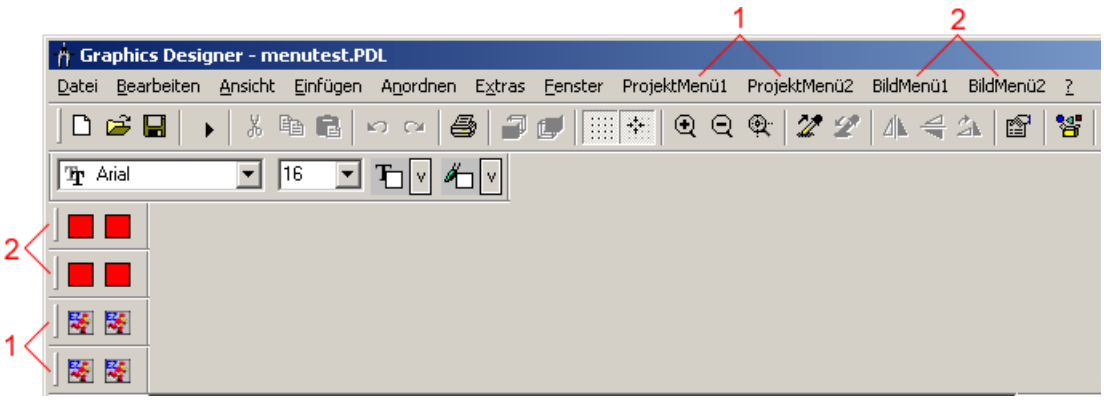


Bild 3-1 Abbildung: Benutzerdefinierte Menüs und Symbolleisten. (1) anwendungsspezifisch, (2) bildspezifisch

In der Abbildung oben wurden zuerst die bildspezifischen, danach die anwendungsspezifischen Menüs und Symbolleisten eingefügt.

Eigenschaften von benutzerdefinierten Menüs und Symbolleisten

Bei benutzerdefinierten Menüs und Symbolleisten können Sie Trennstriche verwenden, um die Einträge z.B. nach Kategorien zu gliedern. Darüberhinaus können Sie in einem benutzerdefinierten Menü auch Untermenüs anlegen.

Für benutzerdefinierte Menüs und Symbolleisten (und deren Einträge) haben Sie folgende Konfigurationsmöglichkeiten:

- Sichtbar (ja/nein): Blendet den Eintrag aus oder ein (Visible-Eigenschaft).
- Aktiv (ja/nein): Aktiviert den Eintrag oder graut ihn aus (Enabled-Eigenschaft).
- Mit Häkchen markiert (ja/nein) - nur für Menüeintrag verfügbar (Checked-Eigenschaft).
- Shortcut: Tastenkombination zum Aufruf des Menüeintrages (ShortCut-Eigenschaft).
- Statustext: Text, der in der Statuszeile angezeigt wird (StatusText-Eigenschaft).
- Tooltiptext - nur für Symbole verfügbar (ToolTipText-Eigenschaft).

Sie können z.B. einen Menüeinträg ausblenden, wenn das Makro zu einem bestimmten Zeitpunkt nicht ausführbar ist. Dadurch können Sie Fehlbedienungen verhindern.

Sie können alle Texte und Beschriftungen von benutzerdefinierten Menüs und Symbolleisten mehrsprachig anlegen, damit auch die benutzerdefinierten Menüs und Symbolleisten auf Sprachumschaltung reagieren können.

Siehe auch

So weisen Sie Menüs und Symbolleisten VBA-Makros zu (Seite 1644)
So weisen Sie Menüs und Symbolleisten Hilfetexte zu (Seite 1641)
So fügen Sie ein neues Symbol zur Symbolleiste hinzu (Seite 1639)
So legen Sie eine neue anwendungsspezifische Symbolleiste an (Seite 1637)
So legen Sie Menüs mehrsprachig an (Seite 1635)
So fügen Sie einen neuen Menüeintrag zum Menü hinzu (Seite 1632)
So legen Sie bildspezifische Menüs und Symbolleisten an (Seite 1657)
So legen Sie ein neues anwendungsspezifisches Menü an (Seite 1631)
Menüs und Symbolleisten konfigurieren (Seite 1628)
VBA-Makros im Graphics Designer ausführen (Seite 1621)

So legen Sie ein neues anwendungsspezifisches Menü an

Einleitung

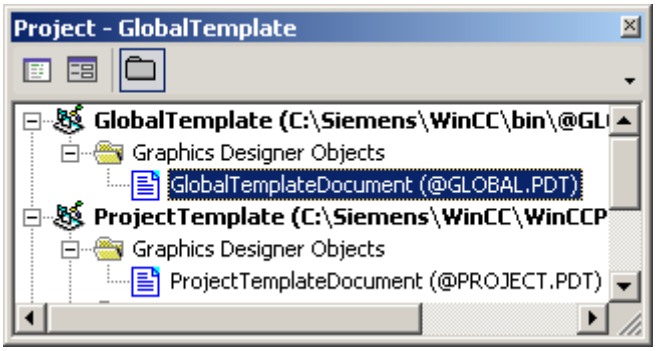
Anwendungsspezifische Menüs sind auch dann noch sichtbar, wenn alle Bilder im Graphics Designer geschlossen sind. Verwende Sie z.B. das Started-Ereignis, um ein anwendungsspezifisches Menü frühzeitig einzufügen.

Platzieren Sie den VBA-Code entweder

- im "GlobalTemplateDocument", wenn das Menü in allen Projekten verfügbar sein soll, oder
- im "ProjectTemplateDocument", wenn das Menü im aktuellen Projektes verfügbar sein soll.

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor")
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



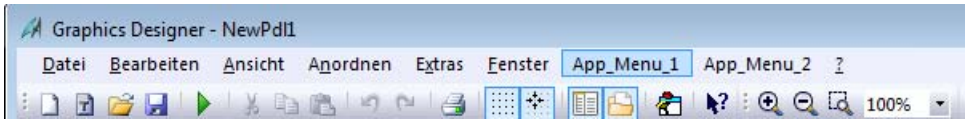
3. Um ein benutzerdefiniertes Menü im Graphics Designer anzulegen, fügen Sie z.B. eine Prozedur "CreateApplicationMenus()" in das Dokument ein. In diesem Beispiel werden zwei benutzerdefinierte Menüs angelegt:

```
Sub CreateApplicationMenus()
'VBA3
'Declaration of menus...:
Dim objMenu1 As HMIMenu
Dim objMenu2 As HMIMenu
'
'Add menus. Parameters are "Position", "Key" und "DefaultLabel":
Set objMenu1 = Application.CustomMenus.InsertMenu(1, "AppMenu1",
"App_Menu_1")
Set objMenu2 = Application.CustomMenus.InsertMenu(2, "AppMenu2",
"App_Menu_2")
End Sub
```

4. Starten Sie die Prozedur mit <F5>.

Ergebnis

Die beiden Menüs "App_Menu_1" und "App_Menu_2" werden rechts vom Menü "Fenster" eingefügt:



Siehe auch

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

InsertMenu-Methode (Seite 1836)

So weisen Sie Menüs und Symbolleisten VBA-Makros zu (Seite 1644)

So weisen Sie Menüs und Symbolleisten Hilfetexte zu (Seite 1641)

So legen Sie Menüs mehrsprachig an (Seite 1635)

So fügen Sie einen neuen Menüeintrag zum Menü hinzu (Seite 1632)

Menüs und Symbolleisten konfigurieren (Seite 1628)

VBA-Code im WinCC-Projekt organisieren (Seite 1617)

So fügen Sie einen neuen Menüeintrag zum Menü hinzu

Voraussetzung

Sie müssen das benutzerdefinierte Menü angelegt haben.

Einleitung

In das benutzerdefinierte Menü können Sie drei unterschiedliche Typen von Menüeinträgen einfügen:

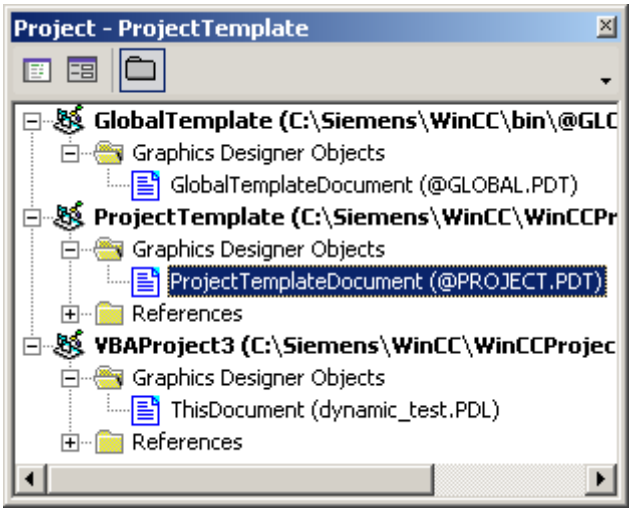
- Menüeintrag: Zum Aufrufen von VBA-Makros.
- Trennlinie: Zum übersichtlicheren Gestalten des benutzerdefinierten Menüs.
- Untermenü: Wie benutzerdefiniertes Menü (z.B. zur Gliederung von Befehlen).

Innerhalb des benutzerdefinierten Menüs bestimmt der Parameter "Position" die Reihenfolge der Menüeinträge.

Der Parameter "Key" identifiziert den Menüeintrag eindeutig. Der Parameter wird verwendet, wenn Sie das "MenuItemClicked"-Ereignis zum Aufrufen von VBA-Makros verwenden.

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor")
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



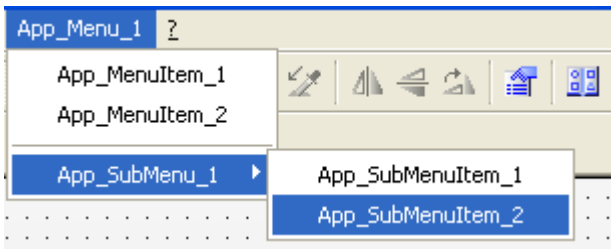
3. Um Menüeinträge in einem zuvor angelegten benutzerdefinierten Menü zu erstellen, fügen Sie z.B. eine Prozedur "InsertMenuItems()" in das Dokument ein. In diesem Beispiel im benutzerdefinieten Menü "App_Menu_1" einige Menüeinträge erstellt:

```
Sub InsertMenuItems()
'VBA4
Dim objMenu1 As HMIMenu
Dim objMenu2 As HMIMenu
Dim objMenuItem1 As HMIMenuItem
Dim objSubMenu1 As HMIMenuItem
'Create Menu:
Set objMenu1 = Application.CustomMenus.InsertMenu(1, "AppMenu1",
"App_Menu_1")
'Next lines add menu-items to userdefined menu.
'Parameters are "Position", "Key" and DefaultLabel:
Set objMenuItem1 = objMenu1.MenuItems.InsertMenuItem(1,
"mItem1_1", "App_MenuItem_1")
Set objMenuItem1 = objMenu1.MenuItems.InsertMenuItem(2,
"mItem1_2", "App_MenuItem_2")
'
'Adds seperator to menu ("Position", "Key")
Set objMenuItem1 = objMenu1.MenuItems.InsertSeparator(3,
"mItem1_3")
'
'Adds a submenu into a userdefined menu
Set objSubMenu1 = objMenu1.MenuItems.InsertSubMenu(4, "mItem1_4",
"App_SubMenu_1")
'
'Adds a menu-item into a submenu
Set objMenuItem1 = objSubMenu1.SubMenu.InsertMenuItem(5,
"mItem1_5", "App_SubMenuItem_1")
Set objMenuItem1 = objSubMenu1.SubMenu.InsertMenuItem(6,
"mItem1_6", "App_SubMenuItem_2")
End Sub
```

4. Starten Sie die Prozedur mit <F5>.

Ergebnis

Die Prozedur "InsertMenuItems()" fügt das Menü "App_Menu_1" mit diesen Menüeinträgen ein:



Siehe auch

InsertSeparator-Methode (Seite 1839)

InsertSubmenu-Methode (Seite 1840)

InsertMenu-Methode (Seite 1836)

So weisen Sie Menüs und Symbolleisten VBA-Makros zu (Seite 1644)

So weisen Sie Menüs und Symbolleisten Hilfetexte zu (Seite 1641)

So legen Sie Menüs mehrsprachig an (Seite 1635)

So legen Sie ein neues anwendungsspezifisches Menü an (Seite 1631)

Menüs und Symbolleisten konfigurieren (Seite 1628)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

So legen Sie Menüs mehrsprachig an

Einleitung

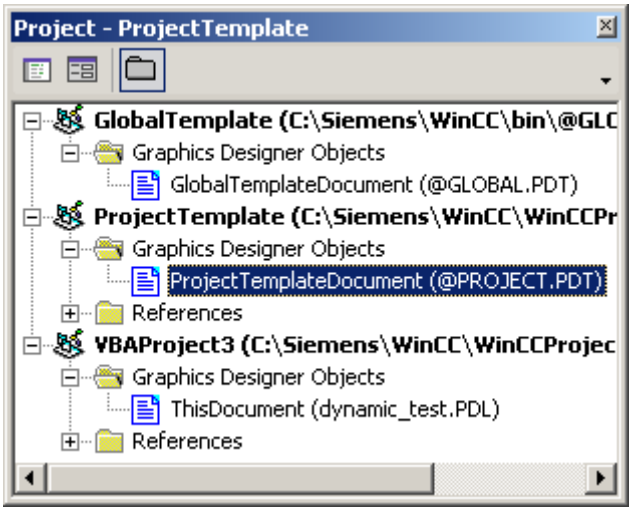
Sie können ein benutzerdefinierte Menü anlegen, das auf Sprachumschaltung reagiert. Dazu definieren Sie für das Menü sowie für jeden Menüeintrag die benötigte Anzahl der fremdsprachigen Beschriftungen.

Die fremdsprachige Beschriftung setzt sich aus der Sprachkennung (LCID) und dem fremdsprachigen Text (DisplayName) zusammen.

Die Liste mit den Sprachkennungen finden Sie in der WinCC-Dokumentation (Index >Language Code).

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor")
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



3. Um mehrsprachige Beschriftungen für ein benutzerdefiniertes Menü zu definieren, fügen Sie z.B. eine Prozedur "MultipleLanguagesForAppMenu1()" in das Dokument ein. In diesem Beispiel werden englische Beschriftungen für das Menü "App_Menu_1" definiert:

```
Sub InsertMenuItems()
'VBA5
'Execute this procedure first
Dim objMenu1 As HMIMenu
Dim objMenu2 As HMIMenu
Dim objMenuItem1 As HMIMenuItem
Dim objSubMenu1 As HMIMenuItem
'Insert Menu:
Set objMenu1 = Application.CustomMenus.InsertMenu(1, "AppMenu1",
"App_Menu_1")
'Next lines inserts menu-items to userdefined menu.
'parameters are "Position", "Key" and DefaultLabel:
Set objMenuItem1 = objMenu1.MenuItems.InsertMenuItem(1,
"mItem1_1", "App_MenuItem_1")
Set objMenuItem1 = objMenu1.MenuItems.InsertMenuItem(2,
"mItem1_2", "App_MenuItem_2")
'
'Inserts seperator into menu ("Position", "Key")
Set objMenuItem1 = objMenu1.MenuItems.InsertSeparator(3,
"mItem1_3")
'
'Inserts a submenu into a userdefined menu
Set objSubMenu1 = objMenu1.MenuItems.InsertSubMenu(4, "mItem1_4",
"App_SubMenu_1")
'
'Inserts a menu-item into a submenu
Set objMenuItem1 = objSubMenu1.SubMenu.InsertMenuItem(5,
"mItem1_5", "App_SubMenuItem_1")
Set objMenuItem1 = objSubMenu1.SubMenu.InsertMenuItem(6,
"mItem1_6", "App_SubMenuItem_2")
End Sub
Sub MultipleLanguagesForAppMenu1()
'   execute this procedure after "InsertMenuItems()" was run
'Object "objLanguageTextMenu1" contains the
'foreign-language labels for the menu
Dim objLanguageTextMenu1 As HMILanguageText
'
'Object "objLanguageTextMenu1Item" contains the
'foreign-language labels for the menu-items
Dim objLanguageTextMenuItem1 As HMILanguageText
Dim objMenu1 As HMIMenu
Dim objSubMenu1 As HMIMenuItem
Set objMenu1 = Application.CustomMenus("AppMenu1")
Set objSubMenu1 =
Application.CustomMenus("AppMenu1").MenuItems("mItem1_4")
'
'Inserts foreign-language label into a menu:
'("Add(LCID, DisplayName)"-Methode:
```

```
Set objLanguageTextMenu1 = objMenu1.LDLabelTexts.Add(1033,
"English_App_Menu_1")
'
'Inserts foreign-language label into a menuitem:
Set objLanguageTextMenuItem1 =
objMenu1.MenuItems("mItem1_1").LDLabelTexts.Add(1033, "My first
menu item")
'
'Adds a foreign-language label into a submenu:
Set objLanguageTextMenuItem1 =
objSubMenu1.SubMenu.Item("mItem1_5").LDLabelTexts.Add(1033, "My
first submenu item")
End Sub
```

4. Starten Sie die Prozedur mit <F5>.

Ergebnis

Wenn Sie jetzt die Projektiersprache auf Englisch umschalten, werden einige Einträge im benutzerdefinierten Menü auf Englisch dargestellt.

Siehe auch

LanguageTexts-Objekt (Auflistung) (Seite 1966)

LDLabelTexts-Eigenschaft (Seite 2261)

So weisen Sie Menüs und Symbolleisten VBA-Makros zu (Seite 1644)

So weisen Sie Menüs und Symbolleisten Hilfetexte zu (Seite 1641)

So fügen Sie einen neuen Menüeintrag zum Menü hinzu (Seite 1632)

So legen Sie ein neues anwendungsspezifisches Menü an (Seite 1631)

Menüs und Symbolleisten konfigurieren (Seite 1628)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

Sprachabhängige Projektierung mit VBA (Seite 1626)

So legen Sie eine neue anwendungsspezifische Symbolleiste an

Einleitung

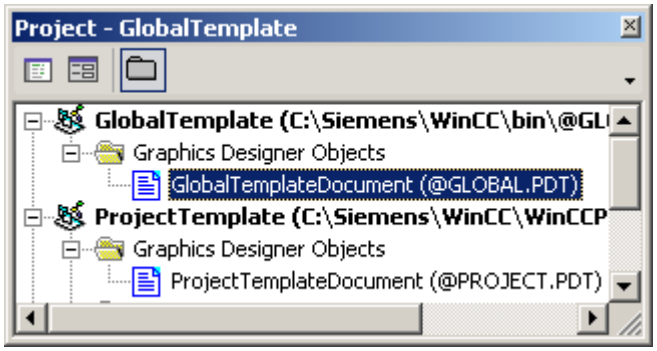
Anwendungsspezifische Symbolleisten sind auch dann noch sichtbar, wenn alle Bilder im Graphics Designer geschlossen sind.

Platzieren Sie den VBA-Code entweder

- im "GlobalTemplateDocument", wenn die Symbolleiste in allen Projekten verfügbar sein soll, oder
- im "ProjectTemplateDocument", wenn die Symbolleiste im aktuellen Projektes verfügbar sein soll.

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor")
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



3. Um ein benutzerdefinierte Symbolleiste im Graphics Designer anzulegen, fügen Sie z.B. eine Prozedur "CreateApplicationToolbars()" in das Dokument ein. In diesem Beispiel werden zwei benutzerdefinierte Symbolleisten angelegt:

```
Sub CreateApplicationToolbars()
'VBA6
'Declare toolbar-objects...:
Dim objToolbar1 As HMIToolbar
Dim objToolbar2 As HMIToolbar
'
'Add the toolbars with parameter "Key"
Set objToolbar1 = Application.CustomToolbars.Add("AppToolbar1")
Set objToolbar2 = Application.CustomToolbars.Add("AppToolbar2")
End Sub
```

4. Starten Sie die Prozedur mit <F5>.

Ergebnis

Die beiden Symbolleisten werden unterhalb der Symbolleisten des Graphics Designers eingefügt.

Siehe auch

Add-Methode (CustomToolbars-Auflistung) (Seite 1779)

So weisen Sie Menüs und Symbolleisten VBA-Makros zu (Seite 1644)

So weisen Sie Menüs und Symbolleisten Hilfetexte zu (Seite 1641)

So fügen Sie ein neues Symbol zur Symbolleiste hinzu (Seite 1639)

Menüs und Symbolleisten konfigurieren (Seite 1628)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

So fügen Sie ein neues Symbol zur Symbolleiste hinzu

Voraussetzung

Sie müssen die benutzerdefinierte Symbolleiste angelegt haben.

Einleitung

In die benutzerdefinierte Symbolleiste können Sie zwei unterschiedliche Typen von Objekten einfügen:

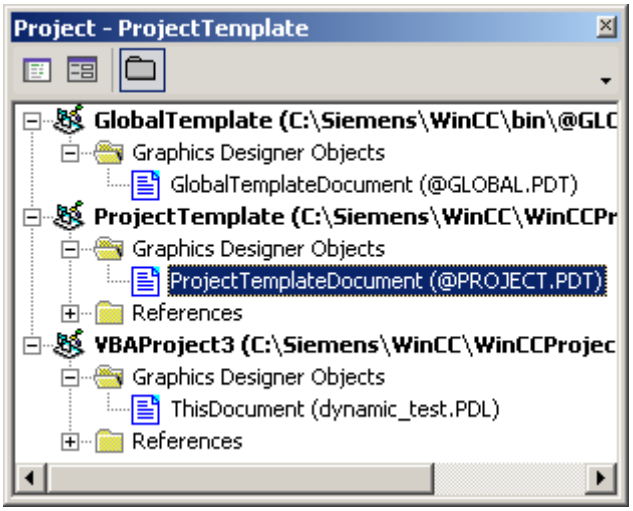
- Symbol: Zum Aufrufen von VBA-Makros.
- Trennlinie: Zum übersichtlicheren Gestalten der benutzerdefinierten Symbolleiste.

Innerhalb der benutzerdefinierten Symbolleiste bestimmt der Parameter "Position" die Reihenfolge der Symbole.

Der Parameter "Key" identifiziert das Symbol eindeutig. Der Parameter wird verwendet, wenn Sie das "ToolbarItemClicked"-Ereignis zum Aufrufen von VBA-Makros verwenden.

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor")
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



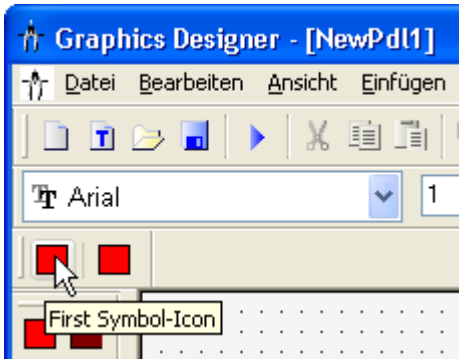
3. Um Symbole in einer zuvor angelegten benutzerdefinierten Symbolleiste zu erstellen, fügen Sie z.B. eine Prozedur "InsertToolbarItems()" in das Dokument ein. In diesem Beispiel werden in der benutzerdefinierten Symbolleiste "AppToolbar1" zwei Symbole erstellt, die durch eine Trennlinie getrennt sind:

```
Sub InsertToolbarItems()
'VBA7
Dim objToolbar1 As HMIToolbar
Dim objToolbarItem1 As HMIToolbarItem
'
'Add a new toolbar:
Set objToolbar1 = Application.CustomToolbars.Add("AppToolbar1")
'Adds two toolbar-items to the toolbar
'("InsertToolbarItem(Position, Key, DefaultToolTipText)"-Methode):
Set objToolbarItem1 =
objToolbar1.ToolbarItems.InsertToolbarItem(1, "tItem1_1", "First
Symbol-Icon")
Set objToolbarItem1 =
objToolbar1.ToolbarItems.InsertToolbarItem(3, "tItem1_2", "Second
Symbol-Icon")
'
'Adds a seperator between the two toolbar-items
'("InsertSeparator(Position, Key)"-Methode):
Set objToolbarItem1 = objToolbar1.ToolbarItems.InsertSeparator(2,
"tSeparator1_3")
End Sub
```

4. Starten Sie die Prozedur mit <F5>.

Ergebnis

Die Prozedur "InsertToolbarItems()" fügt zu den Symbolleisten des Graphics Designers eine Symbolleiste mit zwei Symbolen hinzu, die durch eine Trennlinie getrennt sind:



Hinweis

Verwenden Sie die Icon-Eigenschaft, um für ein Symbol-Icon eine Grafik (*.ICO-Format) festzulegen.

Siehe auch

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

Icon-Eigenschaft (Seite 2218)

InsertSeparator-Methode (Seite 1839)

InsertToolbarItem-Methode (Seite 1842)

So weisen Sie Menüs und Symbolleisten VBA-Makros zu (Seite 1644)

So weisen Sie Menüs und Symbolleisten Hilfetexte zu (Seite 1641)

So legen Sie eine neue anwendungsspezifische Symbolleiste an (Seite 1637)

Menüs und Symbolleisten konfigurieren (Seite 1628)

So weisen Sie Menüs und Symbolleisten Hilfetexte zu

Voraussetzung

Sie müssen das benutzerdefinierte Menü oder die benutzerdefinierte Symbolleiste angelegt haben.

Einleitung

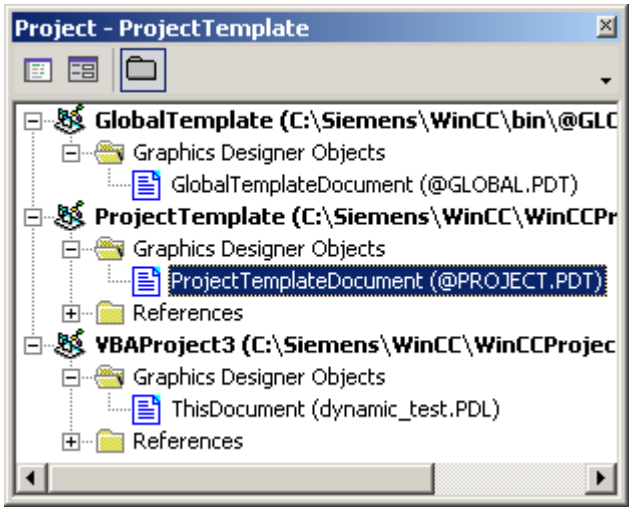
Wenn der Projekteur mit der Maus über einen benutzerdefinierten Menüeintrag oder über ein benutzerdefiniertes Symbol fährt, können Sie zusätzlichen Hilfetext bereitstellen, der die Funktionsweise erläutert:

- Für benutzerdefinierte Menüeinträge und Symbole können Sie Hilfetext definieren, der in der Statuszeile angezeigt wird.
- Bei benutzerdefinierten Symbole legen Sie den Hilfetext standardmäßig als Tooltip an.

Status- und Tooltiptexte können Sie auch für Fremdsprachen definieren.

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor")
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



3. Um einem benutzerdefinierten Menüeintrag einen Statustext zuzuweisen, fügen Sie z.B. eine Prozedur "AddStatusTextsToAppMenu1()" in das Dokument ein. In diesem Beispiel wird dem ersten Menüeintrag des zuvor angelegten Menüs "AppMenu1" ein Statustext auf Deutsch und Englisch zugewiesen:

```
Sub AddStatusTextsToAppMenu1()
'VBA8
Dim objMenu1 As HMIMenu
'
'Object "objStatusTextMenuItem1" contains foreign-language texts
Dim objStatusTextMenuItem1 As HMILanguageText
Set objMenu1 = Application.CustomMenus("AppMenu1")
'
'Assign a statustext to a menuitem:
objMenu1.MenuItems("mItem1_1").StatusText = "Statustext the first
menuitem"
'
'Assign a foreign statustext to a menuitem:
Set objStatusTextMenuItem1 =
objMenu1.MenuItems("mItem1_1").LDStatusTexts.Add(1033, "This is
my first status text in English")
End Sub
```

4. Um einem benutzerdefinierten Symbol Status- und fremdsprachigen Tooltiptext zuzuweisen, fügen Sie z.B. eine Prozedur "AddStatusAndTooltipTextsToAppToolbar1()" in das Dokument ein. In diesem Beispiel wird dem ersten Symbol der zuvor angelegten Symbolleiste Statustext (Deutsch/Englisch) und englischer Tooltiptext zugewiesen:

```
Sub AddStatusAndTooltipTextsToAppToolbar1()
'VBA9
Dim objToolbar1 As HMIToolbar
'
'Variable "StatusTextToolbarItem1" for foreign statustexts
Dim objStatusTextToolbarItem1 As HMILanguageText
'
'Variable "TooltipTextToolbarItem1 for foreign tooltiptexts
Dim objTooltipTextToolbarItem1 As HMILanguageText
Set objToolbar1 = Application.CustomToolbars("AppToolbar1")
'
'Assign a statustext to a toolbaritem:
objToolbar1.ToolbarItems("tItem1_1").StatusText = "Statustext für
das erste Symbol-Icon"
'
'Assign a foreign statustext to a toolbaritem:
Set objStatusTextToolbarItem1 =
objToolbar1.ToolbarItems("tItem1_1").LDStatusTexts.Add(1033,
"This is my first status text in English")
'
'Assign a foreign tooltiptext to a toolbaritem:
Set objTooltipTextToolbarItem1 =
objToolbar1.ToolbarItems("tItem1_1").LDTooltipTexts.Add(1033,
"This is my first tooltip text in English")
End Sub
```

5. Starten Sie die Prozeduren jeweils mit <F5>.

Ergebnis

Der Statustext wird angezeigt, wenn Sie den Mauszeiger über den benutzerdefinierten Menüeintrag oder das Symbol bewegen.

Siehe auch

LDTooltipTexts-Eigenschaft (Seite 2265)

LDStatusTexts-Eigenschaft (Seite 2263)

LanguageTexts-Objekt (Auflistung) (Seite 1966)

Add-Methode (Seite 1776)

So weisen Sie Menüs und Symbolleisten VBA-Makros zu (Seite 1644)

So fügen Sie ein neues Symbol zur Symbolleiste hinzu (Seite 1639)

So fügen Sie einen neuen Menüeintrag zum Menü hinzu (Seite 1632)

Menüs und Symbolleisten konfigurieren (Seite 1628)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

So weisen Sie Menüs und Symbolleisten VBA-Makros zu

Einleitung

Sie haben zwei Möglichkeiten, benutzerdefinierten Menüs und Symbolleisten VBA-Makros zuzuweisen:

- Sie verwenden entweder die VBA-Event-Handler "MenuItemClicked" und "ToolbarItemClicked" oder
- die Eigenschaft "Macro".

Hinweis

Den VBA-Code zum Anlegen der benötigten benutzerdefinierten Menüs und Symbolleisten finden Sie in dieser Dokumentation unter "Neuen Menüeintrag zum Menü hinzufügen" und "Neues Symbol zur Symbolleiste hinzufügen".

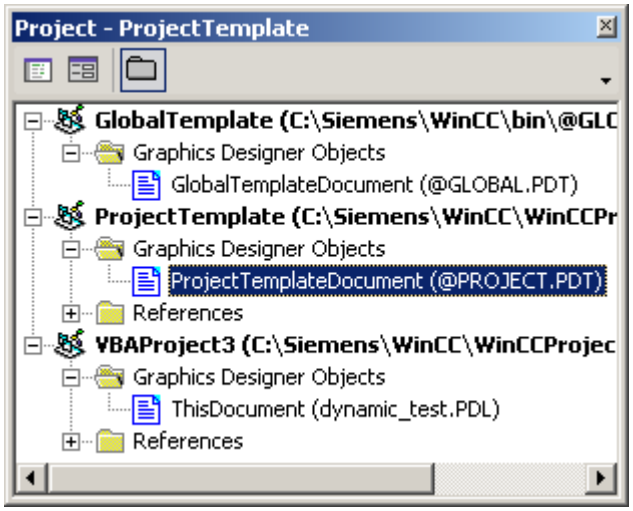
Vorgehensweise

VBA-Makro über VBA-Event-Handler zuweisen

Hinweis

Weitere Informationen zu VBA-Event Handlern finden Sie in dieser Dokumentation unter "Event Handling".

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor")
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:

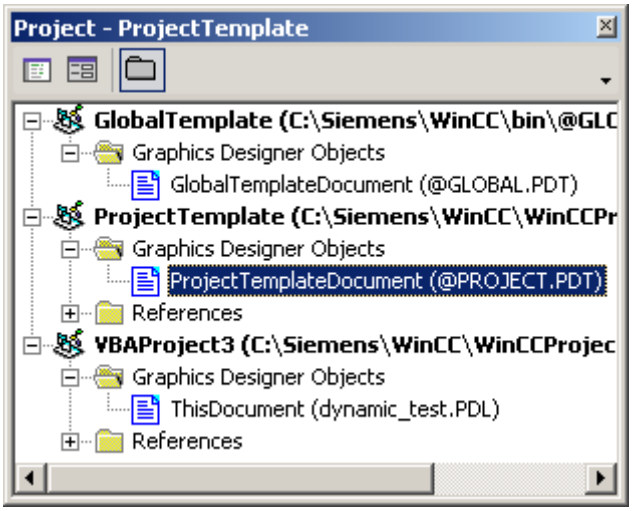


3. Um ein VBA-Makro über die VBA-Event-Handler zu starten, verwenden Sie das "MenuItemClicked"- oder das "ToolbarItemClicked"-Ereignis:

4. Fügen Sie den VBA-Code aus der Tabelle "VBA10" ein.
5. Starten Sie die Prozeduren jeweils mit <F5>.

VBA-Makro über Eigenschaft "Macro" zuweisen

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor")
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



3. Um ein VBA-Makro über die Macro-Eigenschaft zu starten, weisen Sie jedem Menüeintrag oder Symbol das VBA-Makro zu. Im folgenden Beispiel wird ein benutzerdefiniertes Menü mit zwei Menüeinträgen angelegt, die zwei unterschiedliche VBA-Makros aufrufen:
   Der VBA-Code des folgenden Beispiels VBA11 ist abhängig vom Dateityp. Der VBA-Code wird als Beispiel für eine PDL- und eine PDT-Datei angefügt. Die beiden Fälle unterscheiden sich folgendermaßen:
   - PDL-Datei:
     Der VBA-Code in einer PDL-Datei wird nur ausgeführt, wenn diese PDL-Datei gerade angezeigt wird.
   - PDT-Datei:
     Der VBA-Code in einer PDT-Datei wird immer ausgeführt, wenn der Graphics Designer geöffnet ist.
4. Fügen Sie den VBA-Code aus der Tabelle "VBA11: Beispielcode für PDL-Datei" bzw. "VBA821: Beispielcode für PDT-Datei" ein.
   Die beiden folgenden Prozeduren können Sie über die Menüeinträge des benutzerdefinierten Menüs "DocMenu1" aufrufen:
5. Fügen Sie den VBA-Code aus der Tabelle "VBA12" ein.
6. Starten Sie die Prozeduren jeweils mit <F5>.

Die folgenden Tabellen zeigen die VBA-Codes für das Beispiel:

VBA über Event-Handler starten (VBA10)

```
Option Explicit
'VBA10
'The next declaration has to be placed in the modul section
Dim WithEvents theApp As grafexe.Application

Private Sub SetApplication()
'This procedure has to be executed (with "F5") first
Set theApp = grafexe.Application
End Sub

Private Sub theApp_MenuItemClicked(ByVal MenuItem As IHMIMenuItem)
Dim objClicked As HMIMenuItem
Dim varMenuItemKey As Variant
Set objClicked = MenuItem
'
'"varMenuItemKey" contains the value of parameter "Key"
'from clicked menu-item
varMenuItemKey = objClicked.Key
Select Case varMenuItemKey
Case "mItem1_1"
MsgBox "The first menuitem was clicked!"
End Select
End Sub

Private Sub theApp_ToolbarItemClicked(ByVal ToolbarItem As IHMIToolbarItem)
Dim objClicked As HMIToolbarItem
Dim varToolbarItemKey As Variant
Set objClicked = ToolbarItem
'
'"varToolbarItemKey" contains the value of parameter "Key"
'from clicked toolbar-item
varToolbarItemKey = objClicked.Key
Select Case varToolbarItemKey
Case "tItem1_1"
MsgBox "The first symbol-icon was clicked!"
End Select
End Sub
```

Menü erzeugen (VBA11: Beispielcode für PDL-Datei)

```
Sub CreateDocumentMenusUsingMacroProperty()
'VBA11
Dim objDocMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Set objDocMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DocMenu1",
"Doc_Menu_1")
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(1, "dmItem1_1",
"First Menuitem")
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(2, "dmItem1_2",
"Second Menuitem")
```

```
'
'Assign a VBA-macro to every menu item
With ActiveDocument.CustomMenus("DocMenu1")
.MenuItems("dmItem1_1").Macro = "TestMacro1"
.MenuItems("dmItem1_2").Macro = "TestMacro2"
End With
End Sub
```

Menü erzeugen (VBA821: Beispielcode für PDT-Datei)

```
Sub CreateDocumentMenusUsingMacroProperty()
'VBA821
Dim objDocMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Set objDocMenu = Application.CustomMenus.InsertMenu(1, "DocMenu1",
"Doc_Menu_1")
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(1, "dmItem1_1",
"First Menuitem")
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(2, "dmItem1_2",
"Second Menuitem")
'
'Assign a VBA-macro to every menu item
With Application.CustomMenus("DocMenu1")
.MenuItems("dmItem1_1").Macro = "TestMacro1"
.MenuItems("dmItem1_2").Macro = "TestMacro2"
End With
End Sub
```

Makros für benutzerdefinierte Menüeinträge (VBA12)

```
Sub TestMacro1()
'VBA12
MsgBox "TestMacro1 was executed"
End Sub

Sub TestMacro2()
MsgBox "TestMacro2 was executed"
End Sub
```

Siehe auch

Macro-Eigenschaft (Seite 2281)

ToolbarItemClicked-Ereignis (Seite 1772)

MenuItemClicked-Ereignis (Seite 1766)

So fügen Sie ein neues Symbol zur Symbolleiste hinzu (Seite 1639)

So fügen Sie einen neuen Menüeintrag zum Menü hinzu (Seite 1632)

Event-Handling (Seite 1715)

VBA-Makros im Graphics Designer ausführen (Seite 1621)

3.3.2.4 Zugriff auf die Bausteinbibliothek mit VBA

Zugriff auf die Bausteinbibliothek mit VBA

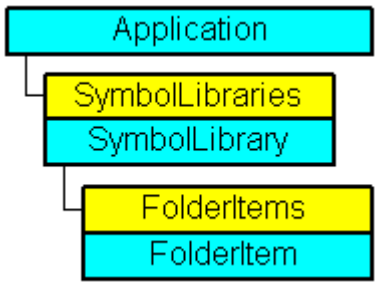
Einleitung

Die Bausteinbibliothek beinhaltet eine große Auswahl von vorgefertigten Objekten, mit denen Sie Ihre Bilder effizient gestalten können. Die Bausteinbibliothek besteht aus einer globalen und einer projektbezogenen Bibliothek:

- Die "Globale Bibliothek" enthält vorgefertigte Objekte, die mit WinCC mitgeliefert werden. Die Objekte sind nach Themen sortiert in Ordnern abgelegt, z.B. Ventile, Motoren, Leitungen und viele mehr.
- Die "Projekt Bibliothek" enthält weder Objekte noch Ordner, wenn Sie ein neues Projekt angelegt haben. In der "Projekt Bibliothek" können Sie Objekte anlegen, die Sie nur in dem Projekt benötigen.

Mit VBA können Sie uneingeschränkt auf die Bausteinbibliothek zugreifen: Sie können Ordner anlegen und löschen sowie Objekte in der Bausteinbibliothek ablegen oder in ein Bild einfügen.

Zugriff auf Bausteinbibliothek mit VBA



Die Bausteinbibliothek wird in VBA durch die Auflistung "SymbolLibraries" repräsentiert. Die Auflistung enthält zwei Elemente, welche die "Globale Bibliothek" und die "Projekt Bibliothek"

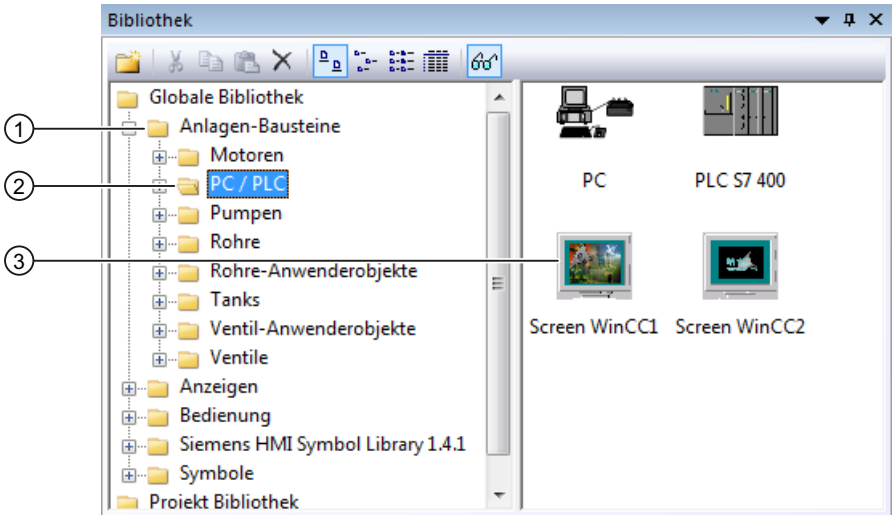
darstellen. Die Auflistung "FolderItems" enthält Elemente, die sowohl Ordner als auch Objekte darstellen.

Hinweis

Um ein Objekt in der Auflistung "SymbolLibraries" anzusprechen, verwenden Sie entweder die Indexnummer oder den internen Namen.

Den internen Namen erhalten Sie, wenn Sie in der Bausteinbibliothek mit der rechten Maustaste auf das gewünschte Objekt klicken und im Kontextmenü den Befehl "Pfad kopieren" wählen.

Der Pfad zu dem Objekt innerhalb der Bausteinbibliothek wird damit in die Zwischenablage kopiert.



Globale Bibliothek (1)

Die "Globale Bibliothek" ist das erste Element der SymbolLibraries-Auflistung, das Sie über die Indexnummer "1" ansprechen. Die "Projekt Bibliothek" sprechen Sie über die Indexnummer "2" an.

Zugriff auf die "Globale Bibliothek" mit VBA:

```
'VBA17
Application.SymbolLibraries(1)
```

Ordner (2)

Ein Ordner der Bausteinbibliothek enthält entweder weitere Ordner oder die Objekte eines Themengebietes. In VBA entspricht ein Ordner dem Objekt "FolderItem" und ist vom Typ "Folder". Die Ordner sind in der Auflistung "FolderItems" enthalten. Mit VBA können Sie einen neuen Ordner anlegen oder löschen und ein Objekt über die Zwischenablage zum Ordner hinzufügen.

Zugriff auf den Ordner "Anlagen-Bausteine" mit VBA:

```
'VBA18
Application.SymbolLibraries(1).FolderItems("Folder2")
```

Objekt (3)

In VBA entspricht ein Objekt dem Objekt "FolderItem" und ist vom Typ "Item". Die Objekte sind in der Auflistung "Folder" enthalten. Mit VBA können Sie ein Objekt löschen oder in die Zwischenablage kopieren.

Zugriff auf das Objekt "PC" mit VBA:

```
'VBA19
Application.SymbolLibraries(1).FolderItems("Folder2").Folder("Folder2").Folder.Item("Objec
t1").DisplayName
```

Ordner in der Bausteinbibliothek anlegen oder löschen

Verwenden Sie folgende Methoden, um Ordner anzulegen oder zu löschen:

- Methode "AddFolder(DefaultName)": Legt einen neuen Ordner in der Bausteinbibliothek an. Ein neu angelegter Ordner erhält als internen Namen "FolderX", wobei das "X" für die laufende Nummer steht.
- Methode "Delete()": Löscht einen vorhandenen Ordner (inklusive aller darin enthaltenen Ordner und Objekte) aus der Bausteinbibliothek.

Objekt in die Bausteinbibliothek einfügen oder löschen

Sie können Objekte innerhalb der Bausteinbibliothek kopieren (z.B. von der "Globalen Bibliothek" in die "Projekt Bibliothek"), ein Objekt aus einem Bild in die Bausteinbibliothek einfügen oder ein Objekt aus der Bausteinbibliothek löschen:

- Methoden "CopyToClipboard()" und "AddFromClipboard()": Kopiert ein Objekt über die Zwischenablage innerhalb der Bausteinbibliothek.
- Methode "AddItem(DefaultName, pHMIObject)": Kopiert ein im Bild vorhandenes Objekt in einen Ordner in der Bausteinbibliothek.
- Methode "Delete()": Löscht ein Objekt.

Objekt oder Ordner in der Bausteinbibliothek suchen

Verwenden Sie die Methode "FindByDisplayName("DisplayName"), um nach einem Objekt oder Ordner zu suchen. Der angegebene Anzeigename ist von der aktuell eingestellten Sprache abhängig. Die Suche endet beim ersten Auftreten des gesuchten Objektes oder Ordners.

Objekt aus der Bausteinbibliothek in Bild einfügen

Verwenden Sie die Methoden "CopyToClipboard()" und "PasteClipboard()", um ein Objekt aus der Bausteinbibliothek in das aktuelle Bild einzufügen.

Siehe auch

CopyToClipboard-Methode (Seite 1812)
PasteClipboard-Methode (Seite 1855)
GetItemByPath-Methode (Seite 1833)
FindByDisplayName-Methode (Seite 1829)
Delete-Methode (Seite 1818)
AddItem-Methode (Seite 1793)
AddFromClipboard-Methode (Seite 1790)
AddFolder-Methode (Seite 1789)
SymbolLibrary-Objekt (Seite 2035)
SymbolLibraries-Objekt (Auflistung) (Seite 2036)
So fügen Sie ein Objekt aus der Bausteinbibliothek mit VBA in ein Bild ein (Seite 1654)
So bearbeiten Sie die Bausteinbibliothek mit VBA (Seite 1651)

So bearbeiten Sie die Bausteinbibliothek mit VBA

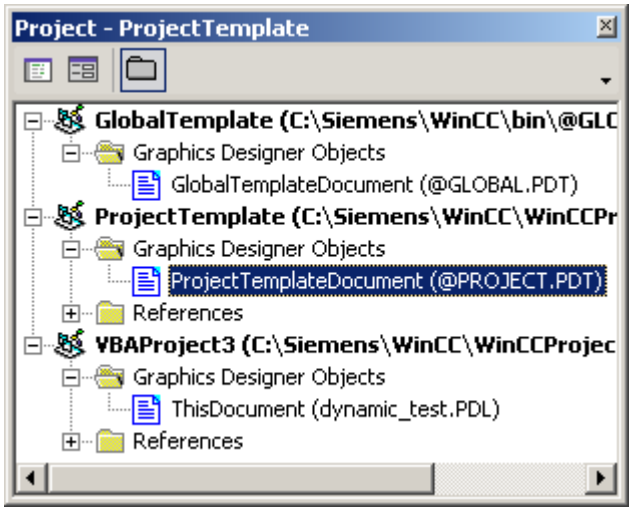
Einleitung

Sie finden hier folgende Anleitungen zum Bearbeiten der Bausteinbibliothek mit VBA:

- Neuen Ordner anlegen
- Objekt innerhalb der Bausteinbibliothek kopieren
- Objekt aus dem aktiven Bild in die Bausteinbibliothek kopieren
- Objekt in der Bausteinbibliothek löschen

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor").
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



3. Um einen neuen Ordner in der Bausteinbibliothek anzulegen, fügen Sie z.B. eine Prozedur "AddNewFolderToProjectLibrary()" in das Dokument ein. In diesem Beispiel wird der Ordner "mein Ordner" angelegt:

```
Sub AddNewFolderToProjectLibrary()
'VBA20
Dim objProjectLib As HMISymbolLibrary
Set objProjectLib = Application.SymbolLibraries(2)
'
'("AddFolder(DefaultName)"-Methode):
objProjectLib.FolderItems.AddFolder ("Custom Folder")
End Sub
```

4. Um ein Objekt von der "Globalen Bibliothek" in die "Projekt Bibliothek" zu kopieren, fügen Sie z.B. eine Prozedur "'CopyObjectFromGlobalLibraryToProjectLibrary()" in das Dokument ein. In diesem Beispiel wird das Objekt "Object1" kopiert:

```
Sub CopyObjectFromGlobalLibraryToProjectLibrary()
'VBA21
Dim objGlobalLib As HMISymbolLibrary
Dim objProjectLib As HMISymbolLibrary
Set objGlobalLib = Application.SymbolLibraries(1)
Set objProjectLib = Application.SymbolLibraries(2)
'
'Copies object "PC" from the "Global Library" into the clipboard
objGlobalLib.FolderItems("Folder2").Folder("Folder2").Folder.Item(
"Object1").CopyToClipboard
'
'The folder "Custom Folder" has to be available
objProjectLib.FolderItems("Folder1").Folder.AddFromClipBoard
("Copy of PC/PLC")
End Sub
```

5. Um ein Objekt aus dem aktiven Bild in die "Projekt Bibliothek" zu kopieren, fügen Sie z.B. eine Prozedur "AddObjectFromPictureToProjectLibrary()" in das Dokument ein. In diesem Beispiel wird das Objekt "Circle1" im aktiven Bild angelegt und dann in den Ordner "Folder1" kopiert:

```
Sub AddObjectFromPictureToProjectLibrary()
'VBA22
Dim objProjectLib As HMISymbolLibrary
Dim objCircle As HMICircle
Set objProjectLib = Application.SymbolLibraries(2)
'
'Insert new object "Circle1"
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle1",
"HMICircle")
'
'The folder "Custom Folder" has to be available
'("AddItem(DefaultName, pHMIObject)"-Methode):
objProjectLib.FolderItems("Folder1").Folder.AddItem "ProjectLib
Circle", ActiveDocument.HMIObjects("Circle1")
End Sub
```

6. Um ein Objekt aus der Bausteinbibliothek zu löschen, fügen Sie z.B. eine Prozedur "DeleteObjectFromProjectLibrary()" in das Dokument ein. In diesem Beispiel wird der zuvor angelegte Ordner "Folder1" gelöscht:

```
Sub DeleteObjectFromProjectLibrary()
'VBA23
Dim objProjectLib As HMISymbolLibrary
Set objProjectLib = Application.SymbolLibraries(2)
'
'The folder "Custom Folder" has to be available
'("Delete"-Methode):
objProjectLib.FolderItems("Folder1").Delete
End Sub
```

7. Starten Sie die Prozeduren jeweils mit <F5>.

Siehe auch

SymbolLibrary-Objekt (Seite 2035)

SymbolLibraries-Objekt (Auflistung) (Seite 2036)

PasteClipboard-Methode (Seite 1855)

Delete-Methode (Seite 1818)

CopyToClipboard-Methode (Seite 1812)

AddItem-Methode (Seite 1793)

AddFromClipboard-Methode (Seite 1790)

AddFolder-Methode (Seite 1789)

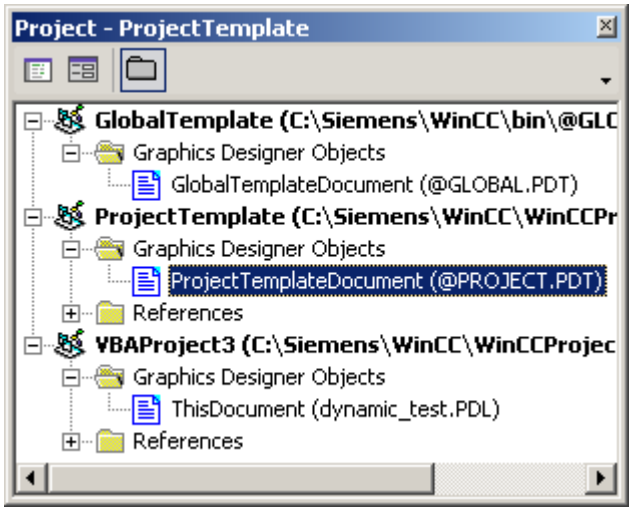
So fügen Sie ein Objekt aus der Bausteinbibliothek mit VBA in ein Bild ein (Seite 1654)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

So fügen Sie ein Objekt aus der Bausteinbibliothek mit VBA in ein Bild ein

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor").
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



3. Um ein Objekt aus der "Globalen Bibliothek" in das aktive Bild einzufügen, fügen Sie z.B. eine Prozedur "CopyObjectFromGlobalLibraryToActiveDocument()" in das Dokument ein. In diesem Beispiel wird das Objekt "Object1" in das aktive Bild eingefügt:

```
Sub CopyObjectFromGlobalLibraryToActiveDocument()
'VBA24
Dim objGlobalLib As HMISymbolLibrary
Dim objHMIObject As HMIObject
Dim iLastObject As Integer
Set objGlobalLib = Application.SymbolLibraries(1)
'
'Copy object "PC" from "Global Library" to clipboard
objGlobalLib.FolderItems("Folder2").Folder("Folder2").Folder.Item(
"Object1").CopyToClipboard
'
'Get object from clipboard and add it to active document
ActiveDocument.PasteClipboard
'
'Get last inserted object
iLastObject = ActiveDocument.HMIObjects.Count
Set objHMIObject = ActiveDocument.HMIObjects(iLastObject)
'
'Set position of the object:
With objHMIObject
.Left = 40
.Top = 40
End With
End Sub
```

4. Starten Sie die Prozedur mit <F5>.

Siehe auch

PasteClipboard-Methode (Seite 1855)

CopyToClipboard-Methode (Seite 1812)

So bearbeiten Sie die Bausteinbibliothek mit VBA (Seite 1651)

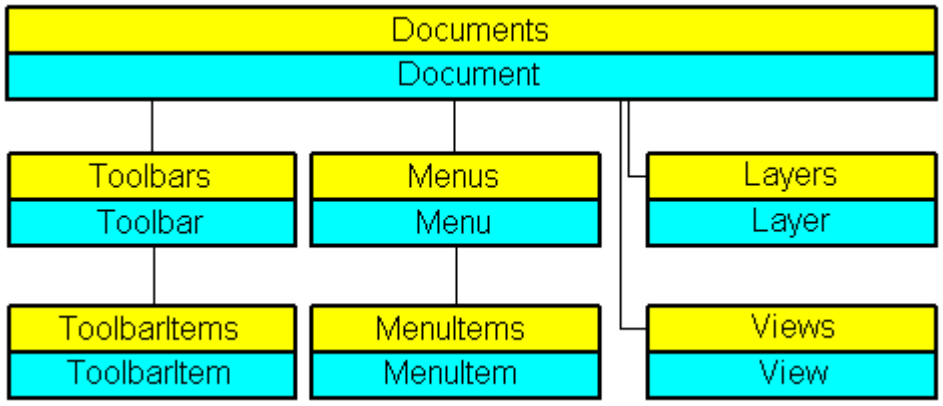
Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

3.3.3 Bilder mit VBA bearbeiten

3.3.3.1 Bilder mit VBA bearbeiten

Einleitung

Bilder visualisieren den zu bedienenden und beobachtenden Prozess. Sie zeigen die wichtigen Prozess-Schritte oder Anlagenteile und stellen schematisch den Produktionsablauf dar. In VBA wird das Bild durch das Document-Objekt dargestellt.



Bildspezifische Menüs und Symbolleisten

Im Gegensatz zu den anwendungsspezifischen Menüs und Symbolleisten sind die bildspezifischen Menüs und Symbolleisten an ein bestimmtes Bild gekoppelt. Die bildspezifischen Menüs und Symbolleisten sind solange sichtbar, wie das Bild aktiv ist.

Verwenden Sie bildspezifische Menüs und Symbolleisten, wenn die aufgerufenen VBA-Makros nur in dem Bild verwendet werden.

Ebenen

Sie können mit VBA auf die Ebenen im Graphics Designer zugreifen. Die Ebene wird durch das Layer-Objekt repräsentiert. Mit den Eigenschaften des Layer-Objektes legen Sie unter anderem den Ebenennamen und die Zoomeinstellungen fest.

Die Sichtbarkeit der RT Ebenen steuern Sie über das Document-Objekt. Die Sichtbarkeit der CS Ebenen steuern Sie über das View-Objekt.

Kopien des Bildes

Sie können mit VBA Kopien eines Bildes erzeugen, um ein Bild in verschiedenen Ansichten darzustellen. Die Kopie eines Bildes wird in VBA durch das View-Objekt repräsentiert.

Über die Eigenschaften des View-Objektes können Sie unter anderem den Zoomfaktor einstellen und festlegen, welcher Bildausschnitt angezeigt werden soll.

Hinweis

Wenn Sie VBA-Code in einem Bild, das in WinCC V7.0 SP1 gespeichert wurde, mit WinCC V7.0 ausführen möchten, deaktivieren Sie im VBA-Editor unter "Tools > References" die "CCHMIDotNetObj 1.0 Type Library".

Das VBA-Programm wird dann mit dem in WinCC V7.0 üblichen Funktionsumfang ausgeführt. Die neuen Funktionen von WinCC V7.0 SP1 können Sie nicht nutzen.

Siehe auch

Kopie eines Bildes mit VBA bearbeiten (Seite 1660)

So legen Sie bildspezifische Menüs und Symbolleisten an (Seite 1657)

Ebenen mit VBA bearbeiten (Seite 1659)

Objekte mit VBA bearbeiten (Seite 1662)

Graphics Designer mit VBA anpassen (Seite 1625)

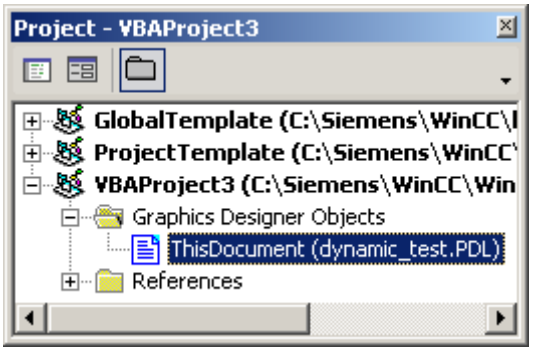
3.3.3.2 So legen Sie bildspezifische Menüs und Symbolleisten an

Einleitung

Bildspezifische Menüs und Symbolleisten sind an ein bestimmtes Bild gebunden und bleiben solange sichtbar, wie das Bild aktiv ist. Verwenden Sie bildspezifische Menüs und Symbolleisten, wenn die dort verwendeten VBA-Makros nur für das Bild relevant sind.

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor").
2. Öffnen Sie im Project Explorer das Dokument "ThisDocument":



3. Um ein bildspezifisches Menü anzulegen, fügen Sie z.B. eine Prozedur "CreateDocumentMenus()" in das Dokument "ThisDocument" ein:

```
Sub CreateDocumentMenus()
'VBA25
'Declare menuobjects:
Dim objMenu1 As HMIMenu
Dim objMenu2 As HMIMenu
'Insert Menus ("InsertMenu"-Methode) with
'Parameters - "Position", "Key", "DefaultLabel":
Set objMenu1 = ActiveDocument.CustomMenus.InsertMenu(1,
"DocMenu1", "Doc_Menu_1")
Set objMenu2 = ActiveDocument.CustomMenus.InsertMenu(2,
"DocMenu2", "Doc_Menu_2")
End Sub
```

4. Um eine bildspezifische Symbolleiste anzulegen, fügen Sie z.B. eine Prozedur "CreateDocumentToolbars()" in das Dokument "ThisDocument" ein:

```
Sub CreateDocumentToolbars()
'VBA26
'Declare required number of toolbarobjects:
Dim objToolbar1 As HMIToolbar
Dim objToolbar2 As HMIToolbar
'
'Insert toolbars ("Add"-Methode) with
'Parameter - "Key":
Set objToolbar1 = ActiveDocument.CustomToolbars.Add("DocToolbar1")
Set objToolbar2 = ActiveDocument.CustomToolbars.Add("DocToolbar2")
End Sub
```

5. Starten Sie die Prozeduren jeweils mit <F5>.

Siehe auch

Add-Methode (CustomToolbars-Auflistung) (Seite 1779)

InsertMenu-Methode (Seite 1836)

So weisen Sie Menüs und Symbolleisten VBA-Makros zu (Seite 1644)

So weisen Sie Menüs und Symbolleisten Hilfetexte zu (Seite 1641)

So fügen Sie ein neues Symbol zur Symbolleiste hinzu (Seite 1639)

So legen Sie Menüs mehrsprachig an (Seite 1635)

So fügen Sie einen neuen Menüeintrag zum Menü hinzu (Seite 1632)

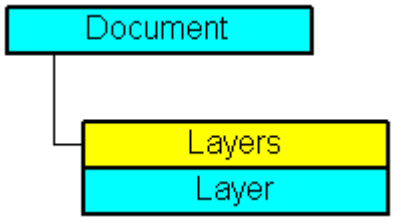
Eigene Menüs und Symbolleisten anlegen (Seite 1629)

Menüs und Symbolleisten konfigurieren (Seite 1628)

3.3.3.3 Ebenen mit VBA bearbeiten

Einleitung

Im Graphics Designer können Sie Objekte in 32 Ebenen anordnen. Die Ebenen werden nach CS- und RT-Ebenen unterschieden, damit die Sichtbarkeit der Ebenen im Bild (CS) und in Runtime (RT) getrennt gesteuert werden kann. In VBA wird eine Ebene durch das Layer-Objekt dargestellt:



Im Graphics Designer entspricht die unterste Ebene der "Ebene 0". Um die unterste Ebene mit VBA zurückzugeben, verwenden Sie den Index "1":

```
ActiveDocument.Layers(1)
```

Verwendung des Layer-Objektes

Sie verwenden das Layer-Objekt, um für eine Ebene den Minimal- und Maximalzoom festzulegen und einen Namen zu vergeben. Im folgenden Beispiel werden im aktiven Bild die Einstellungen der untersten Ebene konfiguriert:

```
Sub ConfigureSettingsOfLayer
'VBA27
Dim objLayer As HMILayer
Set objLayer = ActiveDocument.Layers(1)
With objLayer
'Configure "Layer 0"
.MinZoom = 10
.MaxZoom = 100
.Name = "Configured with VBA"
End With
End Sub
```

Sichtbarkeit von CS- und RT-Ebenen steuern

Die Sichtbarkeit der CS Ebenen steuern Sie über das View-Objekt. Um festzulegen, welche Ebenen in Runtime ein- oder ausgeblendet sein sollen, verwenden Sie das Document-Objekt. Mit folgenden Methoden können Sie die Sichtbarkeit der CS- und RT-Ebenen steuern:

- Methode "IsCSLayerVisible(Index)": Prüft, ob die angegebene CS-Ebene einblendet ist.
- Methode "SetCSLayerVisible(Index, Val)": Blendet die angegebene CS-Ebene ein oder aus.

Verwenden Sie analog die Methoden IsRTLayerVisible und SetRTLayerVisible für die RT-Ebenen.

Siehe auch

IsRTLayerVisible-Methode (Seite 1844)

SetRTLayerVisible-Methode (Seite 1873)

SetCSLayerVisible-Methode (Seite 1870)

IsCSLayerVisible-Methode (Seite 1843)

Layers-Objekt (Auflistung) (Seite 1969)

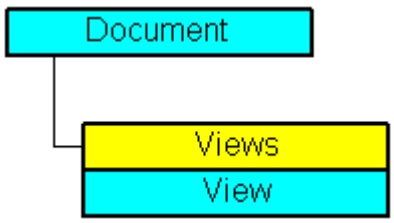
Bilder mit VBA bearbeiten (Seite 1656)

Sprachabhängige Projektierung mit VBA (Seite 1626)

3.3.3.4 Kopie eines Bildes mit VBA bearbeiten

Einleitung

Sie können mit VBA Kopien eines Bildes erzeugen, um ein Bild in verschiedenen Ansichten darzustellen. Jede Ansicht wird in einem eigenen Fenster dargestellt. Die Kopie eines Bildes wird in VBA durch das View-Objekt repräsentiert:



Über die Eigenschaften des View-Objektes können Sie unter anderem den Zoomfaktor einstellen und festlegen, welcher Bildausschnitt angezeigt werden soll.

Kopie eines Bildes erzeugen

Verwenden Sie die Add-Methode, um vom angegebenen Bild eine Kopie zu erzeugen. In diesem Beispiel wird vom aktiven Bild eine Kopie erzeugt und aktiviert:

```
Sub CreateAndActivateView()
'VBA28
Dim objView As HMIView
Set objView = ActiveDocument.Views.Add
objView.Activate
End Sub
```

Kopie eines Bildes bearbeiten

Jede Kopie eines Bildes können Sie wie folgt bearbeiten:

- Zoom-Faktor einstellen: Verwenden Sie die Zoom-Eigenschaft.
- Bildausschnitt festlegen: Verwenden Sie die Eigenschaften "ScrollPosX" und "ScrollPosY", um den Bildausschnitt anhand der Bildlaufleisten festzulegen.
- CS-Ebenen ein- und ausblenden: Verwenden Sie z.B. die SetCSLayerVisible(Index)-Methode, um den angebenen Layer ein- oder auszublenden. Mit der ActiveLayer-Eigenschaft können Sie die Ebene auswählen, deren Objekte Sie bearbeiten möchten.

Im folgenden Beispiel wird vom aktiven Bild eine Kopie erzeugt und aktiviert. Der Zoomfaktor wird auf 150% gesetzt und die Position der Bildlaufleisten verändert:

```
Sub SetZoomAndScrollPositionInActiveView()
'VBA29
Dim objView As HMIView
Set objView = ActiveDocument.Views.Add
With objView
.Activate
.ScrollPosX = 40
.ScrollPosY = 10
.Zoom = 150
End With
End Sub
```

Siehe auch

Add-Methode (Views-Auflistung) (Seite 1784)

ScrollPosY-Eigenschaft (Seite 2361)

ScrollPosX-Eigenschaft (Seite 2360)

ActiveLayer-Eigenschaft (Seite 2068)

View-Objekt (Seite 2062)

SetCSLayerVisible-Methode (Seite 1870)

IsCSLayerVisible-Methode (Seite 1843)

Activate Methode (Seite 1776)

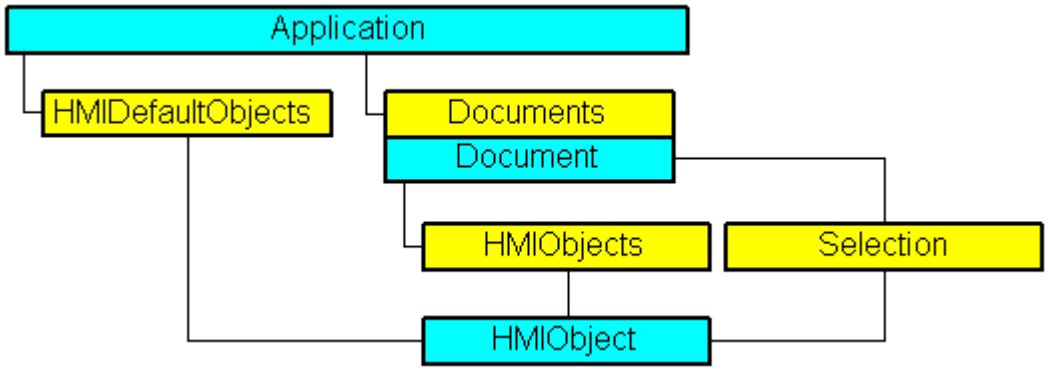
Ebenen mit VBA bearbeiten (Seite 1659)

Bilder mit VBA bearbeiten (Seite 1656)

3.3.4 Objekte mit VBA bearbeiten

3.3.4.1 Objekte mit VBA bearbeiten

Zugriff auf Objekte im Graphics Designer



In VBA sind alle Objekttypen des aktuellen Bildes in der Auflistung "HMIObjects" enthalten. Sie sind nicht wie im Graphics Designer nach Objekttypen (Standard-, Smart-, Windows- und Controls-Objekte) getrennt. Deshalb können Sie mit VBA alle Objekte in einem oder mehreren Bildern mit einer Schleife durchlaufen.

Wenn Sie Objekte im Bild ausgewählt haben, sind diese Objekte in der Auflistung "Selection" enthalten. Verwenden Sie die Auflistung "HMIDefaultObjects", wenn Sie die Vorbelegungen der Eigenschaftswerte eines Objektes ändern wollen.

Um ein Objekt im Bild mit VBA anzusprechen, verwenden Sie entweder den Objektnamen, z.B. "ActiveDocument.HMIObjects("Kreis1"), oder die Indexnummer. "ActiveDocument.HMIObjects(1)" referenziert z.B. das erste Objekt im aktiven Bild.

Bearbeiten von Objekten mit VBA

Sie haben folgende Möglichkeiten, Objekte mit VBA zu bearbeiten:

- Neues Objekt in einem Bild erzeugen
- Vorhandenes Objekt löschen
- Vorhandene Objekte kopieren
- Vorhandene Objekte gruppieren und die Gruppierung wieder aufheben
- Nach Objekten suchen
- Objekteigenschaften anzeigen oder ändern

Wenn Sie ein neues Objekt mit VBA in ein Bild einfügen, verhält sich das Objekt so, als wenn Sie es in der Objektpalette des Graphics Designer doppelklicken würden.

Das Objekt erhält die voreingestellten Eigenschaftswerte und wird in der linken oberen Ecke des Bildes eingefügt.

Der Zugriff auf die Objekteigenschaften ist davon abhängig, wie Sie das Objekt erzeugt haben. Dazu zwei Beispiele:

Beispiel 1:

In diesem Beispiel wird ein Kreis vom Typ "HMIObject" in das aktuelle Bild eingefügt. Ein VBA-Objekt vom Typ "HMIObject" können Sie für alle Objekte im Graphics Designer verwenden. Sie müssen jedoch individuelle Eigenschaften des jeweiligen Objektes explizit über die Eigenschaft "Properties(Index)" ansprechen:

```
Sub AddObject()
'VBA30
Dim objObject As HMIObject
Set objObject = ActiveDocument.HMIObjects.AddHMIObject("CircleAsHMIObject", "HMICircle")
'
'standard-properties (e.g. the position) are available every time:
objObject.Top = 40
objObject.Left = 40
'
'non-standard properties can be accessed using the Properties-collection:
objObject.Properties("FlashBackColor") = True
End Sub
```

Beispiel 2:

In diesem Beispiel wird ein Kreis vom Typ "HMICircle" in das aktuelle Bild eingefügt. Im Gegensatz zu Beispiel 1 können Sie das Objekt "objCircle" aber nur für Objekte vom Typ "HMICircle" verwenden:

```
Sub AddCircle()
'VBA31
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("CircleAsHMICircle", "HMICircle")
'
'The same as in example 1, but here you can set/get direct the
'specific properties of the circle:
objCircle.Top = 80
objCircle.Left = 80
objCircle.FlashBackColor = True
End Sub
```

Siehe auch

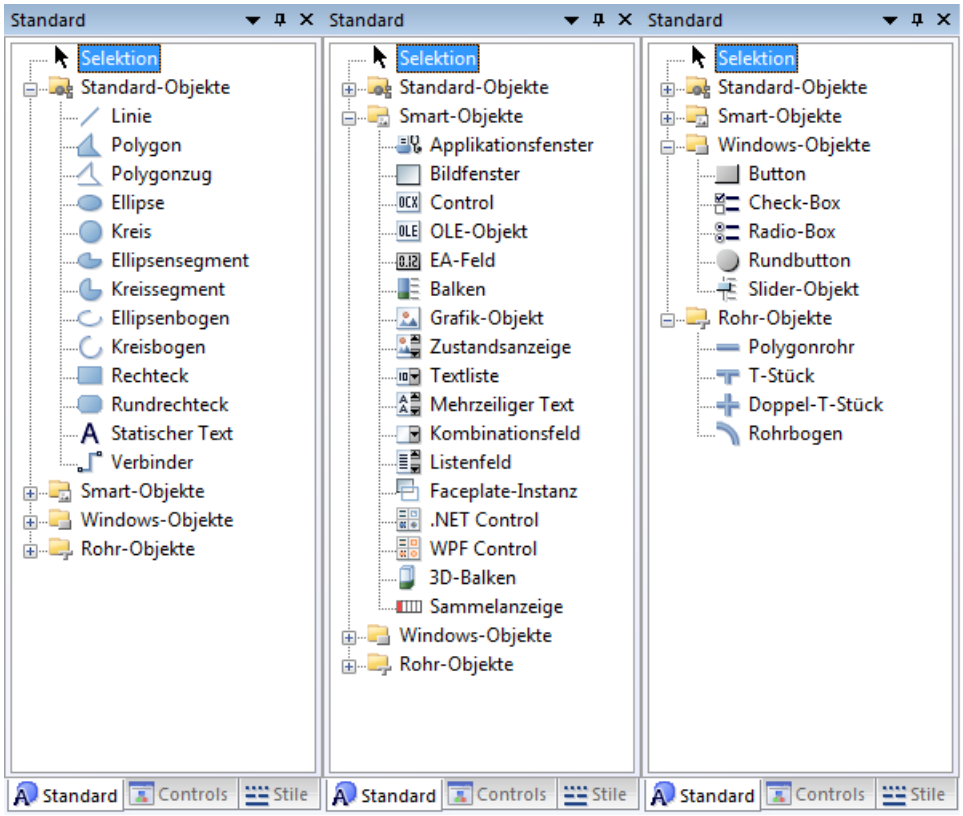
LanguageFonts-Objekt (Auflistung) (Seite 1963)
VBA-Referenz (Seite 1735)
Underlined-Eigenschaft (Seite 2402)
Size-Eigenschaft (Seite 2368)
Parent-Eigenschaft (Seite 2317)
LanguageID-Eigenschaft (Seite 2232)
Italic-Eigenschaft (Seite 2224)
FontFamily-Eigenschaft (Seite 2201)
Bold-Eigenschaft (Seite 2106)
Application-Eigenschaft (Seite 2079)

3.3.4.2 Standard-, Smart-, Windows- und Rohr-Objekte

Standard-, Smart-, Windows- und Rohr-Objekte

Einleitung

Die Standard-, Smart- und Windows-Objekte verwenden Sie, um Ihre Bilder zu gestalten. Im Graphics Designer finden Sie diese Objekte in der Objektpalette auf der Registerkarte "Standard":



VBA bietet Ihnen die Möglichkeit, auf diese Objekte in allen Bildern Ihres Projektes zuzugreifen. Wenn Sie in einem Projekt mit mehreren Bildern z.B. die Hintergrundfarbe aller Kreise ändern wollen, dann können Sie das mit einem VBA-Makro erledigen.

Objekt in Bild einfügen

Verwenden Sie die Methode "AddHMIObject(ObjectName, ProgID)", um ein neues Objekt in ein Bild einzufügen. "ObjectName" steht für den Namen des Objektes (z.B. "my Circle"), "ProgID" für die VBA-Objektbezeichnung (z.B. "HMICircle"):

```
Sub AddCircle()
'VBA32
```

```
'Creates object of type "HMICircle"
Dim objCircle As HMICircle
'
'Add object in active document
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("My Circle", "HMICircle")
End Sub
```

Objekt bearbeiten

Sie haben mit VBA auf alle Objekteigenschaften Zugriff, die Sie über den Eigenschaftendialog des Objektes bearbeiten. Sie können Objekteigenschaften ändern und ausgeben sowie Objekte im Bild auswählen. Wenn Sie kein Objekt ausgewählt haben, können Sie folgende Methoden anwenden:

- "Find()"-Methode: Sucht in der Auflistung "HMIObjects" nach einem Objekt
- "Delete()"-Methode: Löscht ein HMIObject-Objekt

Wenn Sie Objekte ausgewählt haben, können Sie diese über die Auflistung "Selection" unter anderem mit folgenden Methoden bearbeiten:

- "AlignLeft()", "AlignRight()", "AlignTop()", "AlignBottom()": Diese Methoden richten Objekte aus.
- "CreateGroup()", "CreateCustomizedObject()": Diese Methoden erzeugen ein Gruppen- oder Anwender-Objekt.
- "DeselectAll()"-Methode: Hebt die Auswahl aller Objekte auf

Referenzen auf Objekte mit "Nothing" entfernen

Entfernen Sie immer die verwendeten Referenzen für die Controls und Standardobjekte sowie für das Dokument nach dem Schließen des Dokuments. Setzen Sie dazu die Objekte auf "Nothing". Nachfolgend ein Code-Beispiel für ein Control:

```
Public Sub DrawNewControl
   Const strFct = "CreatePdls"
   Dim objControl As HMIObject
   Dim objDoc As Document
   On Local Error GoTo errorhandler
   'open the document
   Set objDoc =
grafexe.Application.Documents.Open(grafexe.Application.ApplicationDa
taPath & "PDL1.pdl", hmiOpenDocumentTypeInvisible)
    'create new object
   Set objControl = objDoc.HMIObjects.AddActiveXControl("Control1",
"CCAxUserArchiveControl.AxUserArchiveControl.1")
   If objControl Is Nothing Then
      GoTo errorhandler
   End If
   'doing something with the control
   '......
      'delete reference to new control
   Set objControl = Nothing
```

```
   'saving PDL und deleting reference to it
   objDoc.Save
   objDoc.Close
   Set objDoc = Nothing
   Exit Sub
' errorhandler
errorhandler:
   If MsgBox("Error ocurred" & vbNewLine & "Yes - resume next" &
vbNewLine & "No  - stop script", vbOKCancel + vbCritical, strFct) =
vbOK Then
      Resume Next
   End If
End Sub
```

Siehe auch

Parent-Eigenschaft (Seite 2317)

Item-Eigenschaft (Seite 2225)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

So bearbeiten Sie Standard-, Smart- Windows- und Rohr-Objekte

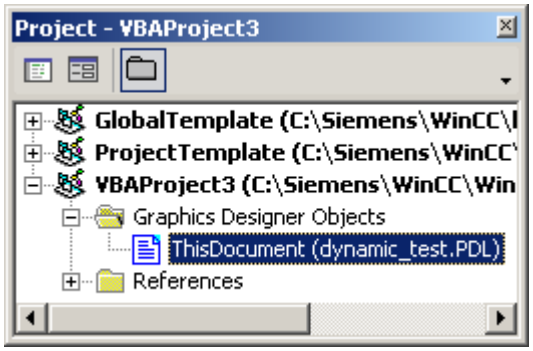
Einleitung

Sie finden hier folgende Anleitungen zum Bearbeiten von Standard-, Smart- und Windows-Objekten:

- Eigenschaften eines bestimmten Objektes definieren
- Eigenschaften eines unbestimmten Objektes definieren
- Objekt im aktiven Bild auswählen
- Objekte im aktiven Bild suchen
- Objekt löschen

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor").
2. Öffnen Sie im Project Explorer das Dokument "ThisDocument":



3. Um die Eigenschaften eines bestimmten Objekttyps (z.B. "HMICircle") zu definieren, fügen Sie z.B. eine Prozedur "EditDefinedObjectType()" in das Dokument "ThisDocument" ein. In diesem Beispiel wird ein Kreis in das aktive Bild eingefügt und dessen Linienbreite und -farbe geändert:

```
Sub EditDefinedObjectType()
'VBA33
Dim objCircle As HMICircle
Set objCircle =
ActiveDocument.HMIObjects.AddHMIObject("myCircleAsCircle",
"HMICircle")
With objCircle
'direct calling of objectproperties available
.BorderWidth = 4
.BorderColor = RGB(255, 0, 255)
End With
End Sub
```

4. Um die Eigenschaften eines unbestimmten Objekttyps ("HMIObject") zu ändern, fügen Sie z.B. eine Prozedur "EditHMIObject()" in das Dokument "ThisDocument" ein. In diesem Beispiel wird ein Kreis in das aktive Bild eingefügt und dessen Linienbreite und -farbe geändert:

```
Sub EditHMIObject()
'VBA34
Dim objObject As HMIObject
Set objObject =
ActiveDocument.HMIObjects.AddHMIObject("myCircleAsObject",
"HMICircle")
With objObject
'Access to objectproperties only with property "Properties":
.Properties("BorderWidth") = 4
.Properties("BorderColor") = RGB(255, 0, 0)
End With
End Sub
```

5. Um ein Objekt im aktuellen Bild auszuwählen, fügen Sie z.B. eine Prozedur "SelectObject()" in das Dokument "ThisDocument" ein. In diesem Beispiel wird ein Kreis in das aktive Bild eingefügt und ausgewählt:

```
Sub SelectObject()
'VBA35
Dim objObject As HMIObject
Set objObject =
ActiveDocument.HMIObjects.AddHMIObject("mySelectedCircle",
"HMICircle")
ActiveDocument.HMIObjects("mySelectedCircle").Selected = True
End Sub
```

6. Um nach einem Objekt im aktuellen Bild zu suchen, fügen Sie z.B. eine Prozeduren "FindObjectsByName()", "FindObjectsByType()" oder "FindObjectsByProperty()" in das Dokument "ThisDocument" ein. In diesem Beispiel wird nach Objekten gesucht, welche die Zeichenkette "Circle" im Objektnamen enthalten:

```
Sub FindObjectsByName()
'VBA36
Dim colSearchResults As HMICollection
Dim objMember As HMIObject
Dim iResult As Integer
Dim strName As String
'
'Wildcards (?, *) are allowed
Set colSearchResults =
ActiveDocument.HMIObjects.Find(ObjectName:="*Circle*")
For Each objMember In colSearchResults
iResult = colSearchResults.Count
strName = objMember.ObjectName
MsgBox "Found: " & CStr(iResult) & vbCrLf & "Objectname: " &
strName
Next objMember
End Sub
```

In diesem Beispiel wird im aktiven Bild nach Objekten vom Typ "HMICircle" gesucht:

```
Sub FindObjectsByType()
'VBA37
Dim colSearchResults As HMICollection
Dim objMember As HMIObject
Dim iResult As Integer
Dim strName As String
Set colSearchResults =
ActiveDocument.HMIObjects.Find(ObjectType:="HMICircle")
For Each objMember In colSearchResults
iResult = colSearchResults.Count
strName = objMember.ObjectName
MsgBox "Found: " & CStr(iResult) & vbCrLf & "Objektname: " &
strName
Next objMember
End Sub
```

In diesem Beispiel wird im aktiven Bild nach Objekten mit der Eigenschaft "BackColor" gesucht:

```
Sub FindObjectsByProperty()
'VBA38
Dim colSearchResults As HMICollection
Dim objMember As HMIObject
Dim iResult As Integer
Dim strName As String
Set colSearchResults =
ActiveDocument.HMIObjects.Find(PropertyName:="BackColor")
For Each objMember In colSearchResults
iResult = colSearchResults.Count
strName = objMember.ObjectName
MsgBox "Found: " & CStr(iResult) & vbCrLf & "Objectname: " &
strName
```

```
Next objMember
End Sub
```

7. Um ein Objekt zu löschen, fügen Sie z.B. eine Prozedur "DeleteObject()" in das Dokument "ThisDocument" ein. In diesem Beispiel wird das erste Objekt im aktiven Bild gelöscht.

```
Sub DeleteObject()
'VBA39
'Delete first object in active document:
ActiveDocument.HMIObjects(1).Delete
End Sub
```

8. Starten Sie die Prozeduren jeweils mit <F5>.

Siehe auch

Find-Methode (Seite 1827)

Delete-Methode (Seite 1818)

AddHMIObject-Methode (Seite 1791)

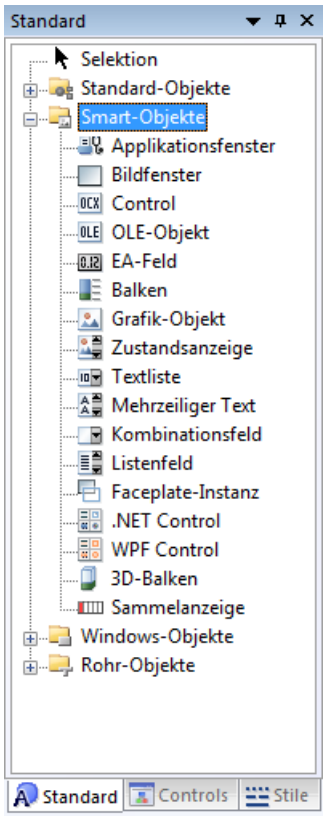
So bearbeiten Sie Standard-, Smart- Windows- und Rohr-Objekte (Seite 1667)

Objekte mit VBA bearbeiten (Seite 1662)

OLE-Objekte

Einleitung

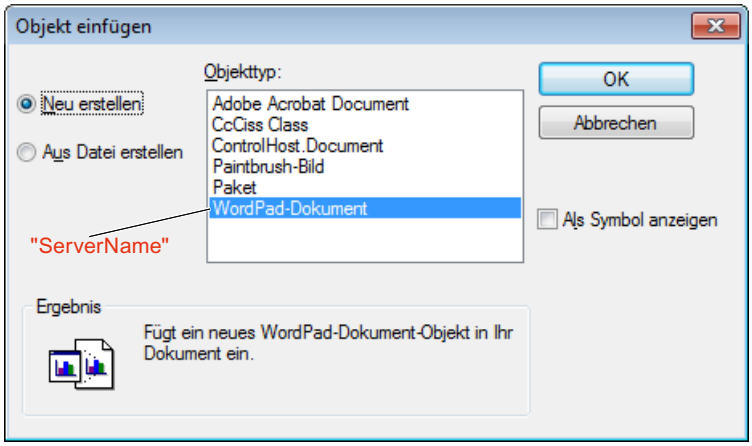
Mit VBA können Sie OLE-Objekte in ein Bild einfügen. Das OLE-Objekt gehört zu den Smart-Objekten. Im Graphics Designer finden Sie es in der Objektpalette auf der Registerkarte "Standard":



OLE-Objekt in Bild einfügen

Verwenden Sie die Methode "AddOLEControl(ObjectName, ServerName, [CreationType], [UseSymbol]), um ein OLE-Objekt in ein Bild einzufügen. "ObjectName" steht dabei für den Objektnamen, "ServerName" für die Anwendung, die im OLE-Objekt enthalten sein soll. Der Parameter "ServerName" entspricht dem "Objekttyp" im Dialog "Objekt einfügen" Die letzten

beiden Parameter sind optional und bilden die Einstellungsmöglichkeiten im gezeigten Dialog ab:



Weiterführende Informationen zu den Parametern finden Sie in dieser Dokumentation unter "AddOLEObject-Methode".

Im folgenden Beispiel wird ein OLE-Objekt, das ein Wordpad-Dokument enthält, in das aktive Bild eingefügt:

```
Sub AddOLEObjectToActiveDocument()
'VBA40
Dim objOLEObject As HMIOLEObject
Set objOLEObject = ActiveDocument.HMIObjects.AddOLEObject("MS Wordpad Document1",
"Wordpad.Document.1")
End Sub
```

Das OLEObject-Objekt wird als letztes Element der Auflistung "HMIObjects" hinzugefügt und erbt die Eigenschaften des HMIObject-Objektes.

Siehe auch

OLEObject-Objekt (Seite 1987)

AddOLEObject-Methode (Seite 1794)

So bearbeiten Sie Standard-, Smart- Windows- und Rohr-Objekte (Seite 1667)

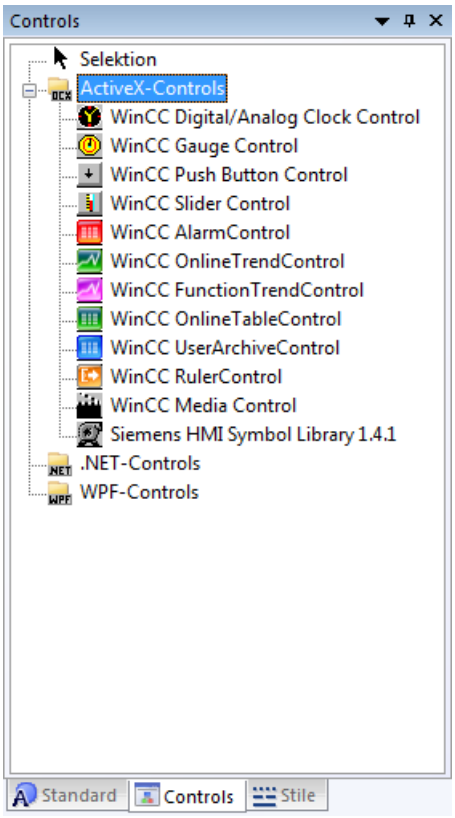
Standard-, Smart-, Windows- und Rohr-Objekte (Seite 1665)

Objekte mit VBA bearbeiten (Seite 1662)

ActiveX Controls

Einleitung

Mit VBA könnnen Sie ActiveX Controls in ein Bild einfügen. Im Graphics Designer finden Sie die von WinCC mitgelieferten ActiveX Controls in der Objektpalette auf der Registerkarte Controls:



Weitere Informationen finden Sie unter "AddActiveXControl-Methode" in dieser Dokumentation und unter "Objektpalette" in der WinCC-Dokumentation.

Standard-ActiveX Controls einbinden

Neben den von WinCC mitgelieferten ActiveX Controls können Sie alle im Betriebssystem registrierten Standard-ActiveX Controls in ein Bild einfügen. Damit haben Sie auch die Möglichkeit, selbst programmierte ActiveX Controls in Ihren Bildern zu verwenden. Eine Liste der mit WinCC getesteten Standard-ActiveX Controls finden Sie in der WinCC-Dokumentation.

ActiveX Control in Bild einfügen

Verwenden Sie die Methode "AddActiveXControl(ObjectName, ProgID)", um ein neues ActiveX Control in ein Bild einzufügen. "ObjectName" steht für den Namen des ActiveX Control (z.B. "WinCC Gauge"), "ProgID" für die VBA-Objektbezeichnung (z.B. "XGauge.XGauge.1"):

```
Sub AddActiveXControl()
'VBA41
Dim objActiveXControl As HMIActiveXControl
Set objActiveXControl = ActiveDocument.HMIObjects.AddActiveXControl("WinCC_Gauge",
"XGAUGE.XGaugeCtrl.1")
End Sub
```

Das ActiveXControl-Objekt wird als letztes Element der Auflistung "HMIObjects" hinzugefügt und erbt die Eigenschaften des HMIObject-Objektes.

Zugriff auf die Eigenschaften des ActiveX Control

Die objektspezifischen Eigenschaften des ActiveX Control müssen Sie über die Eigenschaft "Properties(Index)" ansprechen. Welche Eigenschaften ein ActiveX Control besitzt, erfahren Sie über den Dialog "Objekteigenschaften" im Graphics Designer oder über die Properties-Auflistung:

```
Sub AddActiveXControl()
'VBA42
Dim objActiveXControl As HMIActiveXControl
Set objActiveXControl = ActiveDocument.HMIObjects.AddActiveXControl("WinCC_Gauge2",
"XGAUGE.XGaugeCtrl.1")
'
'move ActiveX-control:
objActiveXControl.Top = 40
objActiveXControl.Left = 60
'
'Change individual property:
objActiveXControl.Properties("BackColor").value = RGB(255, 0, 0)
End Sub
```

Eingeschränkter Zugriff auf Hintergrundgrafiken von Controls

Bei den folgenden Controls kann die Hintergrundgrafik nicht mit VBA konfiguriert werden:

Control	Attribut
WinCC Digital/Analog Clock Control	Hintergrundgrafik
WinCC Gauge Control	Hintergrundbild Rahmengrafik
WinCC Push Button Control	PictureSelected PictureUnselected
WinCC Slider Control	Hintergrundbild Schieberbild

Siehe auch

ActiveXControl-Objekt (Seite 1885)

AddActiveXControl-Methode (Seite 1785)

So bearbeiten Sie Standard-, Smart- Windows- und Rohr-Objekte (Seite 1667)

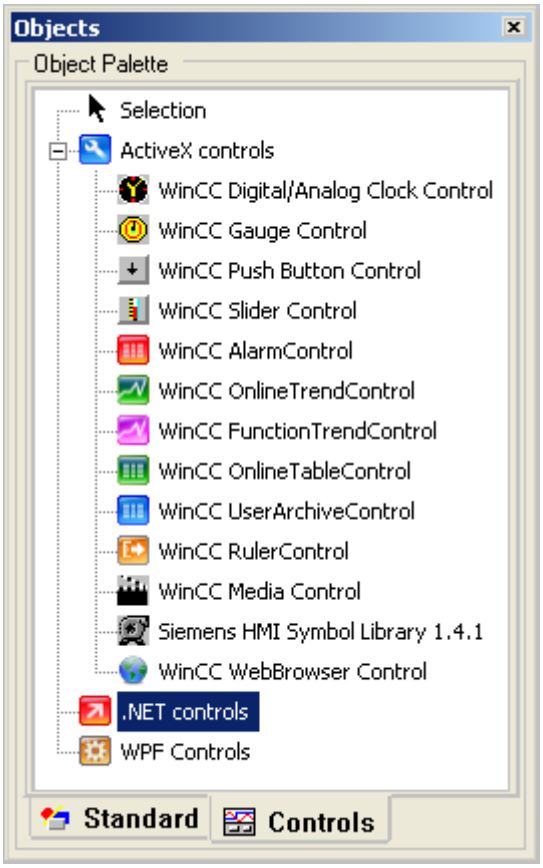
Standard-, Smart-, Windows- und Rohr-Objekte (Seite 1665)

Objekte mit VBA bearbeiten (Seite 1662)

.Net-Controls

Einleitung

Mit VBA könnnen Sie .Net-Controls in ein Bild einfügen. Im Graphics Designer finden Sie die .Net-Controls in der Objektpalette auf der Registerkarte "Controls":



Weitere Informationen finden Sie unter "AddDotNetControl-Methode" in dieser Dokumentation und unter "Prozessbilder erstellen > Arbeiten mit Controls > .Net-Controls" in der WinCC-Dokumentation.

.Net-Control in Bild einfügen

Verwenden Sie die Methode "AddDotNetControl(ObjectName, ControlType, InGAC, AssemblyInfo)", um ein neues .Net-Control in ein Bild einzufügen. "ObjectName" steht für den Namen des .Net-Control. "ControlType" zeigt den Namensraum des Objekts. Wenn "InGAC" den Wert "TRUE" hat, ist das Objekt im Global Assembly Cache registriert und über "AssemblyInfo" werden die dazugehörenden Informationen angegeben.

```
Sub AddDotNetControl()
'VBA851
Dim DotNetControl As HMIDotNetControl
Set DotNetControl = ActiveDocument.HMIObjects.AddDotNetControl("MyVBAControl",
"System.Windows.Forms.Label", True,"Assembly=System.Windows.Forms, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089")
End Sub
```

Das .Net-Control-Objekt wird als Element der Auflistung "HMIObjects" hinzugefügt und erbt die Eigenschaften des HMIObject-Objektes.

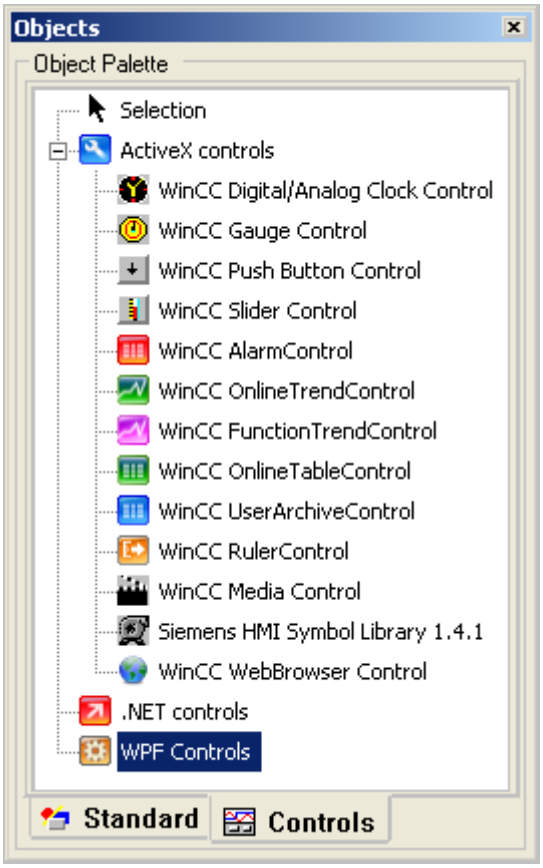
Zugriff auf die Eigenschaften des .Net-Control

Welche Eigenschaften ein .Net-Control besitzt, erfahren Sie über den Dialog "Objekteigenschaften > Control-Eigenschaften" im Graphics Designer.

WPF-Controls

Einleitung

Mit VBA könnnen Sie .Net-Controls in ein Bild einfügen. Im Graphics Designer finden Sie die .Net-Controls in der Objektpalette auf der Registerkarte "Controls":



Weitere Informationen finden Sie unter "AddWPFControl-Methode" in dieser Dokumentation und unter "Prozessbilder erstellen > Arbeiten mit Controls > WPF-Controls" in der WinCC-Dokumentation.

WPF-Control in Bild einfügen

Verwenden Sie die Methode "AddWPFControl(ObjectName, ControlType, InGAC, AssemblyInfo)", um ein neues WPF-Control in ein Bild einzufügen. "ObjectName" steht für den Namen des .Net-Control. "ControlType" zeigt den Namensraum des Objekts. Wenn "InGAC" den Wert "TRUE" hat, ist das Objekt im Global Assembly Cache registriert und über "AssemblyInfo" werden die dazugehörenden Informationen angegeben.

```
Sub AddWPFControl()
'VBA852
Dim WPFControl As HMIWPFControl
```

```
Set WPFControl = ActiveDocument.HMIObjects.AddWPFControl("MyWPFVBAControl",
"WinCCWPFControl.TestControl", False, "Assembly=Z:\TestControl\WinCCWPFControl.dll")
End Sub
```

Das ActiveXControl-Objekt wird als Element der Auflistung "HMIObjects" hinzugefügt und erbt die Eigenschaften des HMIObject-Objektes.

Zugriff auf die Eigenschaften des WPFControl

Welche Eigenschaften ein WPF-Control besitzt, erfahren Sie über den Dialog "Objekteigenschaften > Control-Eigenschaften" im Graphics Designer.

3.3.4.3 Gruppen-Objekte

Gruppen-Objekte

Einleitung

Mit VBA können Sie im Graphics Designer aus ausgewählten Objekten ein Gruppen-Objekt erzeugen. Sie können Objekte zum Gruppen-Objekt hinzufügen oder entfernen, ohne dass Sie das Gruppen-Objekt selbst auflösen müssen. Auf die Objekteigenschaften der einzelnen Objekte im Gruppen-Objekt haben Sie uneingeschränkten Zugriff. Ein Gruppen-Objekt können Sie auch wieder auflösen oder ganz löschen.

Folgende Objekttypen können nicht Bestandteil eines Gruppen-Objektes sein:

- CustomizedObject (Anwender-Objekt)
- ActiveXControl
- OLEObject

Weitere Informationen zum Gruppen-Objekt finden Sie in der WinCC-Dokumentation unter "Gruppen-Objekt".

Gruppen-Objekt erzeugen

Um ein Gruppen-Objekt zu erzeugen, wählen Sie die Objekte im Graphics Designer aus, die Teil des Gruppen-Objektes sein sollen. Die ausgewählten Objekte sind dann in der Auflistung "Selection" enthalten. Mit der Methode "CreateGroup()" erzeugen Sie die Gruppe:

```
Sub CreateGroup()
'VBA43
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objGroup As HMIGroup
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
```

```
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
Set objGroup = ActiveDocument.Selection.CreateGroup
objGroup.ObjectName = "myGroup"
End Sub
```

Das Gruppen-Objekt wird am Ende der Auflistung "HMIObjects" eingefügt. Die Objekte, die im Gruppen-Objekt enthalten sind, behalten ihre Indexnummern und sind weiterhin in der Auflistung "HMIObjects" verfügbar.

Die Objekte im Gruppen-Objekt sind zusätzlich in der Auflistung "GroupedHMIObjects" enthalten, wobei die Indexnummern neu vergeben werden.

Geben Sie dem Gruppen-Objekt einen Namen (objGroup.Name = "My Group"), damit Sie es eindeutig identifizieren können. Wenn Sie keinen Namen vergeben, erhält das Gruppen-Objekt die Standardbezeichnung für das Gruppen-Objekt (z.B. "Group1").

Das Gruppen-Objekt hat dieselben Eigenschaften wie das Objekte vom Typ "Object".

Gruppen-Objekt bearbeiten

Das Gruppen-Objekt können Sie wie folgt bearbeiten:

- Methode "Add(Index)": Fügt ein neues Objekt zum Gruppen-Objekt hinzu.
- Methode "Remove(Index)": Entfernt ein Objekt aus dem Gruppen-Objekt.
- Methode "UnGroup()": Löst das Gruppen-Objekt auf (Gruppierung aufheben).
- Methode "Delete()": Löscht das Gruppen-Objekt und die darin enthaltenen Objekte.

Objekte im Gruppen-Objekt bearbeiten

Verwenden Sie die Auflistung "GroupedHMIObjects", um ein Objekt im Gruppen-Objekt auszuwählen. Um auf dessen Objekteigenschaft zuzugreifen, müssen Sie den Namen der Objekteigenschaft über die Eigenschaft "Properties" ansprechen, z.B.:

```
Sub ModifyPropertyOfObjectInGroup()
'VBA44
Dim objGroup As HMIGroup
Set objGroup = ActiveDocument.HMIObjects("myGroup")
objGroup.GroupedHMIObjects(1).Properties("BorderColor") = RGB(255, 0, 0)
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

GroupedObjects-Objekt (Auflistung) (Seite 1950)

Ungroup-Methode (Seite 1877)

Remove-Methode (Seite 1857)

Delete-Methode (Seite 1818)

Add-Methode (GroupedObjects-Auflistung) (Seite 1781)

So bearbeiten Objekte im Gruppen-Objekt mit VBA (Seite 1684)

So bearbeiten Sie Gruppen-Objekte mit VBA (Seite 1681)

Objekte mit VBA bearbeiten (Seite 1662)

VBA im Graphics Designer (Seite 1623)

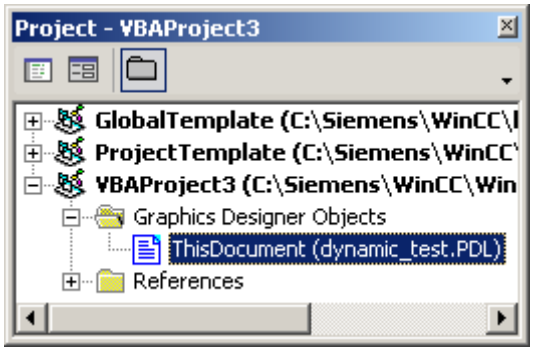
So bearbeiten Sie Gruppen-Objekte mit VBA

Voraussetzung

Sie müssen im Graphics Designer mindestens zwei Grafik-Objekte angelegt und diese ausgewählt haben.

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor").
2. Öffnen Sie im Project Explorer das Dokument "ThisDocument":



3. Um ein Gruppen-Objekt aus ausgewählten Objekten zu erzeugen, fügen Sie z.B. eine Prozedur "CreateGroup()" in das Dokument "ThisDocument" ein. In diesem Beispiel wird aus Objekten das Gruppen-Objekt "My Group" erzeugt:

```
Sub CreateGroup()
'VBA45
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objGroup As HMIGroup
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle",
"HMICircle")
Set objRectangle =
ActiveDocument.HMIObjects.AddHMIObject("sRectangle",
"HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
Set objGroup = ActiveDocument.Selection.CreateGroup
'The name identifies the group-object
objGroup.ObjectName = "My Group"
End Sub
```

4. Um ein Objekt zum Gruppen-Objekt "My Group" hinzuzufügen, fügen Sie z.B. eine Prozedur "AddObjectToGroup()" in das Dokument "ThisDocument" ein. In diesem Beispiel wird dem Gruppen-Objekt "My Group" eine Ellipse hinzugefügt:

```
Sub AddObjectToGroup()
'VBA46
Dim objGroup As HMIGroup
Dim objEllipseSegment As HMIEllipseSegment
'Adds new object to active document
Set objEllipseSegment =
ActiveDocument.HMIObjects.AddHMIObject("EllipseSegment",
"HMIEllipseSegment")
Set objGroup = ActiveDocument.HMIObjects("My Group")
'Adds the object to the group
objGroup.GroupedHMIObjects.Add ("EllipseSegment")
End Sub
```

5. Um ein Objekt aus dem Gruppen-Objekt "My Group" zu entfernen, fügen Sie z.B. eine Prozedur "RemoveObjectFromGroup()" in das Dokument "ThisDocument" ein. In diesem Beispiel wird aus dem Gruppen-Objekt "My Group" das erste Objekt entfernt:
```
Sub RemoveObjectFromGroup()
'VBA47
Dim objGroup As HMIGroup
Set objGroup = ActiveDocument.HMIObjects("My Group")
'delete first object of the group-object
objGroup.GroupedHMIObjects.Remove (1)
End Sub
```
6. Um das Gruppen-Objekt "My Group" wieder aufzulösen, fügen Sie z.B. eine Prozedur "UnGroup()" in das Dokument "ThisDocument" ein. In diesem Beispiel wird das Gruppen-Objekt "My Group" aufgelöst:
```
Sub UnGroup()
'VBA48
Dim objGroup As HMIGroup
Set objGroup = ActiveDocument.HMIObjects("My Group")
objGroup.UnGroup
End Sub
```
7. Um das Gruppen-Objekt "My Group" zu löschen, fügen Sie z.B. eine Prozedur "DeleteGroup()" in das Dokument "ThisDocument" ein. In diesem Beispiel wird das Gruppen-Objekt "My Group" mit den darin enthaltenen Objekten gelöscht:
```
Sub DeleteGroup()
'VBA49
Dim objGroup As HMIGroup
Set objGroup = ActiveDocument.HMIObjects("My Group")
objGroup.Delete
End Sub
```
8. Starten Sie die Prozeduren jeweils mit <F5>.

Siehe auch

Ungroup-Methode (Seite 1877)

Remove-Methode (Seite 1857)

Delete-Methode (Seite 1818)

CreateGroup-Methode (Seite 1816)

Add-Methode (GroupedObjects-Auflistung) (Seite 1781)

Selection-Objekt (Auflistung) (Seite 2022)

GroupedObjects-Objekt (Auflistung) (Seite 1950)

Group-Objekt (Seite 1946)

So bearbeiten Objekte im Gruppen-Objekt mit VBA (Seite 1684)

Gruppen-Objekte (Seite 1679)

Objekte mit VBA bearbeiten (Seite 1662)

VBA im Graphics Designer (Seite 1623)

So bearbeiten Objekte im Gruppen-Objekt mit VBA

Einleitung

Sie finden hier folgende Anleitungen zum Bearbeiten von Objekten im Gruppen-Objekt mit VBA:

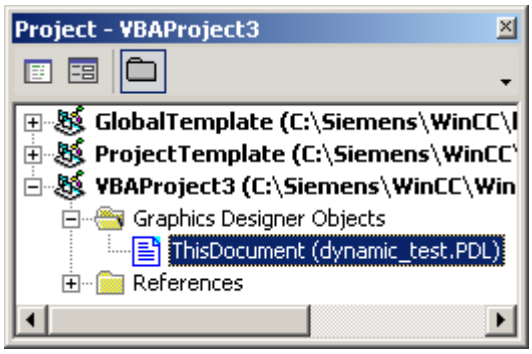
- Eine Eigenschaft eines Objektes im Gruppen-Objekt bearbeiten
- Eine Eigenschaft aller Objekte im Gruppen-Objekt bearbeiten

Voraussetzung

Sie müssen im Graphics Designer mindestens zwei Grafik-Objekte angelegt und diese gruppiert haben.

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor").
2. Öffnen Sie im Project Explorer das Dokument "ThisDocument":



3. Um eine Eigenschaft eines Objekt innerhalb des Gruppen-Objektes zu bearbeiten, fügen Sie z.B. eine Prozedur "ChangePropertiesOfGroupMembers()" in das Dokument "ThisDocument" ein. In diesem Beispiel werden im Gruppen-Objekt "My Group" die Eigenschaften von drei verschiedenen Objekten geändert:

```
Sub ChangePropertiesOfGroupMembers()
'VBA50
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objEllipse As HMIEllipse
Dim objGroup As HMIGroup
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle",
"HMICircle")
Set objRectangle =
ActiveDocument.HMIObjects.AddHMIObject("sRectangle",
"HMIRectangle")
Set objEllipse =
ActiveDocument.HMIObjects.AddHMIObject("sEllipse", "HMIEllipse")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
With objEllipse
.Top = 120
.Left = 120
.Selected = True
End With
MsgBox "Objects selected!"
Set objGroup = ActiveDocument.Selection.CreateGroup
objGroup.ObjectName = "My Group"
'Set bordercolor of 1. object = "red":
objGroup.GroupedHMIObjects(1).Properties("BorderColor") =
RGB(255, 0, 0)
'set x-coordinate of 2. object = "120" :
objGroup.GroupedHMIObjects(2).Properties("Left") = 120
'set y-coordinate of 3. object = "90":
objGroup.GroupedHMIObjects(3).Properties("Top") = 90
End Sub
```

4. Um die Eigenschaften von allen Objekten im Gruppen-Objekt zu ändern, fügen Sie z.B. eine Prozedur "ChangePropertiesOfAllGroupMembers()" in das Dokument "ThisDocument" ein. In diesem Beispiel werden im Gruppen-Objekt "My Group" die Eigenschaft "BorderColor" von jedem Objekt geändert. Damit dieses Beispiel funktioniert, müssen Sie das Gruppen-Objekt "My Group" angelegt haben:

```
Sub ChangePropertiesOfAllGroupMembers()
'VBA51
Dim objGroup As HMIGroup
Dim iMaxMembers As Integer
Dim iIndex As Integer
Set objGroup = ActiveDocument.HMIObjects("My Group")
iIndex = 1
'
'Get number of objects in group-object:
iMaxMembers = objGroup.GroupedHMIObjects.Count
'
'set linecolor of all objects = "yellow":
For iIndex = 1 To iMaxMembers
objGroup.GroupedHMIObjects(iIndex).Properties("BorderColor") =
RGB(255, 255, 0)
Next iIndex
End Sub
```

5. Starten Sie die Prozeduren jeweils mit <F5>.

Siehe auch

Properties-Objekt (Auflistung) (Seite 2004)

GroupedObjects-Objekt (Auflistung) (Seite 1950)

Ungroup-Methode (Seite 1877)

Remove-Methode (Seite 1857)

Delete-Methode (Seite 1818)

Add-Methode (GroupedObjects-Auflistung) (Seite 1781)

So bearbeiten Sie Gruppen-Objekte mit VBA (Seite 1681)

Gruppen-Objekte (Seite 1679)

Objekte mit VBA bearbeiten (Seite 1662)

VBA im Graphics Designer (Seite 1623)

3.3.4.4 Anwender-Objekte

Anwender-Objekte

Einleitung

Mit VBA können Sie im Graphics Designer aus ausgewählten Objekten ein Anwender-Objekt erzeugen. Im Gegensatz zum Gruppen-Objekt sind beim Anwender-Objekt nur die Objekteigenschaften verfügbar, die Sie im "Konfigurationsdialog" des Anwender-Objektes ausgewählt haben. Es ist nicht möglich, ein Anwender-Objekt mit VBA zu konfigurieren.

Weitere Informationen zum Anwender-Objekt finden Sie in der WinCC-Dokumentation unter "Anwender-Objekt".

Anwender-Objekt mit VBA erzeugen

Verwenden Sie die Methode "CreateCustomizedObject()", um ein Anwender-Objekt aus ausgewählten Objekten zu erzeugen:

```
Sub CreateCustomizedObject()
'VBA52
Dim objCustomizedObject As HMICustomizedObject
Set objCustomizedObject = ActiveDocument.Selection.CreateCustomizedObject
objCustomizedObject.ObjectName = "My Customized Object"
End Sub
```

Bei Anwendung der Methode "CreateCustomizedObject()" erscheint der "Konfigurationsdialog", in dem Sie die Objekteigenschaften auswählen. Das so erzeugte Anwender-Objekt wird zur Auflistung "HMIObjects" hinzugefügt. Geben Sie dem Anwender-Objekt einen sprechenden Namen (objCustomizedObject.Name = "My Customized Object"), damit Sie es eindeutig identifizieren können.

Hinweis

Wenn Sie ein Dokument als nicht sichtbar geöffnet haben, erstellen Sie dort per VBA-Skript kein Anwenderobjekt. Der Programmablauf wird durch einen Konfigurationsdialog unterbrochen.

Anwender-Objekt bearbeiten

Das Anwender-Objekt können Sie wie folgt bearbeiten:

- Methode "Destroy": Löst das Anwender-Objekt auf.
- Methode "Delete": Löscht das Anwender-Objekt und die darin enthaltenen Objekte.

Objekte im Anwender-Objekt bearbeiten

Verwenden Sie die Eigenschaft "Properties", um auf die ausgewählten Objekteigenschaften der im Anwender-Objekt enthaltenen Objekte zuzugreifen:

```
Sub EditCustomizedObjectProperty()
'VBA53
Dim objCustomizedObject As HMICustomizedObject
Set objCustomizedObject = ActiveDocument.HMIObjects(1)
objCustomizedObject.Properties("BackColor") = RGB(255, 0, 0)
End Sub
```

Haben Sie mehrere identische Eigenschaften ausgewählt (z.B. die Hintergrundfarbe eines Kreises und eines Rechteckes), werden diese Eigenschaften nummeriert ("BackColor" und "BackColor1").

Siehe auch

HMIObject-Objekt (Seite 1955)

CustomizedObject-Objekt (Seite 1912)

Destroy-Methode (Seite 1822)

Delete-Methode (Seite 1818)

CreateCustomizedObject-Methode (Seite 1813)

So bearbeiten Sie ein Anwender-Objekt mit VBA (Seite 1688)

So bearbeiten Sie Gruppen-Objekte mit VBA (Seite 1681)

Gruppen-Objekte (Seite 1679)

Objekte mit VBA bearbeiten (Seite 1662)

So bearbeiten Sie ein Anwender-Objekt mit VBA

Einleitung

Sie finden hier folgende Anleitungen zum Bearbeiten eines Anwender-Objektes mit VBA:

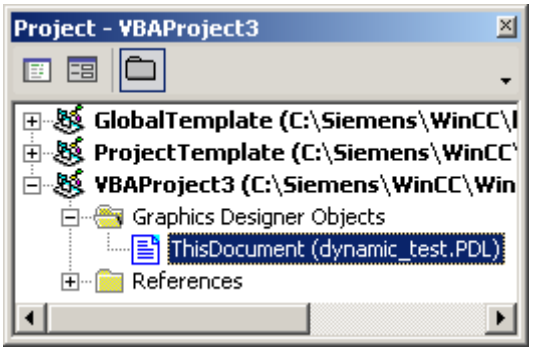
- Anwender-Objekt aus ausgewählten Objekten erzeugen
- Anwender-Objekt auflösen
- Anwender-Objekt löschen

Hinweis

Es ist nicht möglich, ein Anwender-Objekt mit VBA zu konfigurieren.

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor").
2. Öffnen Sie im Project Explorer das Dokument "ThisDocument":



3. Um ein Anwender-Objekt aus ausgewählten Objekten zu erzeugen, fügen Sie z.B. eine Prozedur "CreateCustomizedObject()" in das Dokument "ThisDocument" ein. In diesem Beispiel wird aus ausgewählten Objekten das Anwender-Objekt "My Customized Object" erzeugt:

```
Sub CreateCustomizedObject()
'VBA54
Dim objCustomizedObject As HMICustomizedObject
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle",
"HMICircle")
Set objRectangle =
ActiveDocument.HMIObjects.AddHMIObject("sRectangle",
"HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
Set objCustomizedObject =
ActiveDocument.Selection.CreateCustomizedObject
'
'*** The "Configurationdialog" started. ***
'*** Configure the costumize-object with the "configurationdialog"
***
'
objCustomizedObject.ObjectName = "My Customized Object"
End Sub
```

4. Um ein Anwender-Objekt wieder aufzulösen, fügen Sie z.B. eine Prozedur "DestroyCustomizedObject()" in das Dokument "ThisDocument" ein. In diesem Beispiel wird das zuvor erzeugte Anwender-Objekt "My Customized Object" wieder aufgelöst:
```
Sub DestroyCustomizedObject()
'VBA55
Dim objCustomizedObject As HMICustomizedObject
Set objCustomizedObject = ActiveDocument.HMIObjects("My
Customized Object")
objCustomizedObject.Destroy
End Sub
```
5. Um ein Anwender-Objekt zu löschen, fügen Sie z.B. eine Prozedur "DeleteCustomizedObject()" in das Dokument "ThisDocument" ein. In diesem Beispiel wird das zuvor erzeugte Anwender-Objekt "My Customized Object" gelöscht:
```
Sub DeleteCustomizedObject()
'VBA56
Dim objCustomizedObject As HMICustomizedObject
Set objCustomizedObject = ActiveDocument.HMIObjects("My
Customized Object")
objCustomizedObject.Delete
End Sub
```
6. Starten Sie die Prozeduren jeweils mit <F5>.

Siehe auch

Destroy-Methode (Seite 1822)

Delete-Methode (Seite 1818)

CreateCustomizedObject-Methode (Seite 1813)

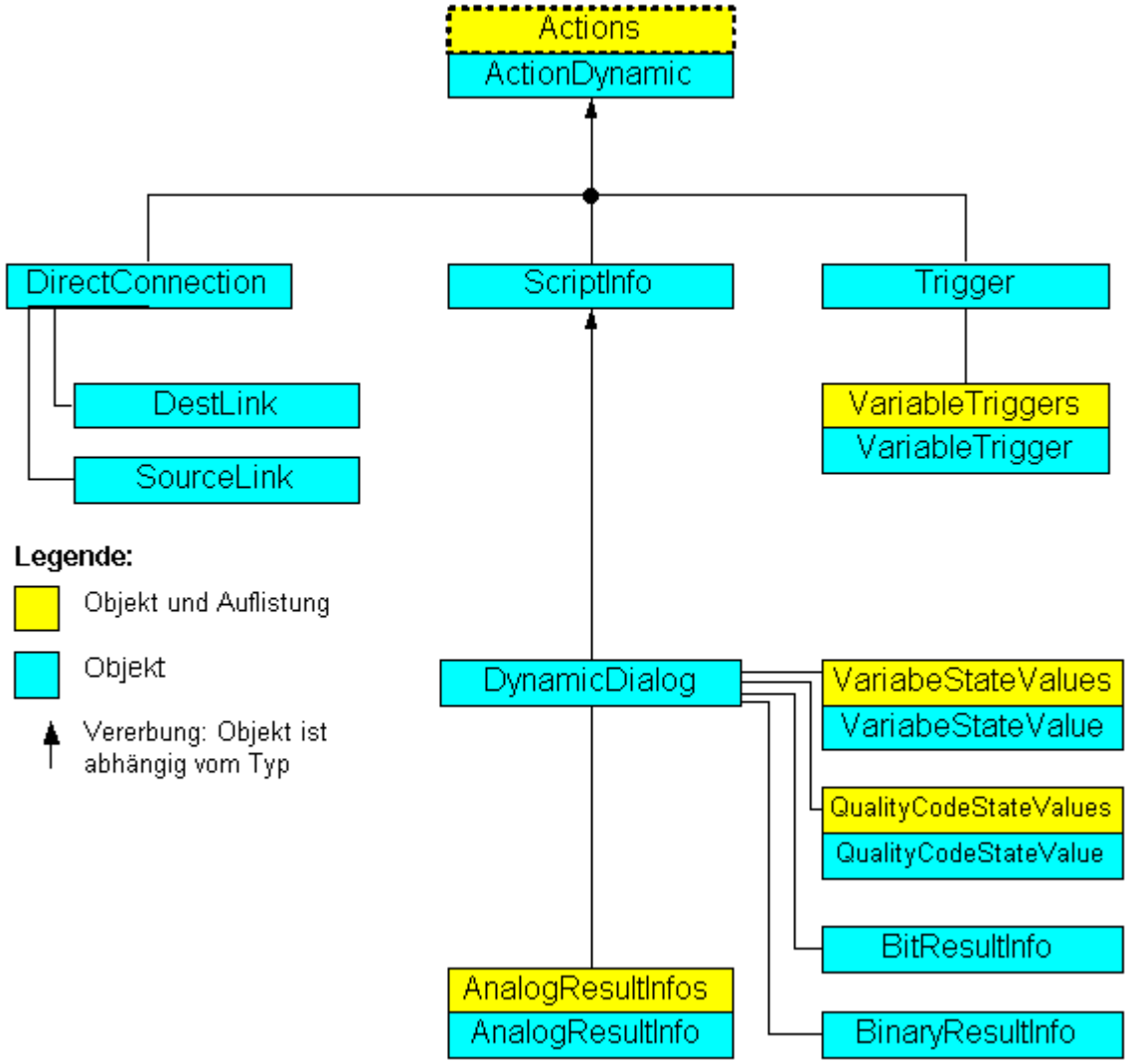
Anwender-Objekte (Seite 1687)

3.3.5 Dynamisierungen anlegen mit VBA

3.3.5.1 Dynamisierungen anlegen mit VBA

Einleitung

Mit VBA haben Sie die Möglichkeit, Eigenschaften von Bildern und Objekten zu dynamisieren sowie ereignisgesteuerte Aktionen zu projektieren. VBA stellt Ihnen dazu das ActionDynamic-Objekt zur Verfügung:



Das ActionDynamic-Objekt stellt eine Schnittstelle dar, die abhängig vom Objekttyp ist:

- Projektieren Sie eine Dynamik an eine Eigenschaft (Property-Objekt), erbt das ActionDynamic-Objekt die Eigenschaften der Objekte ScriptInfo, Trigger und DynamicDialog.
- Projektieren Sie eine ereignisgesteuerte Aktion (Event-Objekt), erbt das ActionDynamic-Objekt die Eigenschaften der Objekte ScriptInfo und DirectConnection.

Eigenschaften von Bildern und Objekten dynamisieren

Mit VBA können Sie Eigenschaften von Bildern und Objekten dynamisieren. Zur Dynamisierung können Sie Variablen, Skripte oder den Dynamik-Dialog verwenden. Mit der Dynamisierung können Sie z.B. in Runtime den Farbumschlag eines Objektes projektieren, wenn sich ein Variablenwert ändert.

Projektierung von ereignisgesteuerten Aktionen

Mit VBA können Sie ereignisgesteuerte Aktionen projektieren. Eine Aktion (Skript oder Direktverbindung) wird ausgelöst, wenn in Runtime das definierte Ereignis eintritt. Ein Ereignis kann z.B. die Änderung einer Objekteigenschaft oder ein Klick auf einen Button sein.

Bearbeiten von Triggern

Mit VBA können Sie Trigger bearbeiten. Trigger werden bei der Dynamisierung benötigt. Sie bestimmen, wann ein dynamisierter Wert in Runtime aktualisiert wird. Dies kann z.B. in regelmäßigen Zeitabständen geschehen oder bei bei einem Bildwechsel.

Beim Projektieren von ereignisgesteuerten Aktionen stellt das Ereignis den Trigger dar.

Siehe auch

Bearbeiten von Triggern (Seite 1712)

Projektierung von ereignisgesteuerten Aktionen mit VBA (Seite 1704)

Dynamisieren von Eigenschaften von Bildern und Objekten (Seite 1692)

3.3.5.2 Dynamisieren von Eigenschaften von Bildern und Objekten

Dynamisieren von Eigenschaften von Bildern und Objekten

Einleitung

Mit VBA können Sie Eigenschaften von Bildern und Objekten dynamisieren. In Runtime können dynamisierte Objekteigenschaften z.B. in Abhängigkeit eines Variablenwertes verändert werden. Folgende Dynamisierungsmethoden sind möglich:

- Variablenanbindung
- Dynamik-Dialog
- Skripte

Prinzip

Das folgende Beispiel zeigt die prinzipielle Vorgehensweise zur Dynamisierung einer Objekteigenschaft:

```
Sub CreateDynamicOnProperty()
'VBA57
Dim objVariableTrigger As HMIVariableTrigger
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle1", "HMICircle")
'
'Create dynamic with type "direct Variableconnection" at the
'property "Radius":
Set objVariableTrigger =
objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeVariableDirect, "'NewDynamic1'")
'
'To complete dynamic, e.g. define cycle:
With objVariableTrigger
.CycleType = hmiVariableCycleType_2s
End With
End Sub
```

Hinweis

Beachten Sie, dass mit dem Festlegen des Variablennamens keine Variable angelegt wird. Verwenden Sie dazu den Variablenauswahldialog, um die Variable anzulegen.

Variablenanbindung

Verwenden Sie das VariableTrigger-Objekt, um eine Eigenschaft mit einer direkten oder indirekten Variablenanbindung zu dynamisieren. Die so dynamisierte Eigenschaft reagiert in Runtime auf die Wertänderung der angegebenen Variablen. In VBA geben Sie dazu den Variablennamen (VarName-Eigenschaft) und den Zyklus (CycleTime-Eigenschaft) an.

Dynamik-Dialog

Verwenden Sie das DynamicDialog-Objekt, um eine Eigenschaft mit Hilfe des Dynamik-Dialoges zu dynamisieren. Die so dynamisierte Eigenschaft reagiert in Runtime auf

Wertebereiche einer Variablen. Für die Festlegung des Wertebereiches stehen folgende Objekte zur Verfügung:

- AnalogResultInfos-Objekt: Verwenden Sie das Objekt, um Wertebereichen einer Variablen oder eines Skriptes einen festen Wert zuzuordnen. Der feste Wert wird der dynamisierten Eigenschaft zugewiesen, wenn sich der Variablen- oder Rückgabewert des Skriptes im angegebenen Wertebereich befindet.
- BinaryResultInfo-Objekt: Verwenden Sie das Objekt, um binären Wertebereichen (Null und ungleich Null) einer Variablen oder eines Skriptes einen festen Wert zuzuordnen. Der feste Wert wird der dynamisierten Eigenschaft zugewiesen, wenn der Variablen- oder Rückgabewert des Skriptes einen der beiden Werte zurückgibt.
- VariableStateValue-Objekt: Verwenden Sie das Objekt, um dem Status (z.B. "Obergrenze überschritten") einer angegebenen Variable einen festen Wert zuzuordnen. Der feste Wert wird dann der dynamisierten Eigenschaft zugewiesen, wenn der Status eintritt.

Skripte

Verwenden Sie das ScriptInfo-Objekt, um eine Eigenschaft mit einem C- oder VB-Skript zu dynamisieren. Die so dynamisierte Eigenschaft reagiert in Runtime auf ein Skript und wird über einen Trigger gesteuert. Verwenden Sie zum Projektieren des Triggers das Trigger-Objekt.

Siehe auch

VariableTrigger-Objekt (Seite 2060)

VariableStateValue-Objekt (Seite 2057)

Trigger-Objekt (Seite 2047)

ScriptInfo-Objekt (Seite 2021)

BinaryResultInfo-Objekt (Seite 1896)

AnalogResultInfos-Objekt (Auflistung) (Seite 1887)

So dynamisieren Sie eine Eigenschaft mit einem VB-Skript (Seite 1702)

So dynamisieren Sie eine Eigenschaft mit einem C-Skript (Seite 1700)

So dynamisieren Sie eine Eigenschaft mit dem Dynamik-Dialog (Seite 1697)

So dynamisieren Sie eine Eigenschaft mit einer Variablenanbindung (Seite 1695)

Dynamisierungen anlegen mit VBA (Seite 1691)

So dynamisieren Sie eine Eigenschaft mit einer Variablenanbindung

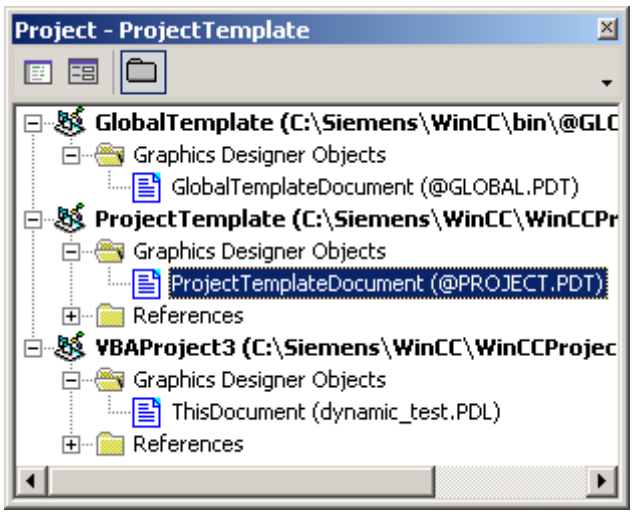
Einleitung

Sie finden hier folgende Anleitungen zum Dynamisieren einer Eigenschaft mit einer Variablenanbindung:

- Eigenschaft mit direkter Variablenanbindung dynamisieren
- Eigenschaft mit indirekter Variablenanbindung dynamisieren

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor").
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



3. Um eine Objekteigenschaft mit einer direkten Variablenanbindung zu dynamisieren, fügen Sie z.B. eine Prozedur "AddDynamicAsVariableDirectToProperty()" in das Dokument ein. In diesem Beispiel wird die Eigenschaft "Top" eines Kreises mit der Variable "Otto" dynamisiert:

```
Sub AddDynamicAsVariableDirectToProperty()
'VBA58
Dim objVariableTrigger As HMIVariableTrigger
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle1",
"HMICircle")
'Create dynamic at property "Top"
Set objVariableTrigger =
objCircle.Top.CreateDynamic(hmiDynamicCreationTypeVariableDirect,
"Otto")
'
'define cycle-time
With objVariableTrigger
.CycleType = hmiVariableCycleType_2s
End With
End Sub
```

4. Um eine Objekteigenschaft mit einer indirekten Variablenanbindung zu dynamisieren, fügen Sie z.B. eine Prozedur "AddDynamicAsVariableIndirectToProperty()" in das Dokument ein. In diesem Beispiel wird die Eigenschaft "Left" eines Kreises mit der Variable "Anton" dynamisiert:

```
Sub AddDynamicAsVariableIndirectToProperty()
'VBA59
Dim objVariableTrigger As HMIVariableTrigger
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle2",
"HMICircle")
'Create dynamic on property "Left":
Set objVariableTrigger =
objCircle.Left.CreateDynamic(hmiDynamicCreationTypeVariableIndirec
t, "Anton")
'
'Define cycle-time
With objVariableTrigger
.CycleType = hmiVariableCycleType_2s
End With
End Sub
```

5. Starten Sie die Prozeduren jeweils mit <F5>.

Siehe auch

CycleType-Eigenschaft (Seite 2160)

VarName-Eigenschaft (Seite 2481)

VariableTrigger-Objekt (Seite 2060)

CreateDynamic-Methode (Seite 1815)

So dynamisieren Sie eine Eigenschaft mit einem VB-Skript (Seite 1702)

So dynamisieren Sie eine Eigenschaft mit einem C-Skript (Seite 1700)

So dynamisieren Sie eine Eigenschaft mit dem Dynamik-Dialog (Seite 1697)

Dynamisieren von Eigenschaften von Bildern und Objekten (Seite 1692)

Dynamisierungen anlegen mit VBA (Seite 1691)

So dynamisieren Sie eine Eigenschaft mit dem Dynamik-Dialog

Einleitung

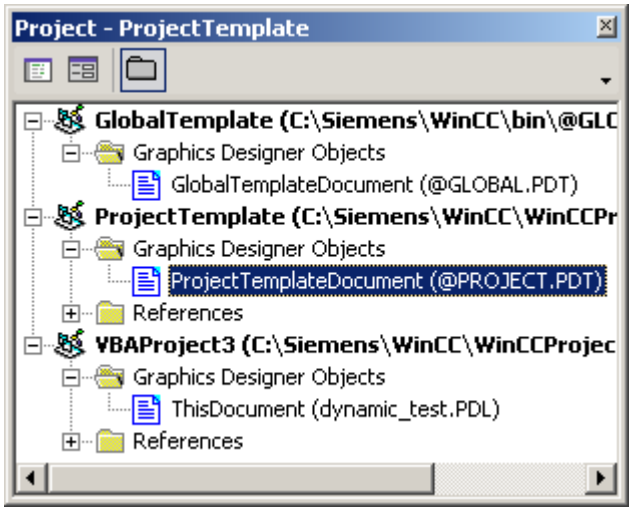
Mit dem Dynamik-Dialog können Sie Eigenschaften von Bildern und Objekten in Abhängigkeit von bestimmten Wertebereichen oder Variablenstati dynamisieren. Folgende Wertebereiche stehen dabei zur Auswahl:

- Analog
- Binär
- Bit
- Direkt

Mit VBA legen Sie die Wertbereichsart mit der ResultType-Eigenschaft fest. Diese Anleitung zeigt die Dynamisierung einer Objekteigenschaft mit analogen Wertebereichen. Weiterführende Informationen zum Dynamisieren mit dem Dynamik-Dialog finden Sie unter "DynamicDialog-Objekt" in der VBA-Referenz in dieser Dokumentation.

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor").
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



3. Um eine Objekteigenschaft mit dem Dynamik-Dialog zu dynamisieren, fügen Sie z.B. eine Prozedur "AddDynamicDialogToCircleRadiusTypeAnalog()" in das Dokument ein. Im nachfolgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert, ein Variablenname vergeben und drei analoge Wertebereiche angelegt.
4. Starten Sie die Prozedur mit <F5>.

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA60
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
'
'Create dynamic
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog, "'NewDynamic1'")
'
'Configure dynamic. "ResultType" defines the type of valuerange:
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.Add 50, 40
.AnalogResultInfos.Add 100, 80
.AnalogResultInfos.ElseCase = 100
End With
End Sub
```

Neue VBA-Methode zur Dynamisierung mit dem Dynamik-Dialog

Aus Optimierungsgründen wurde zusätzlich eine neue Methode zur Verfügung gestellt:

- CreateDynamicDialog([Code as String], iResultType as Long) as HMIActionDynamic

Der Parameter "IResultType" hat folgende Konstanten:

- hmiResultTypeDirect = 0
- hmiResultTypeAnalog= 1
- hmiResultTypeBool = 2
- hmiResultTypeBit = 3

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Dem Dynamik-Dialog werden ein Variablenname und ein "ResultType" zugewiesen.

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA820
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
'Create Object
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("myCircle","HMICircle")
'Create dynamic (Tag "myTest" must be exist")
Set objDynDialog = objCircle.Radius.CreateDynamicDialog("'myTest'",1)
End Sub
```

String-Eigenschaft initialisieren

Eine String-Eigenschaft müssen Sie vor der Dynamisierung initialisieren, indem Sie ihr einen Text zuweisen. Im folgenden ToolTipText-Beispiel geschieht das in "objCircle.ToolTipText = "Text".

```
Sub Dyn()
'VBA823
Dim objCircle As HMICircle
Dim doc As Document
Dim objDynDialog As HMIDynamicDialog
Set doc = ActiveDocument
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle","HMICircle")
objCircle.ObjectName = "Circle1"
objCircle.BorderColor = RGB(255, 0, 0)
objCircle.BackColor = RGB(0, 255, 0)
objCircle.ToolTipText = "Text"
Set objDynDialog = objCircle.ToolTipText.CreateDynamic(hmiDynamicCreationTypeDynamicDialog, "'Var'")
End Sub
```

Siehe auch

So dynamisieren Sie eine Eigenschaft mit einer Variablenanbindung (Seite 1695)

ResultType-Eigenschaft (Seite 2349)

DynamicDialog-Objekt (Seite 1924)

CreateDynamic-Methode (Seite 1815)

So dynamisieren Sie eine Eigenschaft mit einem VB-Skript (Seite 1702)

So dynamisieren Sie eine Eigenschaft mit einem C-Skript (Seite 1700)

Dynamisieren von Eigenschaften von Bildern und Objekten (Seite 1692)

Dynamisierungen anlegen mit VBA (Seite 1691)

TooltipText-Eigenschaft (Seite 2387)

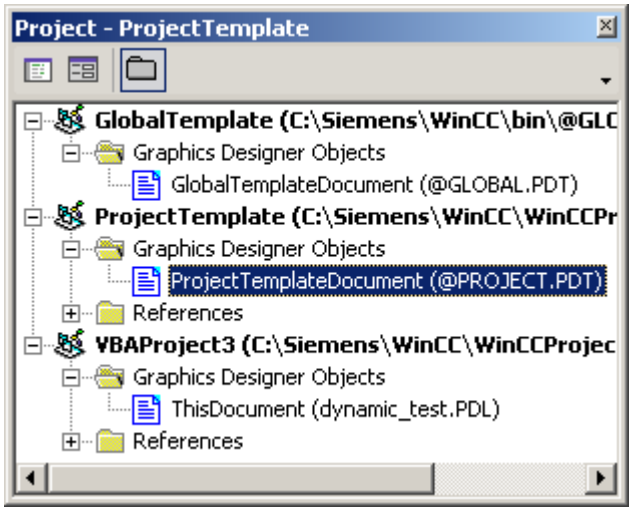
So dynamisieren Sie eine Eigenschaft mit einem C-Skript

Einleitung

Wenn Sie eine Eigenschaft mit einem C-Skript dynamisieren, können Sie der Eigenschaft "SourceCode" den C-Code zuweisen. Das C-Skript wird im Hintergrund kompiliert. Die Eigenschaft "Compiled" liefert "True" zurück, wenn der C-Code erfolgreich kompiliert wurde.

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor").
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



3. Um eine Objekteigenschaft mit einem C-Skript zu dynamisieren, fügen Sie z.B. eine Prozedur "AddDynamicAsCSkriptToProperty()" in das Dokument ein. In diesem Beispiel wird in Runtime die Höhe eines Kreises alle zwei Sekunden um 5 Pixel vergrößert:

```
Sub AddDynamicAsCSkriptToProperty()
'VBA61
Dim objCScript As HMIScriptInfo
Dim objCircle As HMICircle
Dim strCode As String
strCode = "long lHeight;" & vbCrLf & "int check;" & vbCrLf
strCode = strCode & "GetHeight("" & "events.PDL"" & ","" &
"myCircle"" & ");" & vbCrLf
strCode = strCode & "lHeight = lHeight+5;" & vbCrLf
strCode = strCode & "check = SetHeight(""events.PDL"",
""myCircle"", lHeight );"
strCode = strCode & vbCrLf & "//Return-Type: BOOL" & vbCrLf
strCode = strCode & "return check;"
Set objCircle =
ActiveDocument.HMIObjects.AddHMIObject("myCircle", "HMICircle")
'Create dynamic for Property "Height":
Set objCScript =
objCircle.Height.CreateDynamic(hmiDynamicCreationTypeCScript)
'
'set Sourcecode and cycletime:
With objCScript
.SourceCode = strCode
.Trigger.Type = hmiTriggerTypeStandardCycle
.Trigger.CycleType = hmiCycleType_2s
.Trigger.Name = "Trigger1"
End With
End Sub
```

4. Starten Sie die Prozedur mit <F5>.

Siehe auch

Trigger-Eigenschaft (Seite 2391)

ScriptType-Eigenschaft (Seite 2357)

SourceCode-Eigenschaft (Seite 2372)

CycleType-Eigenschaft (Seite 2160)

ScriptInfo-Objekt (Seite 2021)

CreateDynamic-Methode (Seite 1815)

So dynamisieren Sie eine Eigenschaft mit einem VB-Skript (Seite 1702)

So dynamisieren Sie eine Eigenschaft mit dem Dynamik-Dialog (Seite 1697)

So dynamisieren Sie eine Eigenschaft mit einer Variablenanbindung (Seite 1695)

Dynamisieren von Eigenschaften von Bildern und Objekten (Seite 1692)

Dynamisierungen anlegen mit VBA (Seite 1691)

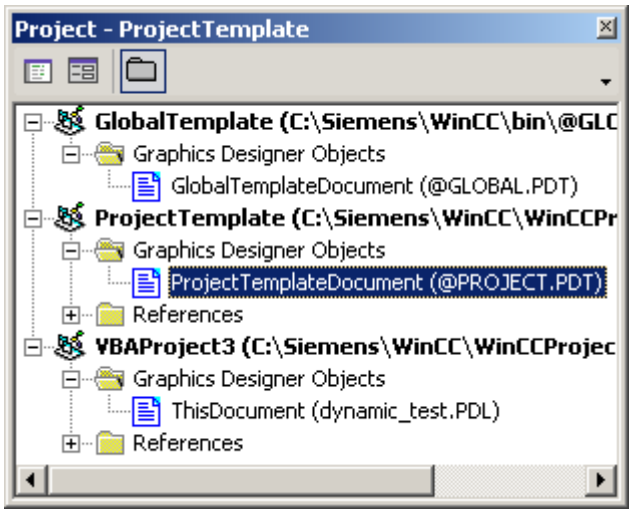
So dynamisieren Sie eine Eigenschaft mit einem VB-Skript

Einleitung

Wenn Sie eine Eigenschaft mit einem VB-Skript dynamisieren, können Sie der Eigenschaft "SourceCode" den VB-Code zuweisen. Das VB-Skript wird im Hintergrund kompiliert. Die Eigenschaft "Compiled" liefert "True" zurück, wenn der VB-Code syntaktisch korrekt ist.

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor").
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



3. Um eine Objekteigenschaft mit einem VB-Skript zu dynamisieren, fügen Sie z.B. eine Prozedur "AddDynamicAsVBSkriptToProperty()" in das Dokument ein. In diesem Beispiel wird in Runtime der Radius eines Kreises alle zwei Sekunden um 5 Pixel vergrößert:

```
Sub AddDynamicAsVBSkriptToProperty()
'VBA62
Dim objVBScript As HMIScriptInfo
Dim objCircle As HMICircle
Dim strCode As String
strCode = "Dim myCircle" & vbCrLf & "Set myCircle = "
strCode = strCode &
"HMIRuntime.ActiveScreen.ScreenItems(""myCircle"")"
strCode = strCode & vbCrLf & "myCircle.Radius = myCircle.Radius +
5"
Set objCircle =
ActiveDocument.HMIObjects.AddHMIObject("myCircle", "HMICircle")
'
'Create dynamic of property "Radius":
Set objVBScript =
objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeVBScript)
'
'Set SourceCode and cycletime:
With objVBScript
.SourceCode = strCode
.Trigger.Type = hmiTriggerTypeStandardCycle
.Trigger.CycleType = hmiCycleType_2s
.Trigger.Name = "Trigger1"
End With
End Sub
```

4. Starten Sie die Prozedur mit <F5>.

Siehe auch

So dynamisieren Sie eine Eigenschaft mit einem C-Skript (Seite 1700)

Trigger-Eigenschaft (Seite 2391)

SourceCode-Eigenschaft (Seite 2372)

CycleType-Eigenschaft (Seite 2160)

ScriptInfo-Objekt (Seite 2021)

CreateDynamic-Methode (Seite 1815)

So dynamisieren Sie eine Eigenschaft mit dem Dynamik-Dialog (Seite 1697)

So dynamisieren Sie eine Eigenschaft mit einer Variablenanbindung (Seite 1695)

Dynamisieren von Eigenschaften von Bildern und Objekten (Seite 1692)

Dynamisierungen anlegen mit VBA (Seite 1691)

3.3.5.3 Projektierung von ereignisgesteuerten Aktionen mit VBA

Projektierung von ereignisgesteuerten Aktionen mit VBA

Einleitung

Mit VBA können Sie bei Bildern und Objekten Aktionen projektieren, die beim Eintreten vordefinierter Ereignisse ausgelöst werden. In Runtime kann z.B. beim Mausklick auf ein Objekt ein C-Skript aufgerufen werden, dessen Rückgabewert zur Dynamisierung einer Objekteigenschaft verwendet wird. Folgende Dynamisierungsmethoden sind möglich:

- Direktverbindung
- Skripte

Die Ereignisse, die für die Projektierung von ereignisgesteuerten Aktionen verwendet werden, treten ausschließlich in Runtime auf und haben mit den VBA-Event-Handlern nichts zu tun.

Prinzip

Für die Projektierung von ereignisgesteuerten Aktionen mit VBA verwenden Sie die Events-Eigenschaft. Die Verwendung der Eigenschaft ist davon abhängig, ob Sie eine Aktion an ein Objekt oder Bild oder eine Eigenschaft projektieren.

Aktion an ein Objekt oder Bild projektieren

Eine Aktion, die Sie an ein Bild oder Objekt projektieren, wird dann ausgelöst, wenn ein vordefiniertes Ereignis eintritt, z.B. das Objekt mit der Maus angeklickt wird. Sie projektieren mit VBA eine Aktion an ein Objekt, indem Sie die Eigenschaft "Events(Index)" verwenden, wobei "Index" für das auslösende Ereignis steht:

```
Sub AddActionToObjectTypeCScript()
'VBA63
Dim objEvent As HMIEvent
Dim objCScript As HMIScriptInfo
Dim objCircle As HMICircle
'Create circle. Click on object executes an C-action
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_AB", "HMICircle")
Set objEvent = objCircle.Events(1)
Set objCScript = objEvent.Actions.AddAction(hmiActionCreationTypeCScript)
'
'Assign a corresponding custom-function to the property "SourceCode":
objCScript.SourceCode = ""
End Sub
```

Aktion an eine Eigenschaft projektieren

Eine Aktion, die Sie an eine Eigenschaft eines Bildes oder Objektes projektieren, wird dann ausgelöst, wenn sich der Eigenschaftswert ändert. Sie projektieren mit VBA eine Aktion an eine Eigenschaft, indem Sie die Eigenschaft "Events(1)" verwenden, wobei der Index "1" für das Ereignis "bei Änderung" steht:

```
Sub AddActionToPropertyTypeCScript()
'VBA64
Dim objEvent As HMIEvent
Dim objCScript As HMIScriptInfo
Dim objCircle As HMICircle
'Create circle. Changing of the Property
'"Radius" should be activate C-Aktion:
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_AB", "HMICircle")
Set objEvent = objCircle.Radius.Events(1)
Set objCScript = objEvent.Actions.AddAction(hmiActionCreationTypeCScript)
'
'Assign a corresponding custom-function to the property "SourceCode":
objCScript.SourceCode = ""
End Sub
```

Direktverbindung

Verwenden Sie das DirectConnection-Objekt, um eine Direktverbindung zu projektieren.

Skripte

Verwenden Sie das ScriptInfo-Objekt, wenn ein Ereignis eine C- oder VB-Aktion auslösen soll.

Siehe auch

So projektieren Sie eine VB-Aktion mit VBA an ein Ereignis (Seite 1710)

Events-Eigenschaft (Seite 2171)

ScriptInfo-Objekt (Seite 2021)

Event-Objekt (Seite 1935)

So projektieren Sie eine C-Aktion mit VBA an ein Ereignis (Seite 1708)

So projektieren Sie eine Direktverbindung mit VBA (Seite 1705)

Event-Handling (Seite 1715)

Dynamisierungen anlegen mit VBA (Seite 1691)

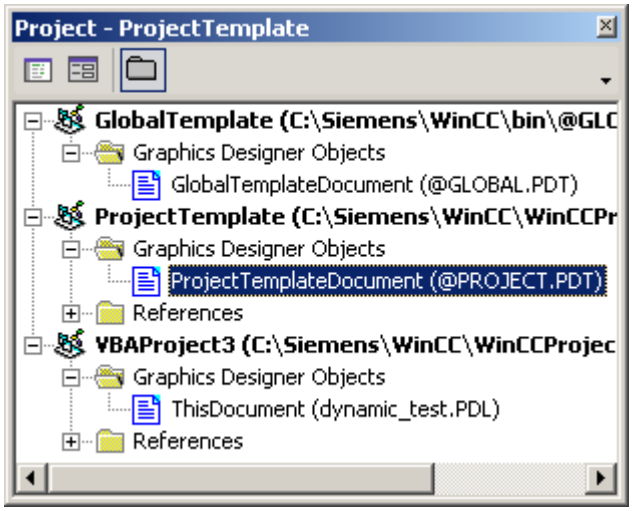
So projektieren Sie eine Direktverbindung mit VBA

Einleitung

Diese Anleitung zeigt die Projektierung einer Direktverbindung anhand zweier Objekteigenschaften. Weitere Informationen zum Projektieren von Direktverbindungen mit VBA erhalten Sie in der VBA-Referenz in dieser Dokumentation unter "AutomationName-Eigenschaft" und "ObjectName-Eigenschaft"

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor").
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



3. Um eine Direktverbindung an eine Objekteigenschaft zu projektieren, fügen Sie z.B. die Prozedur "AddDirectConnectionToObject()" in das Dokument ein. Im folgenden Beispiel wird in Runtime bei einem Mausklick auf den Button die x-Position von "Rectangle_A" in die y-Position von "Rectangle_B" kopiert:

```
Sub DirectConnection()
'VBA65
Dim objButton As HMIButton
Dim objRectangleA As HMIRectangle
Dim objRectangleB As HMIRectangle
Dim objEvent As HMIEvent
Dim objDConnection As HMIDirectConnection
'
'Create objects:
Set objRectangleA =
ActiveDocument.HMIObjects.AddHMIObject("Rectangle_A",
"HMIRectangle")
Set objRectangleB =
ActiveDocument.HMIObjects.AddHMIObject("Rectangle_B",
"HMIRectangle")
Set objButton =
ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objRectangleA
.Top = 100
.Left = 100
End With
With objRectangleB
.Top = 250
.Left = 400
.BackColor = RGB(255, 0, 0)
End With
With objButton
.Top = 10
.Left = 10
.Text = "SetPosition"
End With
'
'Directconnection is initiated by mouseclick:
Set objDConnection =
objButton.Events(1).Actions.AddAction(hmiActionCreationTypeDirectC
onnection)
With objDConnection
'Sourceobject: Property "Top" of Rectangle_A
.SourceLink.Type = hmiSourceTypeProperty
.SourceLink.ObjectName = "Rectangle_A"
.SourceLink.AutomationName = "Top"
'
'Destinationobject: Property "Left" of Rectangle_B
.DestinationLink.Type = hmiDestTypeProperty
.DestinationLink.ObjectName = "Rectangle_B"
.DestinationLink.AutomationName = "Left"
End With
```

```
End Sub
```

4. Starten Sie die Prozedur mit <F5>.

Siehe auch

ObjectName-Eigenschaft (Seite 2307)

AutomationName-Eigenschaft (Seite 2082)

SourceLink-Objekt (Seite 2028)

DestLink-Objekt (Seite 1917)

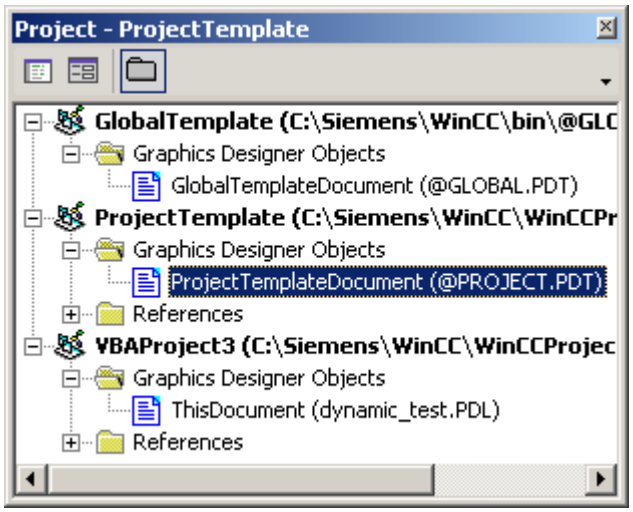
DirectConnection-Objekt (Seite 1918)

Projektierung von ereignisgesteuerten Aktionen mit VBA (Seite 1704)

So projektieren Sie eine C-Aktion mit VBA an ein Ereignis

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor").
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



3. Um eine C-Aktion mit VBA an ein Ereignis zu projektieren, fügen Sie z.B. eine Prozedur "CreateCActionToClickedEvent()" in das Dokument ein. In diesem Beispiel werden ein Button und ein Kreis in das aktive Bild eingefügt. In Runtime vergrößert sich die Höhe mit jedem Klick auf den Button:

```
Sub CreateCActionToClickedEvent()
'VBA66
Dim objButton As HMIButton
Dim objCircle As HMICircle
Dim objEvent As HMIEvent
Dim objCScript As HMIScriptInfo
Dim strCode As String
strCode = "long lHeight;" & vbCrLf & "int check;" & vbCrLf
strCode = strCode & "lHeight = GetHeight (""events.PDL"",
""myCircle"");"
strCode = strCode & vbCrLf & "lHeight = lHeight+5;" & vbCrLf &
"check = "
strCode = strCode & "SetHeight(""events.PDL"",
""myCircle"",lHeight);"
strCode = strCode & vbCrLf & "//Return-Type: Void"
Set objCircle =
ActiveDocument.HMIObjects.AddHMIObject("myCircle", "HMICircle")
Set objButton =
ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objCircle
.Top = 100
.Left = 100
.BackColor = RGB(255, 0, 0)
End With
With objButton
.Top = 10
.Left = 10
.Text = "Increase height"
End With
'Configure directconnection:
Set objCScript =
objButton.Events(1).Actions.AddAction(hmiActionCreationTypeCScript
)
With objCScript
'
'Note: Replace "events.PDL" with your picturename
.SourceCode = strCode
End With
End Sub
```

4. Starten Sie die Prozedur mit <F5>.

Siehe auch

ScriptInfo-Objekt (Seite 2021)

Events-Objekt (Auflistung) (Seite 1936)

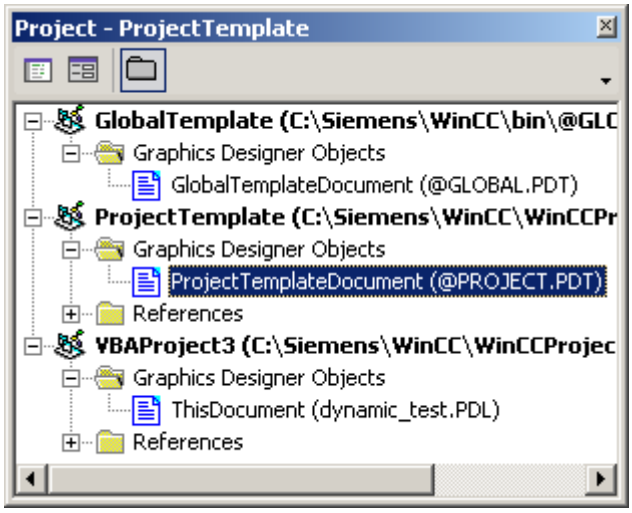
Actions-Objekt (Auflistung) (Seite 1884)

Projektierung von ereignisgesteuerten Aktionen mit VBA (Seite 1704)

So projektieren Sie eine VB-Aktion mit VBA an ein Ereignis

Vorgehensweise

1. Öffnen Sie im Graphics Designer den VBA-Editor (<Alt+F11> oder "Extras" > "Makros" > "Visual Basic Editor").
2. Öffnen Sie im Project Explorer das Dokument, in dem Sie den VBA-Code erstellen wollen:



3. Um eine VB-Aktion mit VBA an ein Ereignis zu projektieren, fügen Sie z.B. eine Prozedur "CreateVBActionToClickedEvent()" in das Dokument ein. In diesem Beispiel werden ein Button und ein Kreis in das aktive Bild eingefügt. In Runtime vergrößert sich der Kreisradius mit jedem Klick auf den Button:

```
Sub CreateVBActionToClickedEvent()
'VBA67
Dim objButton As HMIButton
Dim objCircle As HMICircle
Dim objEvent As HMIEvent
Dim objVBScript As HMIScriptInfo
Dim strCode As String
strCode = "Dim myCircle" & vbCrLf & "Set myCircle = "
strCode = strCode &
"HMIRuntime.ActiveScreen.ScreenItems(""Circle_VB"")"
strCode = strCode & vbCrLf & "myCircle.Radius = myCircle.Radius +
5"
Set objCircle =
ActiveDocument.HMIObjects.AddHMIObject("Circle_VB", "HMICircle")
Set objButton =
ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objCircle
.Top = 100
.Left = 100
.BackColor = RGB(255, 0, 0)
End With
With objButton
.Top = 10
.Left = 10
.Width = 120
.Text = "Increase Radius"
End With
'Define event and assign sourcecode:
Set objVBScript =
objButton.Events(1).Actions.AddAction(hmiActionCreationTypeVBScrip
t)
With objVBScript
.SourceCode = strCode
End With
End Sub
```

4. Starten Sie die Prozedur mit <F5>.

Siehe auch

Actions-Objekt (Auflistung) (Seite 1884)

ScriptInfo-Objekt (Seite 2021)

Events-Objekt (Auflistung) (Seite 1936)

Projektierung von ereignisgesteuerten Aktionen mit VBA (Seite 1704)

3.3.5.4 Bearbeiten von Triggern

Bearbeiten von Triggern

Einleitung

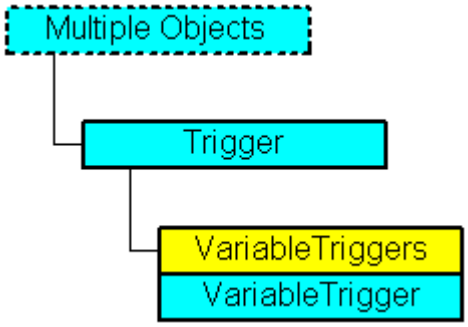
Trigger verwenden Sie beim Dynamisieren von Grafikobjekten und zum Auslösen von Aktionen an Objekteigenschaften. Trigger können z.B. sein:

- Variablen: Änderung, Über- oder Unterschreitung eines Variablenwertes.
- Standardzyklus: Zyklische Ausführung der Aktion. Die Zyklen können gewählt werden zwischen 250 ms und 1 h. Zusätzlich stehen Ihnen selbst definierte Anwenderzyklen zur Verfügung.
- Bildzyklus: Als Trigger wird ein zyklischer Trigger verwendet. Dieser Zyklus bietet Ihnen die Möglichkeit zentral die Zyklen aller in einem Bild projektierten Aktionen, Variablenanbindungen und Dynamik-Dialogen festzulegen.
- Fensterzyklus: Als Trigger wird ein zyklischer Trigger verwendet. Dieser Wert gilt für alle im Bildfenster verwendeten Aktionen, Variablenanbindungen und Dynamik-Dialogen, die mit Triggerart "Fensterzyklus" projektiert wurden.

Wenn Sie eine Aktion projektieren, die auf ein Ereignis an einem Grafikobjekt reagiert, ist das auslösende Ereignis der Trigger.

Trigger mit VBA projektieren

Um einen Trigger mit VBA zu projektieren, verwenden Sie das Trigger-Objekt. Wenn eine Variable als Trigger verwendet soll, verwenden Sie das VariableTrigger-Objekt:



Die Triggerart legen Sie mit der Type-Eigenschaft fest. Wenn Sie eine Variable als Trigger projektieren, verwenden Sie die VariableTriggers-Eigenschaft.

Siehe auch

Beispiele zum Bearbeiten von Triggern mit VBA (Seite 1713)

VariableTrigger-Objekt (Seite 2060)

Trigger-Objekt (Seite 2047)

ScriptInfo-Objekt (Seite 2021)

Beispiele zum Bearbeiten von Triggern mit VBA

Einleitung

Die folgenden vier Beispiele zeigen, wie Sie mit VBA folgende Trigger anlegen:

- Standardzyklus
- Variable
- Bildzyklus
- Fensterzyklus

In allen Beispielen wird ein Kreis in das aktive Bild eingefügt, dessen Radius mit einer VB-Aktion dynamisiert wird.

Das Vorgehen zum Dynamisieren einer Eigenschaft mit Variablenanbindung finden Sie unter "Eigenschaft mit Variablenanbindung dynamisieren" in dieser Dokumentation.

Beispiel 1: Standardzyklus

```
Sub DynamicWithStandardCycle()
'VBA68
Dim objVBScript As HMIScriptInfo
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_Standard", "HMICircle")
Set objVBScript = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeVBScript)
With objVBScript
.Trigger.Type = hmiTriggerTypeStandardCycle
'"CycleType"-specification is necessary:
.Trigger.CycleType = hmiCycleType_10s
.Trigger.Name = "VBA_StandardCycle"
.SourceCode = ""
End With
End Sub
```

Beispiel 2: Variable

```
Sub DynamicWithVariableTriggerCycle()
'VBA69
Dim objVBScript As HMIScriptInfo
Dim objVarTrigger As HMIVariableTrigger
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_VariableTrigger",
"HMICircle")
Set objVBScript = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeVBScript)
With objVBScript
Set objVarTrigger = .Trigger.VariableTriggers.Add("VarTrigger", hmiVariableCycleType_10s)
.SourceCode = ""
End With
```

```
End Sub
```

Beispiel 3: Bildzyklus

```
Sub DynamicWithPictureCycle()
'VBA70
Dim objVBScript As HMIScriptInfo
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_Picture", "HMICircle")
Set objVBScript = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeVBScript)
With objVBScript
.Trigger.Type = hmiTriggerTypePictureCycle
.Trigger.Name = "VBA_PictureCycle"
.SourceCode = ""
End With
End Sub
```

Beispiel 4: Fensterzyklus

```
Sub DynamicWithWindowCycle()
'VBA71
Dim objVBScript As HMIScriptInfo
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_Window", "HMICircle")
Set objVBScript = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeVBScript)
With objVBScript
.Trigger.Type = hmiTriggerTypeWindowCycle
.Trigger.Name = "VBA_WindowCycle"
.SourceCode = ""
End With
End Sub
```

Siehe auch

VariableTrigger-Objekt (Seite 2060)
Trigger-Objekt (Seite 2047)
ScriptInfo-Objekt (Seite 2021)
Bearbeiten von Triggern (Seite 1712)

3.3.6 Event-Handling

Einleitung

Im Graphics Designer treten bei bestimmten Aktionen (z.B. beim Öffnen eines Bildes) Ereignisse ein. Auf ein Ereignis können Sie mit einem vordefinierten VBA-Event-Handler reagieren, um Anweisungen auszuführen.

Die Ereignisse treten nur während der Projektierung im Graphics Designer ein und sind in Runtime nicht verfügbar. Die Ereignisse dürfen nicht mit den Ereignissen (z.B. Mausklick, Eigenschaftsänderung) an Grafik-Objekten und Bildern verwechselt werden.

Hinweis

Wenn der Graphics Designer geöffnet ist, werden Ereignisse auch von anderen Editoren

ausgelöst. Dies betrifft z.B. das Ändern von Bildeigenschaften im WinCCExplorer. Schließen Sie den Graphics Designer, wenn Sie in anderen Editoren Änderungen an Bildern vornehmen. Damit wird das ungewollte Ausführen von Ereignissen verhindert.

Hinweis

Wenn Sie ein Bild im Graphics Designer öffnen, wird nicht nur das "DocumentOpened-Ereignis" des aktiven Bildes, sondern auch des "Project Template" und des "Global Template" ausgelöst. Der VBA-Code des "DocumentOpened-Ereignis" wird dadurch doppelt ausgeführt.

Sie müssen dieses Verhalten mit dem Event-Handler abfangen.

Prinzip

Beim Event-Handling gibt es Ereignisse mit und ohne Weiterleitung. Ein Ereignis mit Weiterleitung erkennen Sie am Parameter "CancelForwarding". Ein Ereignis ohne Weiterleitung besitzt diesen Parameter nicht. Tritt ein Ereignis ein, wird es an das aktive Bild gesendet und danach bis zum "Global Template" weitergeleitet.

Ein Ereignis mit Weiterleitung wird also standardmäßig über das Dokument "Project Template" bis zum Dokument "Global Template" weitergeleitet.

Weiterleitung unterbinden

Die Weiterleitung eines Ereignisses können Sie unterbinden, indem Sie im VBA-Event-Handler den Parameter "CancelForwarding" auf "True" setzen:

```
Sub Document_HMIObjectPropertyChanged(ByVal Property As IHMIProperty, CancelForwarding As
Boolean)
'VBA72
CancelForwarding = True
```

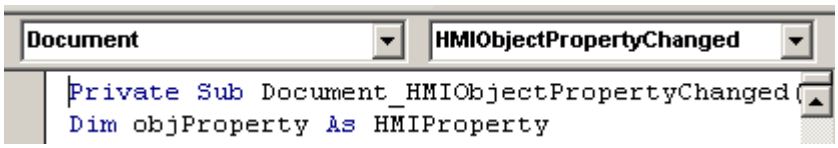
```
MsgBox "Object's property has been changed!"
End Sub
```

Bild- und anwendungsspezifische Ereignisse

Unabhängig von den oben getroffenen Aussagen zu Ereignissen mit und ohne Weiterleitung unterscheidet der Graphics Designer bild- und anwendungsspezifische Ereignisse:

Bildspezifische Ereignisse

Bildspezifische Ereignisse reagieren immer auf Aktionen, die sich im aktiven Bild im Graphics Designer ereignen. Solche Aktionen sind z.B. das Ändern von Objekteigenschaften oder das Speichern des aktiven Bildes. Eine Liste der verfügbaren bildspezifischen Ereignisse erhalten Sie, wenn Sie im VBA-Editor "Document" wählen:



Anwendungsspezifische Ereignisse

Anwendungsspezifische Ereignisse reagieren auf Aktionen, die sich in der Anwendung "Graphics Designer" ereignen. Solche Aktionen sind z.B. das Starten des Graphics Designer oder das Anlegen eines Objektes in der Bausteinbibliothek.

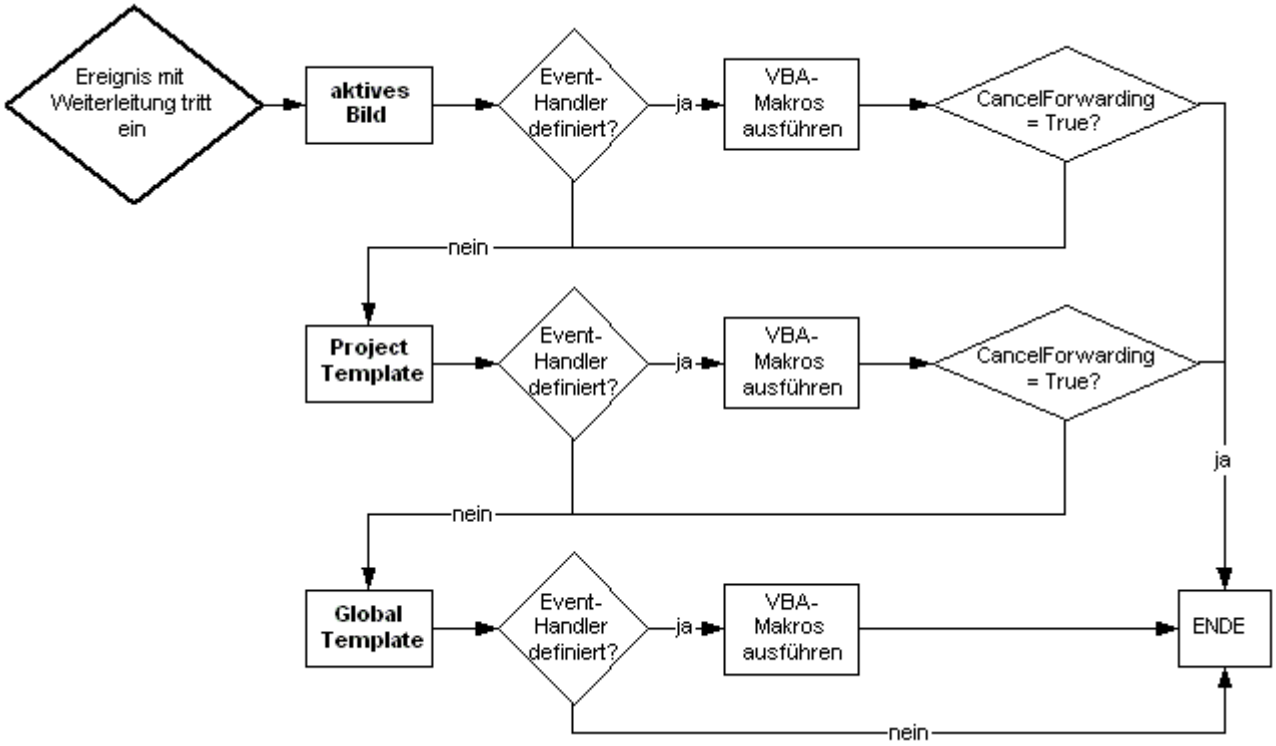
Um die anwendungsspezifischen Ereignisse verfügbar zu machen, schreiben Sie im VBA-Editor folgende Anweisung an den Anfang des Dokumentes (vorzugsweise das "Project Template" oder "Global Template"):

```
Dim WithEvents <Name> As grafexe.Application
```

Diese Anweisung bewirkt, dass nun auch die anwendungsspezifischen Ereignisse aus der Liste im Graphics Designer ausgewählt werden können:

Beispiel 1: Entritt eines Ereignisses mit Weiterleitung

Die Abbildung zeigt den Ablauf beim Eintritt eines Ereignisses mit Weiterleitung:

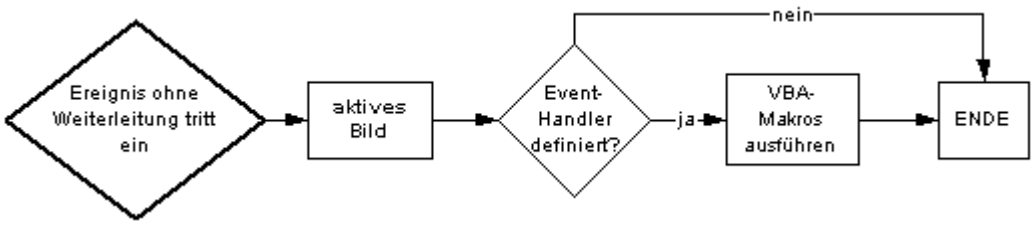


Hinweis

Es gibt Ereignisse, die sowohl bild- als auch anwendungsspezifisch sind (z.B. BeforeDocumentSave) sind. Tritt ein solches Ereignis ein, wird geprüft, ob der entsprechende anwendungsspezifische Event-Handler definiert wurde. Erst danach tritt der oben gezeigte Ablauf ein.

Beispiel 2: Eintritt eines Ereignisses ohne Weiterleitung

Die Abbildung zeigt den Ablauf beim Eintritt eines Ereignisses ohne Weiterleitung:



Event-Handling abschalten

Sie können das Event-Handling abschalten, indem Sie die Eigenschaft "DisableVBAEvents" des Application-Objektes auf "True" setzen.

Siehe auch

DisableVBAEvents-Eigenschaft (Seite 2164)

VBA-Code im WinCC-Projekt organisieren (Seite 1617)

3.3.7 Zugriff auf Fremdapplikationen mit VBA

3.3.7.1 Zugriff auf Fremdapplikationen mit VBA

Einleitung

Sie können mit VBA auf Programme zugreifen, die VBA unterstützen, z.B. die Produkte der Mircosoft Office-Familie. Damit haben Sie z.B. die Möglichkeit, Werte aus einer MS Excel-Tabelle auszulesen und diese dann Objekteigenschaften zuzuweisen.

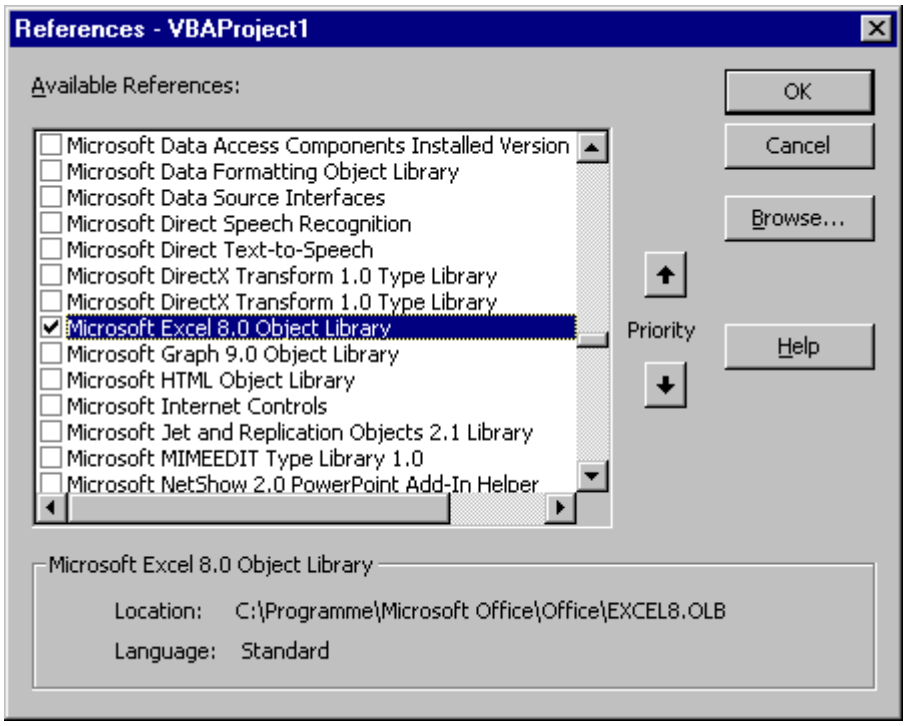
Hinweis

Keine direkte Verwendung von Unicode in Excel-VBA und Word-VBA

In Excel-VBA und Word-VBA können Unicode-Zeichen nur über die Funktion <ChrW(unicode-id) verwendet werden.

Fremdapplikation anmelden

Sie müssen eine Fremdapplikation im VBA-Editor einbinden, um deren Objektbibliothek verfügbar zu machen. Im VBA-Editor wählen dazu Sie im Menü "Tools" den Befehl "References". Im Dialog "References" wählen Sie dann die gewünschte Objektbibliothek aus:



Hinweis

Im Project Explorer des VBA-Editors müssen Sie die Fremdapplikation in allen Projekten einbinden, die auf die Fremdapplikation zugreifen sollen.

Siehe auch

Beispiel: Zugriff auf MS Excel mit VBA (Seite 1719)

3.3.7.2 Beispiel: Zugriff auf MS Excel mit VBA

Einleitung

Die folgenden drei Beispiele zeigen den Zugriff auf MS Excel. Damit die Beispiele funktionieren, muss die Objektbibliothek von MS Excel über eine Referenz eingebunden sein.

Hinweis

Keine direkte Verwendung von Unicode in Excel-VBA und Word-VBA

In Excel-VBA und Word-VBA können Unicode-Zeichen nur über die Funktion <ChrW(unicode-id) verwendet werden.

Beispiel 1

In diesem Beispiel wird die Default-Objektliste des Graphics Designer in einer Excel-Tabelle exportiert. Dabei werden die Objekteigenschaften berücksichtigt und ob diese dynamisierbar sind. Zusätzlich wird der VBA-Datentyp angezeigt.

```
Sub ExportDefObjListToXLS()
'VBA73
'Microsoft Excel Object Library needs to be referenced
Dim objGDApplication As grafexe.Application
Dim objHMIObject As grafexe.HMIObject
Dim objProperty As grafexe.HMIProperty
Dim objXLS As Excel.Application
Dim objWSheet As Excel.Worksheet
Dim objWBook As Excel.Workbook
Dim rngSelection As Excel.Range
Dim lRow As Long
Dim lRowGroupStart As Long

'define local errorhandler
On Local Error GoTo LocErrTrap

'Set references to the applications Excel and GraphicsDesigner
Set objGDApplication = grafexe.Application
Set objXLS = New Excel.Application

'Create workbook
Set objWBook = objXLS.Workbooks.Add()
objWBook.SaveAs objGDApplication.ApplicationDataPath & "DefaultObjekte.xls"

'Adds new worksheet to the new workbook
Set objWSheet = objWBook.Worksheets.Add
objWSheet.Name = "DefaultObjekte"
lRow = 1

'Every object of the DefaultHMIObjects-collection will be written
'to the worksheet with their objectproperties.
'For better overview the objects will be grouped.
For Each objHMIObject In objGDApplication.DefaultHMIObjects
DoEvents
objWSheet.Cells(lRow, 1).value = objHMIObject.ObjectName
objWSheet.Cells(lRow, 2).value = objHMIObject.Type
lRow = lRow + 1
lRowGroupStart = lRow
```

```
For Each objProperty In objHMIObject.Properties
'Write displayed name and automationname of property
'into the worksheet
objWSheet.Cells(lRow, 2).value = objProperty.DisplayName
objWSheet.Cells(lRow, 3).value = objProperty.Name
'Write the value of property, datatype and if their dynamicable
'into the worksheet
If Not IsEmpty(objProperty.value) Then _
        objWSheet.Cells(lRow, 4).value = objProperty.value
objWSheet.Cells(lRow, 5).value = objProperty.IsDynamicable
objWSheet.Cells(lRow, 6).value = TypeName(objProperty.value)
objWSheet.Cells(lRow, 7).value = VarType(objProperty.value)
lRow = lRow + 1
Next objProperty
'Select and groups the range of object-properties in the worksheet
Set rngSelection = objWSheet.Range(objWSheet.Rows(lRowGroupStart), _
                   objWSheet.Rows(lRow - 1))
rngSelection.Select
rngSelection.Group
Set rngSelection = Nothing
'Insert empty row
lRow = lRow + 1
Next objHMIObject
objWSheet.Columns.AutoFit
Set objWSheet = Nothing
objWBook.Save
objWBook.Close
Set objWBook = Nothing
objXLS.Quit
Set objXLS = Nothing
Set objGDApplication = Nothing
Exit Sub
LocErrTrap:
MsgBox Err.Description, , Err.Source
Resume Next
End Sub
```

Beispiel 2

In diesem Beispiel werden alle Objekte des aktiven Bildes in eine Excel-Tabelle exportiert. Berücksichtigt werden dabei die Eigenschaften Position X, Position Y, Breite, Höhe und Ebene:

```
Sub ExportObjectListToXLS()
'VBA74
Dim objGDApplication As grafexe.Application
Dim objDoc As grafexe.Document
Dim objHMIObject As grafexe.HMIObject
Dim objProperty As grafexe.HMIProperty
Dim objXLS As Excel.Application
Dim objWSheet As Excel.Worksheet
Dim objWBook As Excel.Workbook
Dim lRow As Long
```

```
'Define local errorhandler
On Local Error GoTo LocErrTrap

'Set references on the applications Excel and GraphicsDesigner
Set objGDApplication = grafexe.Application
Set objDoc = objGDApplication.ActiveDocument
Set objXLS = New Excel.Application

'Create workbook
Set objWBook = objXLS.Workbooks.Add()
objWBook.SaveAs objGDApplication.ApplicationDataPath & "Export.xls"

'Create worksheet in the new workbook and write headline
'The name of the worksheet is equivalent to the documents name
Set objWSheet = objWBook.Worksheets.Add
objWSheet.Name = objDoc.Name
objWSheet.Cells(1, 1) = "Objektname"
objWSheet.Cells(1, 2) = "Objekttyp"
objWSheet.Cells(1, 3) = "ProgID"
objWSheet.Cells(1, 4) = "Position X"
objWSheet.Cells(1, 5) = "Position Y"
objWSheet.Cells(1, 6) = "Breite"
objWSheet.Cells(1, 7) = "Höhe"
objWSheet.Cells(1, 8) = "Ebene"
lRow = 3

'Every objects will be written with their objectproperties width,
'height, pos x, pos y and layer to Excel. If the object is an
'ActiveX-Control the ProgID will be also exported.
For Each objHMIObject In objDoc.HMIObjects
DoEvents
objWSheet.Cells(lRow, 1).value = objHMIObject.ObjectName
objWSheet.Cells(lRow, 2).value = objHMIObject.Type
If UCase(objHMIObject.Type) = "HMIACTIVEXCONTROL" Then
objWSheet.Cells(lRow, 3).value = objHMIObject.ProgID
End If
objWSheet.Cells(lRow, 4).value = objHMIObject.Left
objWSheet.Cells(lRow, 5).value = objHMIObject.Top
objWSheet.Cells(lRow, 6).value = objHMIObject.Width
objWSheet.Cells(lRow, 7).value = objHMIObject.Height
objWSheet.Cells(lRow, 8).value = objHMIObject.Layer
lRow = lRow + 1
Next objHMIObject
objWSheet.Columns.AutoFit
Set objWSheet = Nothing
objWBook.Save
objWBook.Close
Set objWBook = Nothing
objXLS.Quit
Set objXLS = Nothing
Set objDoc = Nothing
Set objGDApplication = Nothing
Exit Sub
LocErrTrap:
MsgBox Err.Description, , Err.Source
Resume Next
```

```
End Sub
```

Beispiel 3

In diesem Beispiel werden Objekte aus der im Beispiel 2 angelegten Excel-Tabelle importiert. Berücksichtigt werden dabei die Eigenschaften Position X, Position Y, Breite, Höhe und Ebene:

```
Sub ImportObjectListFromXLS()
'VBA75
Dim objGDApplication As grafexe.Application
Dim objDoc As grafexe.Document
Dim objHMIObject As grafexe.HMIObject
Dim objXLS As Excel.Application
Dim objWSheet As Excel.Worksheet
Dim objWBook As Excel.Workbook
Dim lRow As Long
Dim strWorkbookName As String
Dim strWorksheetName As String
Dim strSheets As String

'define local errorhandler
On Local Error GoTo LocErrTrap

'Set references on the applications Excel and GraphicsDesigner
Set objGDApplication = Application
Set objDoc = objGDApplication.ActiveDocument
Set objXLS = New Excel.Application

'Open workbook. The workbook have to be in datapath of GraphicsDesigner
strWorkbookName = InputBox("Name of workbook:", "Import of objects")
Set objWBook = objXLS.Workbooks.Open(objGDApplication.ApplicationDataPath &
strWorkbookName)
If objWBook Is Nothing Then
MsgBox "Open workbook fails!" & vbCrLf & "This function is cancled!", vbCritical, "Import
od objects"
Set objDoc = Nothing
Set objGDApplication = Nothing
Set objXLS = Nothing
Exit Sub
End If

'Read out the names of all worksheets contained in the workbook
For Each objWSheet In objWBook.Sheets
strSheets = strSheets & objWSheet.Name & vbCrLf
Next objWSheet
strWorksheetName = InputBox("Name of table to import:" & vbCrLf & strSheets, "Import of
objects")
Set objWSheet = objWBook.Sheets(strWorksheetName)
lRow = 3

'Import the worksheet as long as in actual row the first column is empty.
'Add with the outreaded data new objects to the active document and
'assign the values to the objectproperties
```

```
With objWSheet
While (.Cells(lRow, 1).value <> vbNullString) And (Not IsEmpty(.Cells(lRow, 1).value))
'Add the objects to the document as its objecttype,
'do nothing by groups, their have to create before.
If (UCase(.Cells(lRow, 2).value) = "HMIGROUP") Then
Else
  If (UCase(.Cells(lRow, 2).value) = "HMIACTIVEXCONTROL") Then
    Set objHMIObject = objDoc.HMIObjects.AddActiveXControl(.Cells(lRow,
1).value, .Cells(lRow, 3).value)
Else
    Set objHMIObject = objDoc.HMIObjects.AddHMIObject(.Cells(lRow, 1).value, .Cells(lRow,
2).value)
  End If
  objHMIObject.Left = .Cells(lRow, 4).value
  objHMIObject.Top = .Cells(lRow, 5).value
  objHMIObject.Width = .Cells(lRow, 6).value
  objHMIObject.Height = .Cells(lRow, 7).value
  objHMIObject.Layer = .Cells(lRow, 8).value
End If
Set objHMIObject = Nothing
lRow = lRow + 1
Wend
End With
objWBook.Close
Set objWBook = Nothing
objXLS.Quit
Set objXLS = Nothing
Set objDoc = Nothing
Set objGDApplication = Nothing
Exit Sub
LocErrTrap:
MsgBox Err.Description, , Err.Source
Resume Next
End Sub
```

3.4 AddIns

3.4.1 AddIns

Einleitung

Ein AddIn ist ein Code, der nicht einsehbar und als DLL abgespeichert ist. Mit AddIns können neue Funktionen zur Verfügung gestellt werden, indem die entsprechende DLL im Betriebssystem registriert und im Graphics Designer geladen wird.

Als Anwender profitieren Sie von AddIns, die für den jeweiligen Einsatzzweck maßgeschneiderte Funktionen bieten. Wenn Sie z.B. an verschiedenen Projektierungsrechnern arbeiten und häufig VBA-Makros verwenden, können Sie diese VBA-Makros in einem oder mehreren AddIns zusammenfassen. Beim Wechsel des Projektierungsrechners müssen Sie nur das AddIn kopieren, um am neuen Arbeitsplatz auf die gewohnten Funktionen zurückgreifen zu können.

Als Entwickler können Sie mit der Programmbibliothek des Graphics Designer in einer Entwicklungsumgebung wie z.B. MS Visual Studio 6.0 AddIns erstellen und so Ihren Code vor Eingriffen schützen.

AddIns im Graphics Designer

Im Graphics Designer können Sie alle AddIns verwenden, die für den Graphics Designer entwickelt wurden und im Betriebssystem des Projektierungsrechners registriert sind.

Ein AddIn können Sie beim Start des Graphics Designer automatisch laden, wenn Sie die Funktionen des AddIns häufig benötigen.

Wenn Sie die Funktionen eines AddIns nicht mehr benötigen, können Sie es jederzeit wieder entladen.

Siehe auch

So laden Sie ein AddIn im Graphics Designer (Seite 1727)

Beispiel: Erstellung von AddIns (Seite 1729)

Einbinden von AddIns (Seite 1725)

3.4.2 Einbinden von AddIns

Einleitung

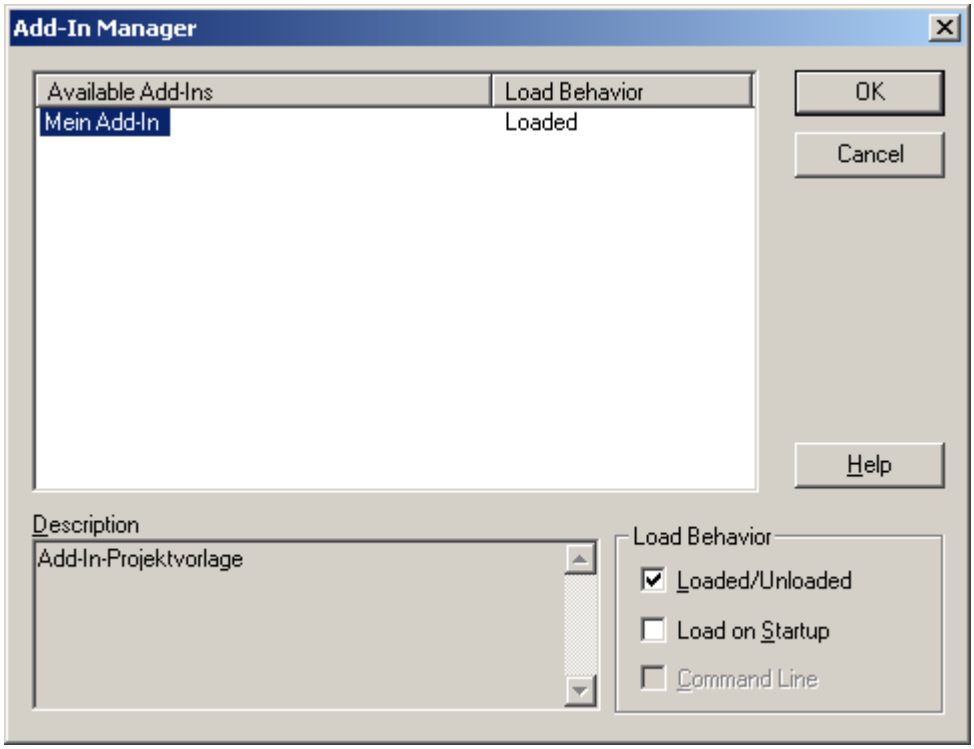
Im Graphics Designer verwenden Sie den AddIn Manager, um das Ladeverhalten von AddIns festzulegen, die im Graphics Designer verwendet werden können.

Voraussetzungen

- Ein AddIn muss im Betriebssystem registriert sein, z. B. über die Eingabeaufforderung mit dem Befehl "regsvr32 filename.dll".
- Um VBA-AddIns registrieren zu können, muss "Microsoft Visual Basic for Applications" installiert sein. Die Installation erhalten Sie über folgende Wege:
  - Microsoft Office: Bei der Installation von Microsoft Office-Produkten, z. B. MS Excel oder MS Word, wird Visual Basic for Applications automatisch mitinstalliert.
  - Nachinstallation aus dem Microsoft Office Setup: Im Microsoft Office Setup können Sie über die benutzerdefinierte Installation gezielt nur Visual Basic for Applications installieren.
  - Download der VBA Runtime-Umgebung: Microsoft bietet über die folgenden Links die VBA Runtime-Umgebung zum Download an:
    - Datei "VBRun60.exe" für V6.0: http://support.microsoft.com/kb/192461/ (http://support.microsoft.com/kb/192461/)
    - Datei "VBRun60sp6.exe" für V6.0 SP6: http://support.microsoft.com/kb/290887/ (http://support.microsoft.com/kb/290887/)

AddIn Manager starten

Um den AddIn Manager zu starten, wählen Sie im Graphics Designer den Befehl "Makros > AddIn Manager":



AddIn automatisch laden

Wenn das AddIn neue Funktionen enthält, die Sie im Graphics Designer immer benötigen, können Sie das AddIn automatisch beim Öffnen des Graphics Designer laden.

Dazu wählen Sie das AddIn im AddIn-Manager aus und aktivieren das Kontrollkästchen "Load on Startup".

Hinweis

Je nach Programmierung des AddIns kann die im AddIn enthaltene Funktion auch im Menü "Extras > Makros > AddIns" eingetragen werden. Die Funktion können Sie dann über Mausklick starten.

AddIn manuell laden oder entladen

Sie können ein AddIn auch manuell laden, wenn Sie dessen Funktionen nur zu bestimmten Zwecken (z.B. Prüfroutinen) benötigen.

Um ein AddIn manuell zu laden oder zu entladen, wählen Sie das AddIn im AddIn-Manager aus und aktivieren das Kontrollkästchen "Loaded/Unloaded".

Siehe auch

So laden Sie ein AddIn im Graphics Designer (Seite 1727)

Beispiel: Erstellung von AddIns (Seite 1729)

AddIns (Seite 1725)

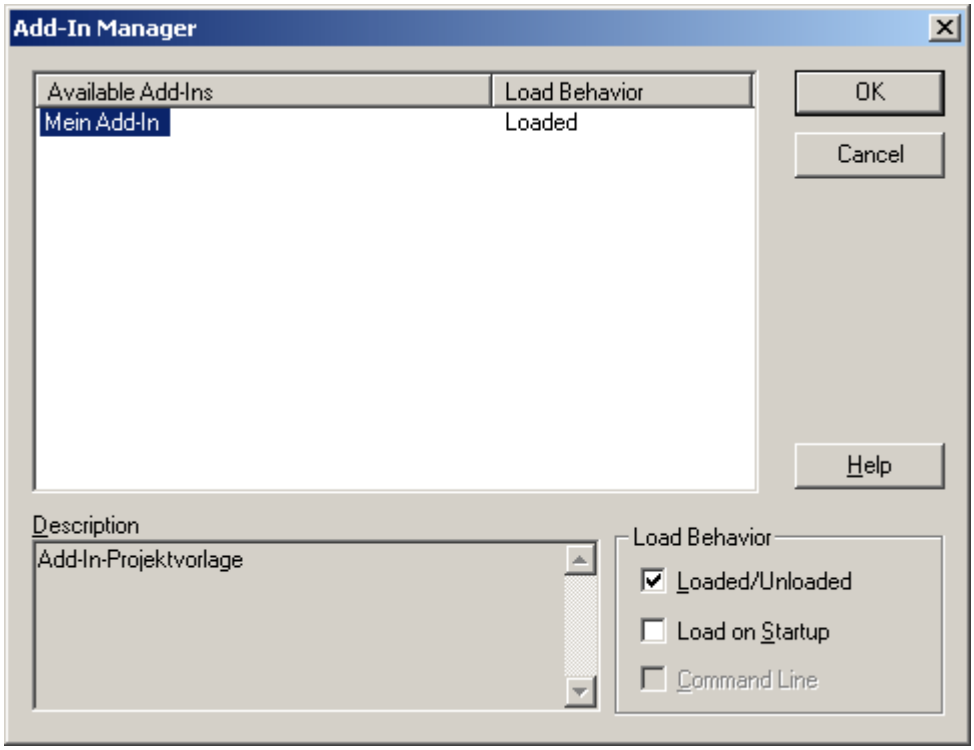
3.4.3 So laden Sie ein AddIn im Graphics Designer

Voraussetzung

Ein AddIn muss im Betriebssystem registriert sein, z. B. über die Eingabeaufforderung mit dem Befehl "regsvr32 filename.dll".

Vorgehensweise

1. Starten Sie den Graphics Designer und öffnen Sie das Projekt, in dem Sie das AddIn einbinden wollen.
2. Wählen Sie den Menübefehl "Extras" > "AddIn Manager", um den AddIn Manager aufzurufen.
   Der AddIn Manager wird geöffnet. Unter "Available Add-Ins" werden alle verfügbaren AddIns aufgelistet sowie deren derzeitiger Ladezustand angezeigt:



3. Legen Sie für jedes AddIn fest, ob und wann es geladen werden soll. Dazu wählen Sie das gewünschte AddIn aus und aktivieren unter "Load Behavior" das entsprechende Kontrollkästchen.
4. Um ein AddIn zu entladen, wählen Sie das gewünschte AddIn aus und deaktivieren Sie unter "Load Behavior" das Kontrollkästchen "Load/Unload".
5. Klicken Sie auf OK.

Ergebnis

Abhängig von der Programmierung des AddIns wird die im AddIn enthaltene Funktion entweder im Menü "Extras" > "AddIns" aufgelistet oder reagiert auf einen EventHandler im Graphics Designer.

Wenn das AddIn beim Eintreten eines EventHandlers gestartet wird (z.B. Started-Ereignis), sollte für das AddIn das Kontrollkästchen "On Startup" aktiviert sein.

Siehe auch

Einbinden von AddIns (Seite 1725)

AddIns (Seite 1725)

3.4.4 Beispiel: Erstellung von AddIns

3.4.4.1 Beispiel: Erstellung von AddIns

Einleitung

Zum Erstellen von AddIns finden Sie in dieser Dokumentation ein Beispiel für Visual Basic 6.0, das ein lauffähiges AddIn für die Verwendung im Graphics Designer erzeugt.

Voraussetzungen

Auf dem Projektierungsrechner muss MS Visual Studio 6.0 installiert sein.

Sie sollten Programmiererfahrung besitzen, wenn Sie den Beispielcode als Grundlage für das Entwickeln eigener AddIns verwenden wollen.

Beispiel: Programmvorlage für Visual Basic 6.0

Verwenden Sie den Event-Handler "AddInInstance_OnConnection", um eine Instanz des Graphics Designer zu erzeugen. Das Bekanntmachen der Anwendung ist unbedingt notwendig, damit das AddIn auf den Graphics Designer zugreifen kann.

Siehe auch

Beispiel: AddIn mit Visual Basic 6.0 erstellen (Seite 1729)

Einbinden von AddIns (Seite 1725)

3.4.4.2 Beispiel: AddIn mit Visual Basic 6.0 erstellen

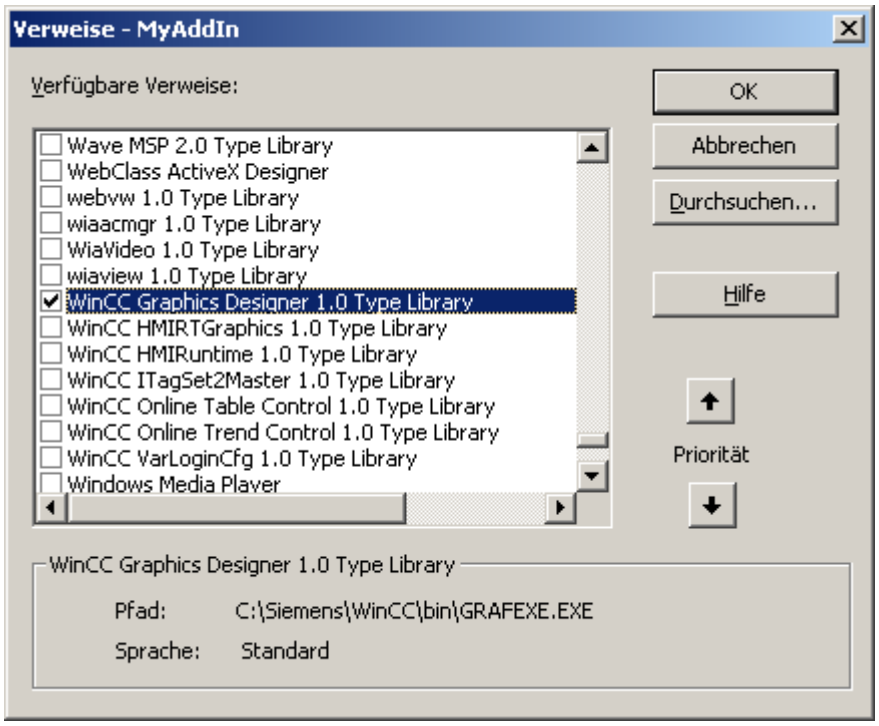
Einleitung

Der in diesem Beispiel enthaltene Programmcode erzeugt die Datei "MyAddIn.DLL". Damit das AddIn im Graphics Designer funktioniert, müssen Sie im AddIn Manager des Graphics Designer das Kontrollkästchen "Load on Startup" für dieses AddIn aktiveren. Sie können dazu auch im AddIn die Funktion "LoadOnStartup" verwenden.

Beim Öffnen des Graphics Designer erzeugt das AddIn ein benutzerdefiniertes Menü. Zusätzlich können Sie im Menü "Extras" > "AddIns" die im AddIn enthaltene Funktion aufrufen.

Voraussetzung

Damit Sie aus dem Beispielcode ein lauffähiges AddIn erzeugen können, muss auf Ihrem Rechner "MS Visual Studio 6.0" installiert sein. Des weiteren müssen Sie im "MS Visual Studio 6.0" die "WinCC Graphics Designer 1.0 Type Library" referenziert haben:



Vorgehen

1. Öffnen Sie "MS Visual Studio 6.0" und legen Sie ein neues Projekt an. Wählen Sie im Dialog "Neues Projekt" den Eintrag "AddIn" aus und klicken Sie auf OK, um das Projekt anzulegen.
2. Öffnen Sie im Project Explorer den Ordner "Designer" und doppelklicken Sie den Eintrag "Connect". Der Dialog "Connect (AddIn Designer)" wird geöffnet.
3. Wählen Sie unter "Anwendung" den Eintrag "Graphics Designer" aus und wählen Sie das "anfänglich Ladeverhalten" des AddIns aus. Schließen Sie den Dialog "Connect (AddIn Designer)".
4. Öffnen Sie im Project Explorer den Ordner "Designer" und wählen Sie für den Eintrag "Connect" aus dem Kontextmenü den Befehl "Code anzeigen".

5. Ersetzen Sie den kompletten Programmcode durch den folgenden Programmcode:

```
Option Explicit
'-----------------------------------------------------
'Member Variables
'-----------------------------------------------------
'Reference to the AddIn Connection
Dim WithEvents ThisAddIn As grafexe.AddinHook
'Reference to the Grafics Designer Application
Dim WithEvents GrafApp As grafexe.Application
'-----------------------------------------------------
'WithEvents AddInInstance IDTExtensibility2 (automatic)
'-----------------------------------------------------
'-----------------------------------------------------
'This method connects the AddIn To the Graphic Designer Application
'-----------------------------------------------------
Private Sub AddinInstance_OnConnection(ByVal Application As
Object, _
                                       ByVal ConnectMode As
AddInDesignerObjects.ext_ConnectMode, _
                                       ByVal AddInInst As Object,
custom() As Variant)
On Error GoTo AddinInstance_OnConnection_Error

    '-----------------------------------------------------
   ' Hook up to the Graphics Designer application.IAddInHookEvents
interface.
    ' It is necessary referencing the application this AddIn hooks
up to
    '-----------------------------------------------------
    Dim GDApplication As grafexe.Application
    Set GDApplication = Application
    If (Not GDApplication Is Nothing) Then
        '-----------------------------------------------------
        ' Explanation on filters ( first parameter to
AddIns.Attach() )
        '
        ' sbAddInFilterExecute : AddIn is not shown in the AddIn-
Menu
       '                       (ThisAddIn_Execute() is NOT called)
        ' sbAddInFilterNone    : AddIn is shown in the AddIn-Menu
and by
       '                       clicking on the AddIn's menu entry
ThisAddIn_Execute()
        '                        is called (see the figure below)
        '-----------------------------------------------------
        Set ThisAddIn =
GDApplication.Addins.Attach(sbAddinFilterNone, "Create Rectangle")
        Set GrafApp = GDApplication

        RegisterApplicationMenus
    End If
```

```
    Exit Sub

AddinInstance_OnConnection_Error:
    MsgBox Err.Description
End Sub
'------------------------------------------------------------
'This method removes the Add-In from VB by event disconnect
'------------------------------------------------------------
Private Sub AddinInstance_OnDisconnection(ByVal RemoveMode As
AddInDesignerObjects.ext_DisconnectMode, _
                                          custom() As Variant)
On Error GoTo AddinInstance_OnDisconnection_Error

    If (RemoveMode = ext_dm_UserClosed) Then
        RemoveApplicationMenus
    End If

    ' ------------------------------------------------------------
    ' Release reference to IAddInHookEvents interface - Important
    ' ------------------------------------------------------------
    Set ThisAddIn = Nothing
    Set GrafApp = Nothing
    Exit Sub

AddinInstance_OnDisconnection_Error:
    MsgBox Err.Description
End Sub
'------------------------------------------------------------------
'This method describes the 2nd way to make AddIn functions
available in Graphics Designer
'
'By adding an application menu in Graphics Designer the menu click
events can be catched by
'the MenuItemClicked event from the application object
'------------------------------------------------------------------
Private Sub RegisterApplicationMenus()
    Dim objDocMenu As HMIMenu
    Dim objMenuItem As HMIMenuItem
   Set objDocMenu = GrafApp.CustomMenus.InsertMenu(1, "DocMenu1",
"Doc_Menu_1")
    Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(1,
"dmItem1_1", "My first menu entry")
    Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(2,
"dmItem1_2", "My second menu entry")

    Set objMenuItem = Nothing
    Set objDocMenu = Nothing
End Sub
'------------------------------------------------------------------
'This method removes the AddIn menus available in Graphics Designer
'------------------------------------------------------------------
Private Sub RemoveApplicationMenus()
```

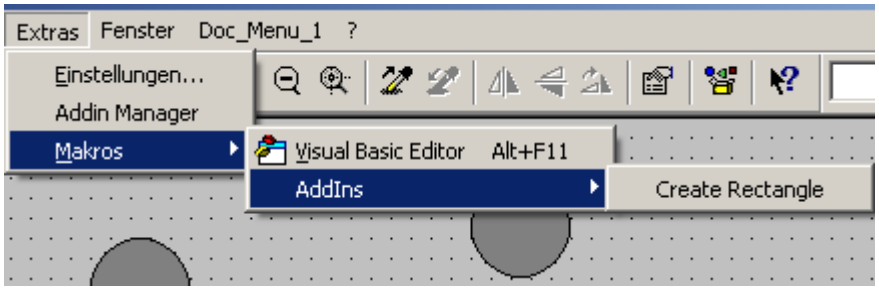
```
    Dim objDocMenu As HMIMenu
    Dim objMenuItem As HMIMenuItem

    For Each objMenuItem In
GrafApp.CustomMenus("DocMenu1").MenuItems
        Set objMenuItem = Nothing
    Next objMenuItem

    GrafApp.CustomMenus("DocMenu1").Delete

    Set objMenuItem = Nothing
    Set objDocMenu = Nothing
End Sub
Private Sub AddinInstance_Terminate()
    ' ------------------------------------------------
    ' Release reference to IAddInHookEvents interface - Important
    ' ------------------------------------------------
    Set ThisAddIn = Nothing
    Set GrafApp = Nothing
End Sub
Private Sub GrafApp_MenuItemClicked(ByVal MenuItem As
grafexe.IHMIMenuItem)

    Select Case MenuItem.Key
        Case "dmItem1_1"
            TestCall1
        Case "dmItem1_2"
            TestCall2
        Case Else
            Debug.Assert False
    End Select
End Sub
'------------------------------------------------------------
'You can call both of the following procedures by clicking the menu
item in the "DocMenu1"
'------------------------------------------------------------
Sub TestCall1()
    Call MsgBox("AddIn Menu: dmItem1_1 Clicked", vbInformation,
"GrafApp_MenuItemClicked")
End Sub
Sub TestCall2()
    Call MsgBox("AddIn Menu: dmItem1_2 Clicked", vbInformation,
"GrafApp_MenuItemClicked")
End Sub
'------------------------------------------------------------
'Registering an AddInHook creates an object which event
'can be executed by clicking "Extras\Macros\AddIns\<Registered
Name>"
'------------------------------------------------------------
Private Sub ThisAddIn_Execute()
    MsgBox ("AddIn : Execute! Will create a new Rectangle now")
    Dim NewShape As HMIObject
```

```
    Set NewShape =
GrafApp.ActiveDocument.HMIObjects.AddHMIObject("Rectangle1",
"HMIRectangle")
    With NewShape
        .Top = 40
        .Left = 40
        .BackColor = 255
    End With
    MsgBox (NewShape.ObjectName)
End Sub
```

6. Erzeugen Sie das AddIn und laden Sie es im Graphics Designer.

Ergebnis

Beim nächsten Öffnen wird im Graphics Designer das benutzerdefinierte Menü "DocMenu1" eingefügt. Im Menü "Extras" > "AddIns" erscheint der Eintrag "Create Rectangle", der ein Rechteck in das aktive Bild einfügt:



Siehe auch

So laden Sie ein AddIn im Graphics Designer (Seite 1727)

Beispiel: Erstellung von AddIns (Seite 1729)

3.5 VBA Referenz

3.5.1 Das Objektmodell des Graphics Designer

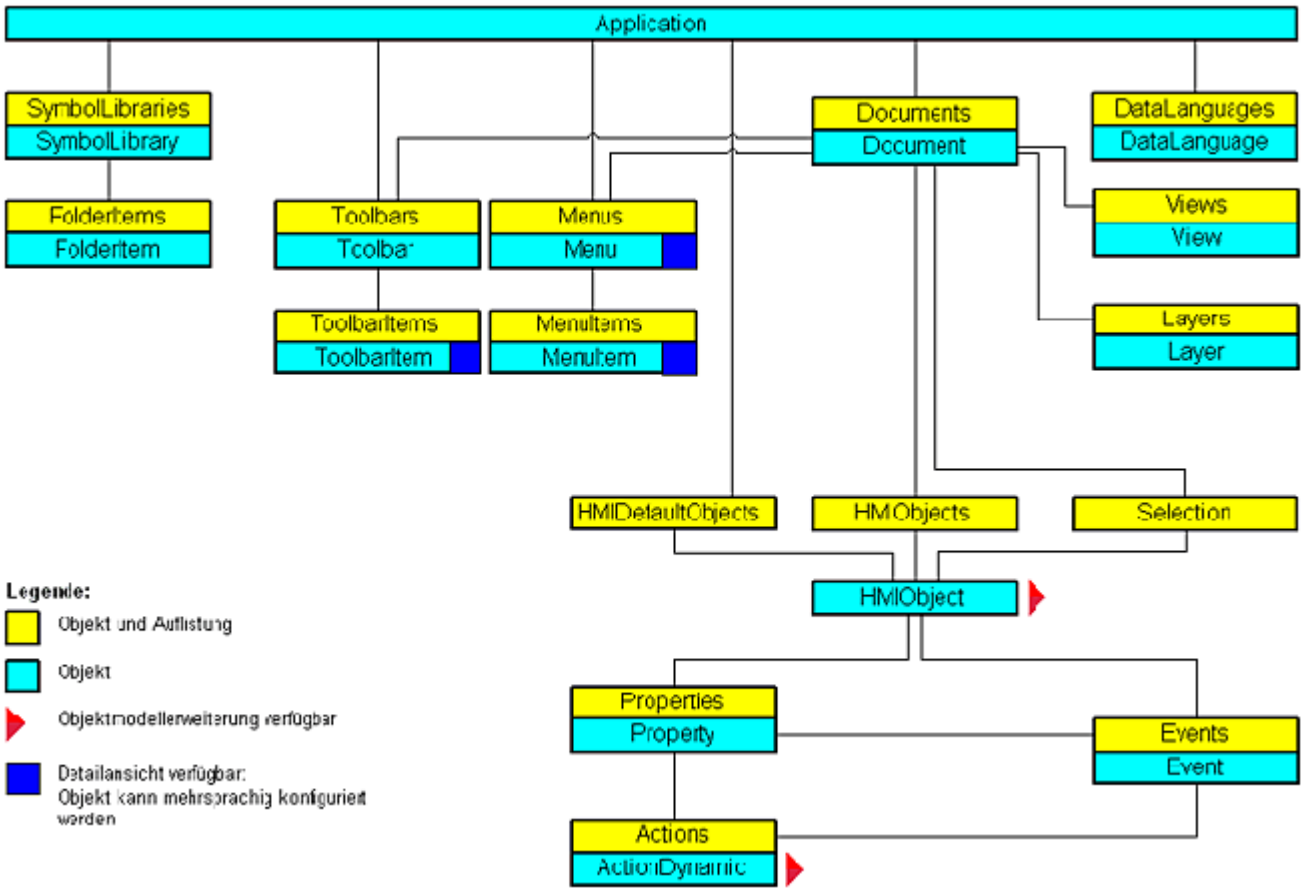
3.5.1.1 VBA-Referenz

VBA-Objektmodell

Wenn Sie mit der Maus auf einen Objektnamen klicken, erhalten Sie eine detaillierte Beschreibung.

Hinweis

Im Folgenden wird auf das Präfix "HMI" in den Beschreibungen verzichtet. Beachten Sie, dass Sie im Code die Objekte mit dem Präfix "HMI" verwenden müssen, z.B. "HMISymbolLibrary".



Siehe auch

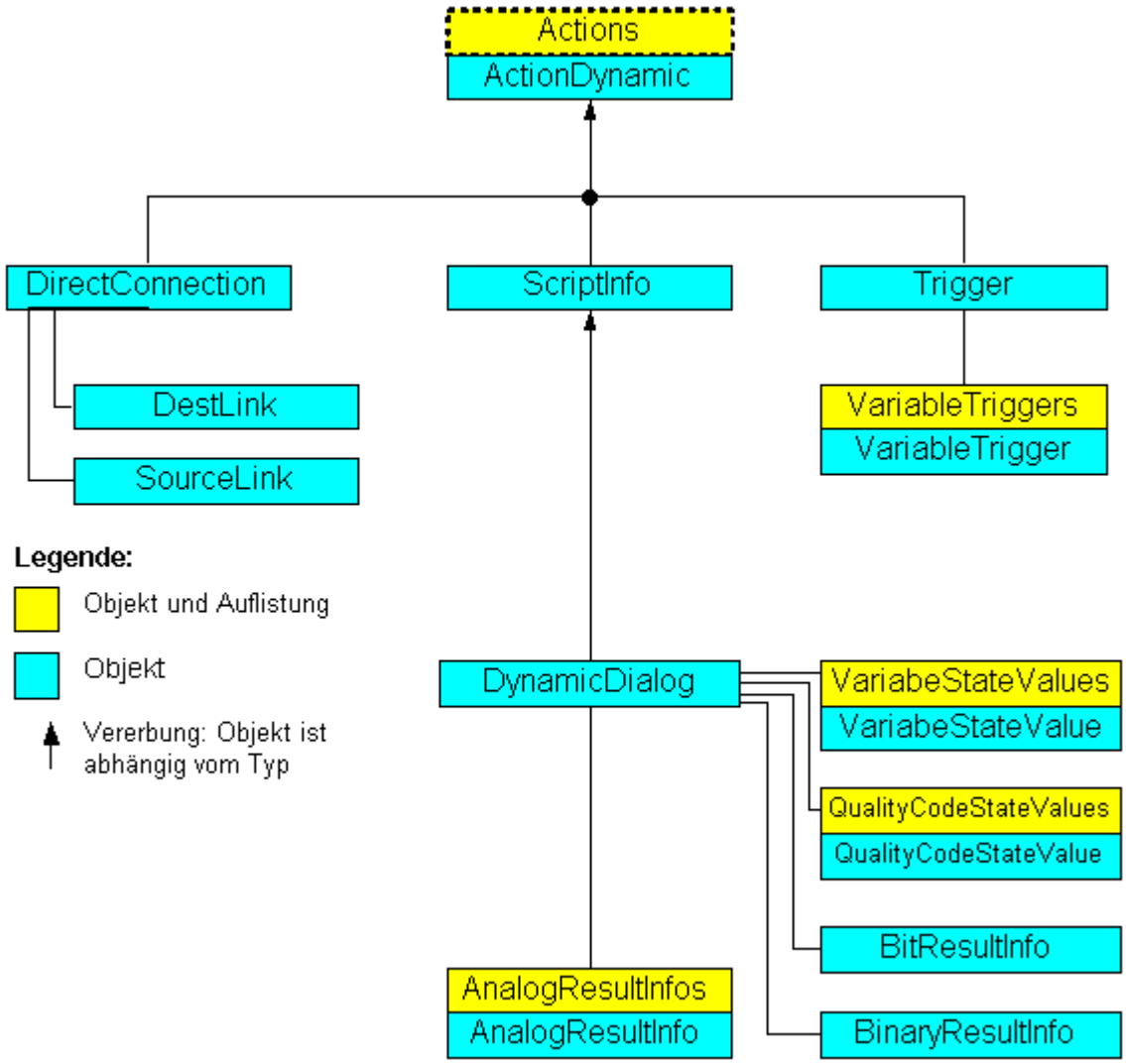
Events-Objekt (Auflistung) (Seite 1936)
SymbolLibraries-Objekt (Auflistung) (Seite 2036)
Actions-Objekt (Auflistung) (Seite 1884)
Application-Objekt (Seite 1888)
DataLanguage-Objekt (Seite 1914)
DataLanguages-Objekt (Auflistung) (Seite 1915)
Document-Objekt (Seite 1920)
Documents-Objekt (Auflistung) (Seite 1923)
Event-Objekt (Seite 1935)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
HMIObject-Objekt (Seite 1955)
HMIObjects-Objekt (Auflistung) (Seite 1957)
FolderItem-Objekt (Seite 1939)
FolderItems-Objekt (Auflistung) (Seite 1941)
VBA-Referenz: ActionDynamic (Seite 1737)
VBA-Referenz: HMIObjects (Seite 1739)
VBA-Referenz: Languages (Seite 1741)
Layer-Objekt (Seite 1967)
Layers-Objekt (Auflistung) (Seite 1969)
Menu-Objekt (Seite 1976)
Menus-Objekt (Auflistung) (Seite 1977)
MenuItem-Objekt (Seite 1979)
MenuItems-Objekt (Auflistung) (Seite 1982)
Properties-Objekt (Auflistung) (Seite 2004)
Toolbar-Objekt (Seite 2040)
Toolbars-Objekt (Auflistung) (Seite 2041)
ToolbarItem-Objekt (Seite 2043)
ToolbarItems-Objekt (Auflistung) (Seite 2046)
View-Objekt (Seite 2062)
Views-Objekt (Auflistung) (Seite 2064)
Selection-Objekt (Auflistung) (Seite 2022)
SymbolLibrary-Objekt (Seite 2035)
Property-Objekt (Seite 2005)

3.5.1.2 VBA-Referenz: ActionDynamic

VBA-Objektmodell: ActionDynamic

"ActionDynamic" stellt die Schnittstelle zu Dynamiken und Aktionen wie Skripten, dem Dynamik-Dialog, der Direktverbindung und den Triggern dar.

Wenn Sie mit der Maus auf einen Objektnamen klicken, erhalten Sie eine detaillierte Beschreibung.



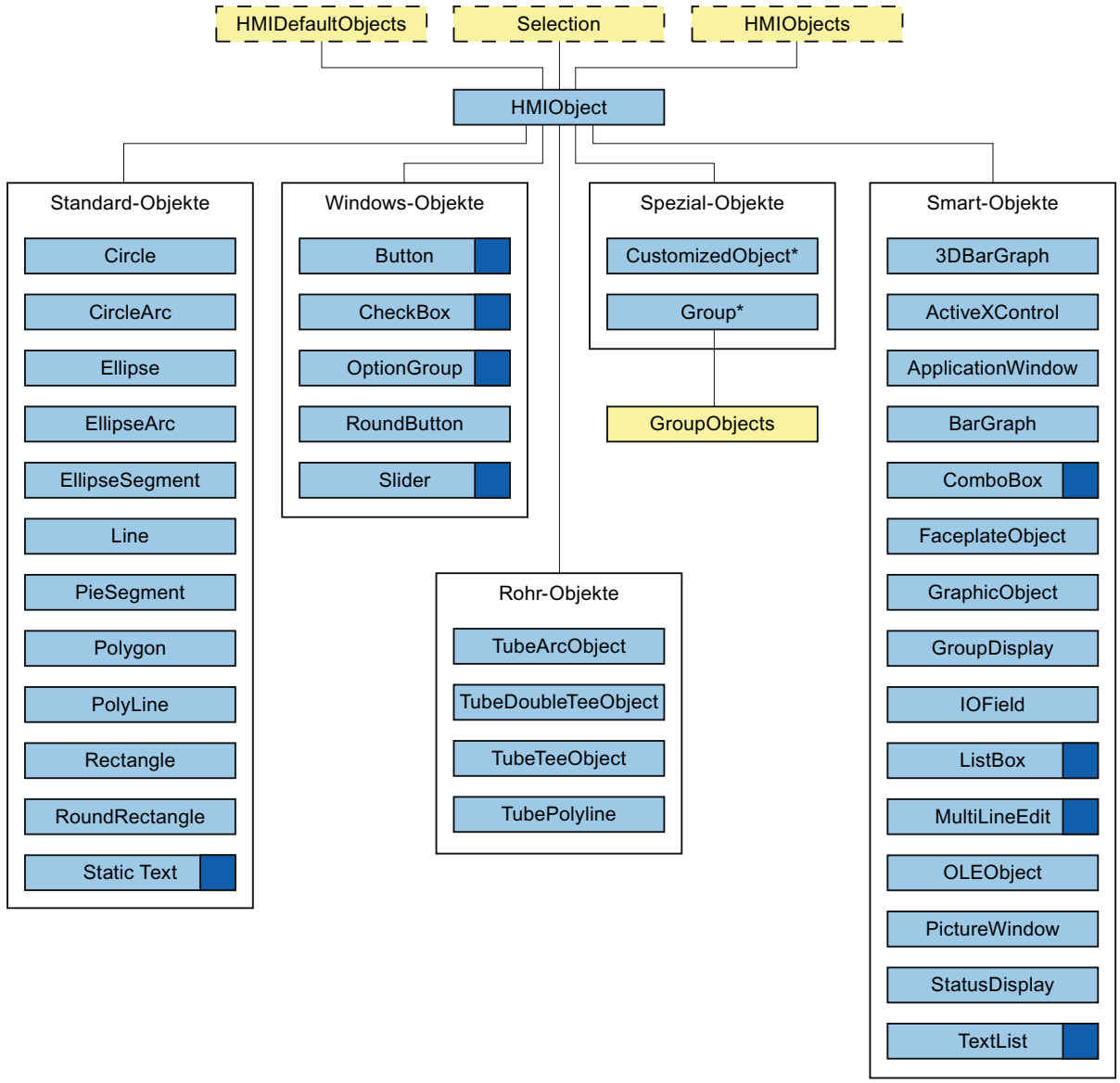
Siehe auch

VBA-Referenz (Seite 1735)
AnalogResultInfo-Objekt (Seite 1886)
AnalogResultInfos-Objekt (Auflistung) (Seite 1887)
BinaryResultInfo-Objekt (Seite 1896)
BitResultInfo-Objekt (Seite 1897)
Actions-Objekt (Auflistung) (Seite 1884)
DestLink-Objekt (Seite 1917)
DirectConnection-Objekt (Seite 1918)
DynamicDialog-Objekt (Seite 1924)
QualityCodeStateValue-Objekt (Seite 2007)
QualityCodeStateValues-Objekt (Auflistung) (Seite 2009)
ScriptInfo-Objekt (Seite 2021)
SourceLink-Objekt (Seite 2028)
Trigger-Objekt (Seite 2047)
VariableStateValue-Objekt (Seite 2057)
VariableStateValues-Objekt (Auflistung) (Seite 2058)
VariableTrigger-Objekt (Seite 2060)
VariableTriggers-Objekt (Auflistung) (Seite 2061)

3.5.1.3 VBA-Referenz: HMIObjects

VBA-Objektmodell: HMIObjects

Wenn Sie mit der Maus auf einen Objektnamen klicken, erhalten Sie eine detaillierte Beschreibung.



Objekt und Auflistung

Objekt

Detailansicht verfügbar.
Objekt kann mehrsprachig konfiguriert werden.

* Nicht in der Auflistung DefaultObjects enthalten.

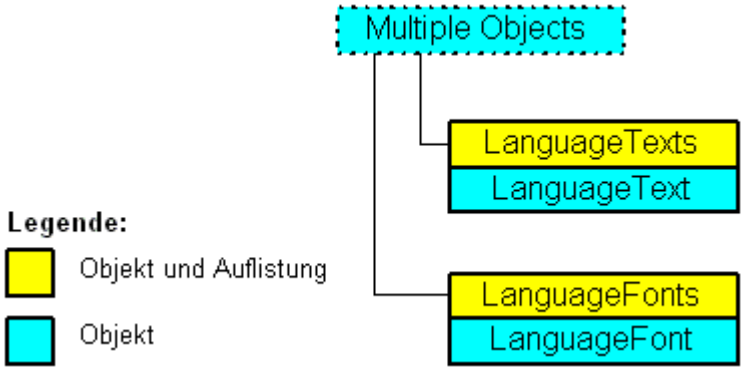
Siehe auch

VBA-Referenz (Seite 1735)
PolyLine-Objekt (Seite 2001)
GroupDisplay-Objekt (Seite 1947)
3DBarGraph-Objekt (Seite 1879)
ActiveXControl-Objekt (Seite 1885)
ApplicationWindow-Objekt (Seite 1891)
Button-Objekt (Seite 1898)
CheckBox-Objekt (Seite 1901)
Circle-Objekt (Seite 1902)
CircularArc-Objekt (Seite 1905)
Line-Objekt (Seite 1970)
OLEObject-Objekt (Seite 1987)
OptionGroup-Objekt (Seite 1989)
PictureWindow-Objekt (Seite 1992)
PieSegment-Objekt (Seite 1995)
Polygon-Objekt (Seite 1998)
Property-Objekt (Seite 2005)
Rectangle-Objekt (Seite 2012)
RoundButton-Objekt (Seite 2015)
RoundRectangle-Objekt (Seite 2018)
Slider-Objekt (Seite 2025)
StaticText-Objekt (Seite 2029)
StatusDisplay-Objekt (Seite 2032)
TextList-Objekt (Seite 2037)
Ellipse-Objekt (Seite 1926)
EllipseArc-Objekt (Seite 1929)
EllipseSegment-Objekt (Seite 1932)
GraphicObject-Objekt (Seite 1943)
Group-Objekt (Seite 1946)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
HMIObject-Objekt (Seite 1955)
HMIObjects-Objekt (Auflistung) (Seite 1957)
IOField-Objekt (Seite 1959)
BarGraph-Objekt (Seite 1893)
GroupedObjects-Objekt (Auflistung) (Seite 1950)
VBA-Referenz: Languages (Seite 1741)
Selection-Objekt (Auflistung) (Seite 2022)
CustomizedObject-Objekt (Seite 1912)
FaceplateObject-Objekt (Seite 1938)

3.5.1.4 VBA-Referenz: Languages

VBA-Objektmodell: Languages

Wenn Sie mit der Maus auf einen Objektnamen klicken, erhalten Sie eine detaillierte Beschreibung.



Siehe auch

VBA-Referenz (Seite 1735)

LanguageFont-Objekt (Seite 1962)

LanguageFonts-Objekt (Auflistung) (Seite 1963)

LanguageText-Objekt (Seite 1965)

LanguageTexts-Objekt (Auflistung) (Seite 1966)

3.5.1.5 Ereignisse

A-D

Activated-Ereignis

Beschreibung

Tritt ein, wenn ein Bild im Graphics Designer aktiviert wird. Dies geschieht z.B. beim Umschalten zwischen zwei Bildern.

Syntax

```
Document_Activated(CancelForwarding As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

Im folgenden Beispiel wird eine Meldung ausgegeben, wenn das Bild aktiviert wird:

```
Private Sub Document_Activated(CancelForwarding As Boolean)
'VBA76
MsgBox "The document got the focus." & vbCrLf &_
"This event (Document_Activated) is raised by the document itself"
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

Event-Handling (Seite 1715)

BeforeClose-Ereignis

Beschreibung

Tritt ein, unmittelbar bevor ein Bild geschlossen wird.

Syntax

```
Document_BeforeClose(Cancel As Boolean, CancelForwarding As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
Cancel (Boolean)	TRUE, wenn die Verarbeitung des Befehls abgebrochen werden soll.
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

Im folgenden Beispiel wird eine Meldung ausgegeben, bevor das Bild geschlossen wird:

```
Private Sub Document_BeforeClose(Cancel As Boolean, CancelForwarding As Boolean)
'VBA77
MsgBox "Event Document_BeforeClose is raised"
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

BeforeDocumentClose-Ereignis

Beschreibung

Tritt ein, unmittelbar bevor das Bild geschlossen wird.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_BeforeDocumentClose(Document As HMIDocument, Cancel
As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
Document (HMIDocument)	Das Bild, das geschlossen wird.
Cancel (Boolean)	TRUE, wenn die Verarbeitung des Befehls abgebrochen werden soll.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird eine Meldung ausgegeben, bevor das Bild geschlossen wird:

```
Private Sub objGDApplication_BeforeDocumentClose(ByVal Document As IHMIDocument, Cancel As
Boolean)
'VBA78
MsgBox "The document " & Document.Name & " will be closed after press ok"
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

BeforeDocumentSave-Ereignis

Beschreibung

Tritt ein, unmittelbar bevor das Bild gespeichert wird.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_BeforeDocumentSave(Document As HMIDocument, Cancel
As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
Document (HMIDocument)	Das Bild, das geschlossen wird.
Cancel (Boolean)	TRUE, wenn die Verarbeitung des Befehls abgebrochen werden soll.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird eine Meldung ausgegeben, bevor das Bild geschlossen wird:

```
Private Sub objGDApplication_BeforeDocumentSave(ByVal Document As IHMIDocument, Cancel As
Boolean)
'VBA79
MsgBox Document.Name & "-saving will start after press ok."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

BeforeHMIObjectDelete-Ereignis

Beschreibung

Tritt ein, unmittelbar bevor ein Objekt im Bild gelöscht wird.

Syntax

```
BeforeHMIObjectDelete(ByVal HMIObject As IHMIObject, Cancel As
Boolean, CancelForwarding As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
HMIObject (IHMIObject)	Identifiziert das zu löschende Objekt.
Cancel (Boolean)	TRUE, wenn die Verarbeitung des Befehls abgebrochen werden soll.
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

Im folgenden Beispiel wird eine Meldung ausgegeben, welches Objekt gelöscht werden wird:

```
Private Sub Document_BeforeHMIObjectDelete(ByVal HMIObject As IHMIObject, Cancel As
Boolean, CancelForwarding As Boolean)
'VBA80
Dim strObjName As String
Dim strAnswer As String
'
'"strObjName" contains the name of the deleted object
strObjName = HMIObject.ObjectName
strAnswer = MsgBox("Are you sure to delete " & strObjName & "?", vbYesNo)
If strAnswer = vbNo Then
'if pressed "No" -> set Cancel to true for prevent delete
Cancel = True
End If
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

BeforeLibraryFolderDelete-Ereignis

Beschreibung

Tritt ein, unmittelbar bevor ein Ordner in der Bausteinbibliothek gelöscht wird.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_BeforeLibraryFolderDelete(LibObject As
HMIFolderItem, Cancel As Boolean)
```

Parameter (optional)

Parameter (Datentyp)	Beschreibung
LibObject (HMIFolderItem)	Der Ordner, der gelöscht wird.
Cancel (Boolean)	TRUE, wenn die Verarbeitung des Befehls abgebrochen werden soll.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird eine Meldung ausgegeben, bevor ein Ordner in der Bausteinbibliothek gelöscht wird:

```
Private Sub objGDApplication_BeforeLibraryFolderDelete(ByVal LibObject As HMIFolderItem,
Cancel As Boolean)
'VBA81
MsgBox "The library-folder " & LibObject.Name & " will be delete..."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

BeforeLibraryObjectDelete-Ereignis

Beschreibung

Tritt ein, unmittelbar bevor ein Objekt in der Bausteinbibliothek gelöscht wird.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_BeforeLibraryObjectDelete(LibObject As
HMIFolderItem, Cancel As Boolean)
```

Parameter (optional)

Parameter (Datentyp)	Beschreibung
LibObject (HMIFolderItem)	Das Objekt, das gelöscht wird.
Cancel (Boolean)	TRUE, wenn die Verarbeitung des Befehls abgebrochen werden soll.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird eine Meldung ausgegeben, bevor ein Ordner in der Bausteinbibliothek gelöscht wird:

```
Private Sub objGDApplication_BeforeLibraryObjectDelete(ByVal LibObject As HMIFolderItem,
Cancel As Boolean)
'VBA82
MsgBox "The object " & LibObject.Name & " will be delete..."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

BeforeQuit-Ereignis

Beschreibung

Tritt ein, unmittelbar bevor der Graphics Designer geschlossen wird.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_BeforeQuit(Cancel As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
Cancel (Boolean)	TRUE, wenn die Verarbeitung des Befehls abgebrochen werden soll.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

In diesem Beispiel wird eine Meldung ausgegeben, kurz bevor der Graphics Designer geschlossen wird.

```
Private Sub objGDApplication_BeforeQuit(Cancel As Boolean)
'VBA83
MsgBox "The Graphics Designer will be shut down"
```

```
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

BeforeSave-Ereignis

Beschreibung

Tritt ein, unmittelbar bevor ein Bild gespeichert wird.

Syntax

```
Document_BeforeSave(Cancel As Boolean, CancelForwarding As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
Cancel (Boolean)	TRUE, wenn die Verarbeitung des Befehls abgebrochen werden soll.
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

Im folgenden Beispiel wird eine Meldung ausgegeben, bevor das Bild gespeichert wird:

```
Private Sub Document_BeforeSave(Cancel As Boolean, CancelForwarding As Boolean)
'VBA84
MsgBox "The document will be saved..."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

BeforeVisibleFalse-Ereignis

Beschreibung

Tritt ein, unmittelbar bevor die Applikation Graphics Designer von sichtbar auf unsichtbar gesetzt wird.

Syntax

```
Document_BeforeVisibleFalse(Cancel As Boolean, CancelForwarding As
Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
Cancel (Boolean)	TRUE, wenn die Verarbeitung des Befehls abgebrochen werden soll.
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

--

Siehe auch

VBA-Referenz (Seite 1735)

ConnectionEvent-Ereignis

Beschreibung

Tritt ein, wenn zwei Objekte mit dem Verbinder verbunden werden.

Syntax

```
ConnectionEvent(eConnEventType, HMIConnector, HMIConnectedObject,
CancelProcess, CancelForwarding)
```

Parameter (optional)

Parameter (Datentyp)	Beschreibung
eConnEventType (HMIConnectionEventType)	--
HMIConnector (HMIObject)	--
HMIConnectedObject (HMIObject)	--
CancelProcess (Boolean)	TRUE, wenn die Verarbeitung des Befehls abgebrochen werden soll.
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

--

Siehe auch

VBA-Referenz (Seite 1735)

DataLanguageChanged-Ereignis

Beschreibung

Tritt ein, wenn wenn die Projektiersprache umgeschaltet wurde.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_DataLanguageChanged(lCID As Long)
```

Parameter

Parameter (Datentyp)	Beschreibung
lCID (Long)	Die Sprachkennung der Projektiersprache

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird die neu eingestellte Projektiersprache ausgegeben:

```
Private Sub objGDApplication_DataLanguageChanged(ByVal lCID As Long)
```

```
'VBA87
MsgBox "The datalanguage is changed to " & Application.CurrentDataLanguage & "."
End Sub
```

Siehe auch

Sprachabhängige Projektierung mit VBA (Seite 1626)

VBA-Referenz (Seite 1735)

DesktopLanguageChanged-Ereignis

Beschreibung

Tritt ein, wenn die Oberflächen-Sprache umgeschaltet wurde.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_DesktopLanguageChanged(lCID As Long)
```

Parameter

Parameter (Datentyp)	Beschreibung
lCID (Long)	Die Sprachkennung der Oberflächen-Sprache

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird die neu eingestellte Oberflächen-Sprache ausgegeben:

```
Private Sub objGDApplication_DesktopLanguageChanged(ByVal lCID As Long)
'VBA88
MsgBox "The desktop-language is changed to " & Application.CurrentDesktopLanguage & "."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

Sprachabhängige Projektierung mit VBA (Seite 1626)

DocumentActivated-Ereignis

Beschreibung

Tritt ein, wenn ein Bild im Graphics Designer aktiviert wird. Dies geschieht z.B. beim Umschalten zwischen zwei Bildern.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_DocumentActivated(Document As HMIDocument)
```

Parameter

Parameter (Datentyp)	Beschreibung
Document (HMIDocument)	Das Bild, das aktiviert wird.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird eine Meldung ausgegeben, welches Bild aktiviert wurde:

```
Private Sub objGDApplication_DocumentActivated(ByVal Document As IHMIDocument)
'VBA89
MsgBox "The document " & Document.Name & " got the focus." & vbCrLf &_
"This event is raised by the application."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

DocumentCreated-Ereignis

Beschreibung

Tritt ein, wenn ein neues Bild im Graphics Designer erstellt wurde.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_DocumentCreated(Document As HMIDocument)
```

Parameter

Parameter (Datentyp)	Beschreibung
Document (HMIDocument)	Das Bild, das erstellt wurde.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird der Bildname vom neu erstellten Bild ausgegeben:

```
Private Sub objGDApplication_DocumentCreated(ByVal Document As IHMIDocument)
'VBA90
MsgBox Document.Name & " will be created."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

DocumentOpened-Ereignis

Beschreibung

Tritt ein, wenn ein Bild geöffnet wurde.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_DocumentOpened(Document As HMIDocument)
```

Parameter

Parameter (Datentyp)	Beschreibung
Document (HMIDocument)	Das Bild, das geöffnet wurde.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird eine Meldung ausgegeben, welches Bild geöffnet wurde:

```
Private Sub objGDApplication_DocumentOpened(ByVal Document As IHMIDocument)
'VBA91
MsgBox Document.Name & " is opened."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

DocumentSaved-Ereignis

Beschreibung

Tritt ein, wenn ein Bild im Graphics Designer gespeichert wurde.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_DocumentSaved(Document As HMIDocument)
```

Parameter

Parameter (Datentyp)	Beschreibung
Document (HMIDocument)	Das Bild, das gespeichert wurde.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird eine Meldung ausgegeben, welches Bild gespeichert wurde:

```
Private Sub objGDApplication_DocumentSaved(ByVal Document As IHMIDocument)
'VBA92
MsgBox Document.Name & " is saved."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

DocumentPropertyChanged-Ereignis

Beschreibung

Tritt ein, wenn eine Bildeigenschaft verändert wird.

Syntax

```
Document_DocumentPropertyChanged(ByVal Property As IHMIProperty,
CancelForwarding As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
Property (IHMIProperty)	Identifiziert die veränderte Eigenschaft.
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

Im folgenden Beispiel wird eine Meldung ausgegeben, welche Bildeigenschaft verändert wird:

```
Private Sub Document_DocumentPropertyChanged(ByVal Property As IHMIProperty,
CancelForwarding As Boolean)
'VBA93
Dim strPropName As String
'"strPropName" contains the name of the modified property
strPropName = Property.Name
MsgBox "The picture-property " & strPropName & " is modified..."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

F-Z

HMIObjectAdded-Ereignis

Beschreibung

Tritt ein, wenn ein Objekt hinzugefügt wird.

Syntax

```
Document_HMIObjectAdded(ByVal HMIObject As IHMIObject,
CancelForwarding As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
HMIObject (IHMIObject)	Identifiziert die hinzugefügte Objekt.
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

Im folgenden Beispiel wird eine Meldung ausgegeben, welches Objekt hinzugefügt wurde:

```
Private Sub Document_HMIObjectAdded(ByVal HMIObject As IHMIObject, CancelForwarding As
Boolean)
'VBA94
Dim strObjName As String
'
'"strObjName" contains the name of the added object
strObjName = HMIObject.ObjectName
MsgBox "Object " & strObjName & " is added..."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

HMIObjectMoved-Ereignis

Beschreibung

Tritt ein, wenn ein Objekt verschoben wird.

Syntax

```
Document_HMIObjectMoved(ByVal HMIObject As IHMIObject,
CancelForwarding As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
HMIObject (IHMIObject)	Identifiziert die verschobene Objekt.
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

Im folgenden Beispiel wird eine Meldung ausgegeben, welches Objekt verschoben wurde:

```
Private Sub Document_HMIObjectMoved(ByVal HMIObject As IHMIObject, CancelForwarding As
Boolean)
'VBA95
Dim strObjName As String
'
'"strObjName" contains the name of the moved object
strObjName = HMIObject.ObjectName
MsgBox "Object " & strObjName & " was moved..."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

HMIObjectPropertyChanged-Ereignis

Beschreibung

Tritt ein, wenn eine Objekteigenschaft verändert wird.

Syntax

```
Document_HMIObjectPropertyChanged(ByVal Property As IHMIProperty,
CancelForwarding As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
Property (IHMIProperty)	Identifiziert die veränderte Eigenschaft.
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

Im folgenden Beispiel wird eine Meldung ausgegeben, welche Objekteigenschaft verändert wurde:

```
Private Sub Document_HMIObjectPropertyChanged(ByVal Property As IHMIProperty,
CancelForwarding As Boolean)
'VBA96
Dim strObjProp As String
Dim strObjName As String
Dim varPropValue As Variant
'
'"strObjProp" contains the name of the modified property
'"varPropValue" contains the new value
strObjProp = Property.Name
varPropValue = Property.value
'
'"strObjName" contains the name of the selected object,
'which property is modified
strObjName = Property.Application.ActiveDocument.Selection(1).ObjectName
MsgBox "The property " & strObjProp & " of object " & strObjName & " is modified... " &
vbCrLf & "The new value is: " & varPropValue
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

HMIObjectResized-Ereignis

Beschreibung

Tritt ein, wenn die Größe eines Objektes verändert wird.

Syntax

```
Document_HMIObjectResized(ByVal HMIObject As IHMIObject,
CancelForwarding As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
HMIObject (IHMIObject)	Identifiziert das in der Größe veränderte Objekt.
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

Im folgenden Beispiel wird eine Meldung ausgegeben, wenn die Größe eines Objektes verändert wurde:

```
Private Sub Document_HMIObjectResized(ByVal HMIObject As IHMIObject, CancelForwarding As
Boolean)
'VBA97
Dim strObjName As String
'
'"strObjName" contains the name of the modified object
strObjName = HMIObject.ObjectName
MsgBox "The size of " & strObjName & " was modified..."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

LibraryFolderRenamed-Ereignis

Beschreibung

Tritt ein, wenn ein Ordner in der Bausteinbibliothek umbenannt wurde.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_LibraryFolderRenamed(LibObject As HMIFolderItem,
OldName As String)
```

Parameter

Parameter (Datentyp)	Beschreibung
LibObject (HMIFolderItem)	Der umbenannte Ordner.
OldName (String)	Der ursprüngliche Name des umbenannten Ordners.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird der alte und neue Ordnername ausgegeben:

```
Private Sub objGDApplication_LibraryFolderRenamed(ByVal LibObject As HMIFolderItem, ByVal
OldName As String)
'VBA98
MsgBox "The Library-folder " & OldName & " is renamed in: " & LibObject.DisplayName
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

LibraryObjectRenamed-Ereignis

Beschreibung

Tritt ein, wenn ein Objekt in der Bausteinbibliothek umbenannt wurde.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_LibraryObjectRenamed(LibObject As HMIFolderItem,
OldName As String)
```

Parameter

Parameter (Datentyp)	Beschreibung
LibObject (HMIFolderItem)	Das umbenannte Objekt.
OldName (String)	Der ursprüngliche Name des umbenannten Objektes.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird der alte und neue Objektname ausgegeben:

```
Private Sub objGDApplication_LibraryObjectRenamed(ByVal LibObject As IHMIFolderItem, ByVal
OldName As String)
'VBA99
MsgBox "The object " & OldName & " is renamed in: " & LibObject.DisplayName
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

LibraryObjectAdded-Ereignis

Beschreibung

Tritt ein, wenn ein Objekt zur Bausteinbibliothek hinzugefügt wurde.

Syntax

```
HMIObjectPropertyChanged(ByVal Property As IHMIProperty,
CancelForwarding As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
LibObject (IHMIFolderItem)	Identifiziert das Library-Objekt.
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

Im folgenden Beispiel wird eine Meldung ausgegeben, wenn ein Objekt in die Bausteinbibliothek eingefügt wurde:

```
Private Sub Document_LibraryObjectAdded(ByVal LibObject As IHMIFolderItem,
CancelForwarding As Boolean)
'VBA100
Dim strObjName As String
'
'"strObjName" contains the name of the added object
strObjName = LibObject.DisplayName
MsgBox "Object " & strObjName & " was added to the picture."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

MenuItemClicked-Ereignis

Beschreibung

Tritt ein, wenn ein Eintrag eines benutzerdefinierten Menüs angeklickt wird.

Hinweis

Dieses Ereignis ist applikations- und dokumentspezifisch.

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Syntax

```
Document_MenuItemClicked(ByVal MenuItem As IHMIMenuItem)
```

Parameter

Parameter (Datentyp)	Beschreibung
MenuItem (IHMIMenuItem)	Identifiziert das benutzerdefinierte Menü.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird eine Meldung ausgegeben, wenn der erste Eintrag eines benutzerdefinierten Menüs angeklickt wird:

```
Private Sub Document_MenuItemClicked(ByVal MenuItem As IHMIMenuItem)
'VBA101
Dim objMenuItem As HMIMenuItem
Dim varMenuItemKey As Variant
Set objMenuItem = MenuItem
'
'"objMenuItem" contains the clicked menu-item
'"varMenuItemKey" contains the value of parameter "Key"
'from the clicked userdefined menu-item
```

```
varMenuItemKey = objMenuItem.Key
Select Case MenuItem.Key
Case "mItem1_1"
MsgBox "The first menu-item was clicked!"
End Select
End Sub
```

Siehe auch

So weisen Sie Menüs und Symbolleisten VBA-Makros zu (Seite 1644)

VBA-Referenz (Seite 1735)

NewLibraryFolder-Ereignis

Beschreibung

Tritt ein, wenn ein Ordner in der Bausteinbibliothek angelegt wurde.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_NewLibraryFolder(LibObject As HMIFolderItem)
```

Parameter

Parameter (Datentyp)	Beschreibung
LibObject (HMIFolderItem)	Der neu erstellte Ordner.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird der neue Ordnername ausgegeben:

```
Private Sub objGDApplication_NewLibraryFolder(ByVal LibObject As IHMIFolderItem)
'VBA102
MsgBox "The library-folder " & LibObject.DisplayName & " was added."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

NewLibraryObject-Ereignis

Beschreibung

Tritt ein, wenn ein Objekt in der Bausteinbibliothek angelegt wurde.

Syntax

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

```
objGDApplication_NewLibraryObject(LibObject As HMIFolderItem)
```

Parameter

Parameter (Datentyp)	Beschreibung
LibObject (HMIFolderItem)	Das neu erstellte Objekt.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
```

```
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird der neue Objektname ausgegeben:

```
Private Sub objGDApplication_NewLibraryObject(ByVal LibObject As IHMIFolderItem)
'VBA103
MsgBox "The object " & LibObject.DisplayName & " was added."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

Opened-Ereignis

Beschreibung

Tritt ein, wenn ein Bild geöffnet wird.

Syntax

```
Document_Opened(CancelForwarding As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

Im folgenden Beispiel wird eine Meldung ausgegeben, wenn das Bild geöffnet wird:

```
Private Sub Document_Opened(CancelForwarding As Boolean)
'VBA104
MsgBox "The Document is open now..."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

Saved-Ereignis

Beschreibung

Tritt ein, nachdem ein Bild gespeichert wurde.

Syntax

```
Document_Saved(CancelForwarding As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

Im folgenden Beispiel wird eine Meldung ausgegeben, wenn das Bild gespeichert wurde:

```
Private Sub Document_Saved(CancelForwarding As Boolean)
'VBA105
MsgBox "The document is saved..."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

SelectionChanged-Ereignis

Beschreibung

Tritt ein, wenn die Auswahl geändert wurde.

Syntax

```
Document_SelectionChanged(CancelForwarding As Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

Im folgenden Beispiel wird eine Meldung ausgegeben, wenn ein neues Objekt angewählt wird:

```
Private Sub Document_SelectionChanged(CancelForwarding As Boolean)
'VBA106
MsgBox "The selection is changed..."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

Started-Ereignis

Beschreibung

Tritt ein, wenn der Graphics Designer gestartet wurde.

Syntax

```
objGDApplication_Started()
```

Hinweis

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Im folgenden Beispiel wird für <Name> die Bezeichnung "objGDApplication" verwendet.

Parameter

--

Beispiel

Anwendung deklarieren.

```
Dim WithEvents objGDApplication As grafexe.Application
```

Ereignisvariable setzen.

```
Private Sub Document_Opened(CancelForwarding As Boolean)
 Set objGDApplication = Me.Application
End Sub
```

Ereignis "Started" abfragen und Meldung ausgeben.

```
Private Sub objGDApplication_Started()
'VBA107
'This event is raised before objGDApplication_Started()
 MsgBox "The Graphics Designer is started!"
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

ToolbarItemClicked-Ereignis

Beschreibung

Tritt ein, wenn ein Symbol in einer benutzerdefinierten Symbolleiste angeklickt wurde

Hinweis

Dieses Ereignis ist applikations- und dokumentspezifisch.

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Syntax

```
Document_ToolbarItemClicked(ByVal ToolbarItem As IHMIToolbarItem)
```

Parameter

Parameter (Datentyp)	Beschreibung
ToolbarItem (IHMIToolbarItem)	Identifiziert das Symbol.

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird eine Meldung ausgegeben, wenn das erste benutzerdefinierte Symbol angeklickt wird:

```
Private Sub Document_ToolbarItemClicked(ByVal ToolbarItem As IHMIToolbarItem)
'VBA108
Dim objToolbarItem As HMIToolbarItem
Dim varToolbarItemKey As Variant
Set objToolbarItem = ToolbarItem
'
'"varToolbarItemKey" contains the value of parameter "Key"
'from the clicked userdefined toolbar-item
varToolbarItemKey = objToolbarItem.Key
'
Select Case varToolbarItemKey
Case "tItem1_1"
MsgBox "The first Toolbar-Icon was clicked!"
End Select
End Sub
```

Siehe auch

So weisen Sie Menüs und Symbolleisten VBA-Makros zu (Seite 1644)

VBA-Referenz (Seite 1735)

ViewCreated-Ereignis

Beschreibung

Tritt ein, wenn eine Kopie eines Bildes erstellt wurde.

Hinweis

Dieses Ereignis ist applikations- und dokumentspezifisch.

Damit das applikationsspezifische Ereignis im Projekt verfügbar ist, muss die Anwendung Graphics Designer bekannt gemacht werden. Das geschieht mit folgender Anweisung:

Dim WithEvents <Name> As grafexe.Application

Syntax

```
Document_ViewCreated(ByVal pView As IHMIView, CancelForwarding As
Boolean)
```

Parameter

Parameter (Datentyp)	Beschreibung
pView (IHMIView)	Identifiziert die Kopie des Bildes.
CancelForwarding (Boolean)	TRUE, wenn das Ereignis nicht weitergeleitet werden soll. Standardeinstellung ist "False".

Beispiel

Führen Sie folgende Prozedur aus, damit das unten gezeigte Beispiel funktioniert:

```
Private Sub SetApplication()
'This procedure have to execute with "F5" first
Set objGDApplication = grafexe.Application
End Sub
```

Im folgenden Beispiel wird die Anzahl der Kopien von Bildern ausgegeben, wenn eine neue Kopie des Bildes erstellt wurde.

```
Private Sub Document_ViewCreated(ByVal pView As IHMIView, CancelForwarding As Boolean)
'VBA109
Dim iViewCount As Integer
'
'To read out the number of views
iViewCount = pView.Application.ActiveDocument.Views.Count
```

```
MsgBox "A new copy of the picture (number " & iViewCount & ") was created."
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

WindowStateChange-Ereignis

Beschreibung

Tritt ein, wenn die Fenstergröße geändert wird (z.B. von "minimiert" zu "maximiert").

Syntax

```
objGDApplication_WindowStateChanged()
```

Parameter (optional)

--

Beispiel

Im folgenden Beispiel wird eine Meldung ausgegeben, wenn die Fenstergröße verändert wird.

```
Private Sub objGDApplication_WindowStateChanged()
'VBA110
MsgBox "The state of the application-window is changed!"
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

3.5.1.6 Methoden

A-C

Activate Methode

Beschreibung

Aktiviert das angegebene Objekt.

Syntax

Ausdruck.`Activate()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Application" oder "View" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel wird vom aktiven Bild eine Kopie erzeugt und diese dann aktiviert:

```
Sub CreateAndActivateView()
'VBA111
Dim objView As HMIView
Set objView = ActiveDocument.Views.Add
objView.Activate
End Sub
```

Siehe auch

View-Objekt (Seite 2062)

Application-Objekt (Seite 1888)

VBA-Referenz (Seite 1735)

Add-Methode

Beschreibung

Fügt einer Auflistung ein weiteres Element hinzu.

Die folgende Tabelle zeigt Ihnen die Auflistungen, auf welche die Add-Methode angewendet werden kann. Parameter und Syntax der jeweiligen Add-Methode finden Sie unter "Methoden".

Auflistung	Anwendung der Add-Methode
AnalogResultInfos-Auflistung	Fügt im Dynamik-Dialog einen neuen, analogen Wertebereich hinzu.
Documents-Auflistung	Erstellt in neues Bild im Graphics Designer.
GroupedObjects-Auflistung	Fügt einem Gruppen-Objekt ein neues Objekt hinzu.
Toolbars-Auflistung	Erstellt eine neue benutzerdefinierte Symbolleiste.
VariableTriggers-Auflistung	Erstellt einen neuen Variable-Trigger.
Views-Auflistung	Erstellt eine Kopie des angegebenen Bildes.

Siehe auch

Add-Methode (Views-Auflistung) (Seite 1784)

Add-Methode (VariableTriggers-Auflistung) (Seite 1783)

Add-Methode (CustomToolbars-Auflistung) (Seite 1779)

Add-Methode (GroupedObjects-Auflistung) (Seite 1781)

Add-Methode (Documents-Auflistung) (Seite 1780)

Add-Methode (AnalogResultInfos-Auflistung) (Seite 1777)

Add-Methode (AnalogResultInfos-Auflistung)

Beschreibung

Fügt im Dynamik-Dialog einen neuen, analogen Wertebereich hinzu.

Syntax

Ausdruck.`Add(RangeTo, ResultValue)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "AnalogResultInfos" zurückgibt.

Parameter

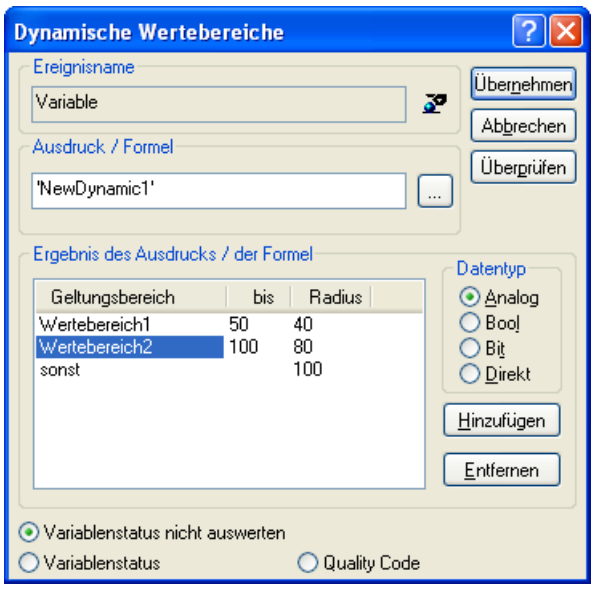
Parameter (Datentyp)	Beschreibung
RangeTo (Variant)	Der Wertebereich, der die Änderung der Eigenschaft hervorruft.
ResultValue (Variant)	Der Wert, welcher der Objekteigenschaft zugewiesen wird, wenn der Wertebereich erreicht ist.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Im folgenden Beispiel wird ein Variablenname vergeben und drei analoge Wertebereiche angelegt:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA112
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.Add 50, 40
.AnalogResultInfos.Add 100, 80
.AnalogResultInfos.ElseCase = 100
End With
End Sub
```

Die Abbildung zeigt den Dynamik-Dialog nach der Ausführung der Prozedur:



Siehe auch

DynamicDialog-Objekt (Seite 1924)

AnalogResultInfos-Objekt (Auflistung) (Seite 1887)

CreateDynamic-Methode (Seite 1815)

So dynamisieren Sie eine Eigenschaft mit dem Dynamik-Dialog (Seite 1697)

Add-Methode (CustomToolbars-Auflistung)

Beschreibung

Erstellt eine neue benutzerdefinierte Symbolleiste. Es wird zwischen anwendungs- und bildspezifischen benutzerdefinierten Symbolleisten unterschieden:

- Anwendungsspezifische Symbolleiste: Ist an den Graphics Designer gebunden und auch dann noch sichtbar, wenn alle Bilder im Graphics Designer geschlossen sind. Platzieren Sie den VBA-Code entweder im Dokument "GlobalTemplateDocument" oder "ProjectTemplateDocument" und verwenden Sie die Eigenschaft "Application".
- Bildspezifische Symbolleiste: Ist an ein bestimmtes Bild gebunden und bleibt solange sichtbar, wie das Bild aktiv ist. Platzieren Sie den VBA-Code im Dokument "ThisDocument" des gewünschten Bildes und verwenden Sie die Eigenschaft "ActiveDocument".

Syntax

Ausdruck.`Add(Key)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "CustomToolbars" zurückgibt.

Parameter

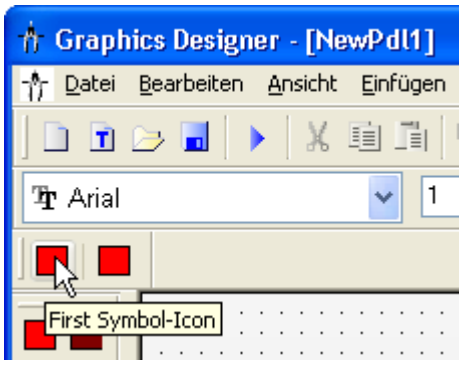
Parameter (Datentyp)	Beschreibung
Key (Variant)	Identifiziert die benutzerdefinierte Symbolleiste. Verwenden Sie für "Key" eindeutige Bezeichnungen (z.B. "DocToolbar1")

Beispiel

Im folgenden Beispiel wird im aktiven Bild eine benutzerdefinierte Symbolleiste mit zwei Symbolen angelegt, die durch eine Trennlinie getrennt sind:

```
Sub AddDocumentSpecificCustomToolbar()
'VBA115
Dim objToolbar As HMIToolbar
Dim objToolbarItem As HMIToolbarItem

Set objToolbar = ActiveDocument.CustomToolbars.Add("DocToolbar")

'Add toolbar-items to the userdefined toolbar
Set objToolbarItem = objToolbar.ToolbarItems.InsertToolbarItem(1, "tItem1_1", "My first
Symbol-Icon")
Set objToolbarItem = objToolbar.ToolbarItems.InsertToolbarItem(3, "tItem1_3", "My second
Symbol-Icon")
'
'Insert seperatorline between the two tollbaritems
```

```
Set objToolbarItem = objToolbar.ToolbarItems.InsertSeparator(2, "tSeparator1_2")
End Sub
```



Siehe auch

Toolbars-Objekt (Auflistung) (Seite 2041)

InsertToolbarItem-Methode (Seite 1842)

InsertSeparator-Methode (Seite 1839)

InsertFromMenuItem-Methode (Seite 1834)

VBA-Referenz (Seite 1735)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

Add-Methode (Documents-Auflistung)

Beschreibung

Erstellt in neues Bild im Graphics Designer.

Syntax

Ausdruck.`Add [HMIOpenDocumentType]`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Documents" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
HMIOpenDocumentType (HMIDocumentType)	Legt fest, wie das Bild geöffnet wird: • HMIDocumentTypeVisible: Öffnet das Bild zur direkten Bearbeitung. Dies ist die Standardeinstellung, wenn Sie den Parameter nicht angeben. • HMIDocumentTypeInvisible: Öffnet das Bild unsichtbar, d.h. es wird nicht im Graphics Designer angezeigt. Sie können das Bild nur über die Documents-Auflistung ansprechen und mit der Hide-Eigeschaft wieder auf sichtbar setzen.

Beispiel

Im folgenden Beispiel wird im Graphics Designer ein neues Bild erzeugt:

```
Sub AddNewDocument()
'VBA113
Application.Documents.Add hmiOpenDocumentTypeVisible
End Sub
```

Siehe auch

Hide-Eigenschaft (Seite 2214)

Documents-Objekt (Auflistung) (Seite 1923)

VBA-Referenz (Seite 1735)

Add-Methode (GroupedObjects-Auflistung)

Beschreibung

Fügt dem angegebenen Gruppen-Objekt ein vorhandenes Objekt hinzu.

Syntax

Ausdruck.`Add(Index)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "GroupedObjects" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
Index (Variant)	Das Objekt, das hinzugefügt werden soll. Sie können entweder die Indexnummer oder den Objektnamen verwenden.

Beispiel

In diesem Beispiel wird aus Objekten das Gruppen-Objekt "My Group" erzeugt. Danach wird dem Gruppen-Objekt ein Ellipsensegment hinzugefügt:

```
Sub CreateGroup()
'VBA114
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objEllipseSegment As HMIEllipseSegment
Dim objGroup As HMIGroup

Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With

MsgBox "Objects selected!"
Set objGroup = ActiveDocument.Selection.CreateGroup

'Set name for new group-object
'The name identifies the group-object
objGroup.ObjectName = "My Group"

'Add new object to active document...
Set objEllipseSegment = ActiveDocument.HMIObjects.AddHMIObject("EllipseSegment",
"HMIEllipseSegment")
Set objGroup = ActiveDocument.HMIObjects("My Group")

'...and add it to the group:
objGroup.GroupedHMIObjects.Add ("EllipseSegment")
End Sub
```

Siehe auch

GroupedObjects-Objekt (Auflistung) (Seite 1950)

Add-Methode (VariableTriggers-Auflistung)

Beschreibung

Erstellt einen neuen Variable-Trigger.

Syntax

Ausdruck.`Add(VarName, Type)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "VariableTriggers" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
VarName (String)	Der Name der Variablen, die als Trigger verwendet werden soll. Beachten Sie, dass Sie die Variable im Variablenauswahldialog erstellen müssen.
Type (CycleType)	Die Zyklusart. Die Zyklusart wählen Sie aus einer Liste im VBA-Editor aus, wenn Sie diese Methode anwenden.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit einem Variablentrigger dynamisiert:

```
Sub DynamicWithVariableTriggerCycle()
'VBA69
Dim objVBScript As HMIScriptInfo
Dim objVarTrigger As HMIVariableTrigger
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_VariableTrigger",
"HMICircle")
Set objVBScript = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeVBScript)
With objVBScript
Set objVarTrigger = .Trigger.VariableTriggers.Add("VarTrigger", hmiVariableCycleType_10s)
.SourceCode = ""
End With
End Sub
```

Siehe auch

VariableTriggers-Objekt (Auflistung) (Seite 2061)

VBA-Referenz (Seite 1735)

Add-Methode (Views-Auflistung)

Beschreibung

Erstellt eine Kopie des angegebenen Bildes.

Syntax

Ausdruck.`Add()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Views" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel vom aktiven Bild eine Kopie erzeugt und diese aktiviert:

```
Sub CreateViewAndActivateView()
'VBA117
Dim objView As HMIView
Set objView = ActiveDocument.Views.Add
objView.Activate
End Sub
```

Siehe auch

Views-Objekt (Auflistung) (Seite 2064)

VBA-Referenz (Seite 1735)

AddAction-Methode

Beschreibung

Projektiert eine Aktion an ein Objekt oder eine Eigenschaft, die beim Eintritt des definierten Ereignisses ausgelöst wird.

Syntax

Ausdruck.`Methode(HMIActionCreationType)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Actions" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
HMIActionCreationType (Variant)	Legt die Aktion fest: • hmiActionCreationTypeCScript: Projektiert eine C-Aktion • hmiActionCreationTypeVBScript: Projektiert eine VBS-Aktion • hmiActionCreationTypeDirectConnection: Projektiert eine Direktverbindung

Beispiel

Im folgenden Beispiel wird eine VBS-Aktion an die Radiusänderung eines Kreises projektiert:

```
Sub AddActionToPropertyTypeVBScript()
'VBA118
Dim objEvent As HMIEvent
Dim objVBScript As HMIScriptInfo
Dim objCircle As HMICircle
'Create circle in picture. By changing of property "Radius"
'a VBS-action will be started:
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_AB", "HMICircle")
Set objEvent = objCircle.Radius.Events(1)
Set objVBScript = objEvent.Actions.AddAction(hmiActionCreationTypeVBScript)
End Sub
```

Siehe auch

Event-Objekt (Seite 1935)

Actions-Objekt (Auflistung) (Seite 1884)

AddActiveXControl-Methode

Beschreibung

Fügt der Auflistung "HMIObjects" ein neues ActiveXControl-Objekt hinzu. Das Objekt wird in der linken oberen Ecke des angegebenen Bildes eingefügt.

Syntax

Ausdruck.`AddActiveXControl("ObjectName", "ProgID")`

Ausdruck

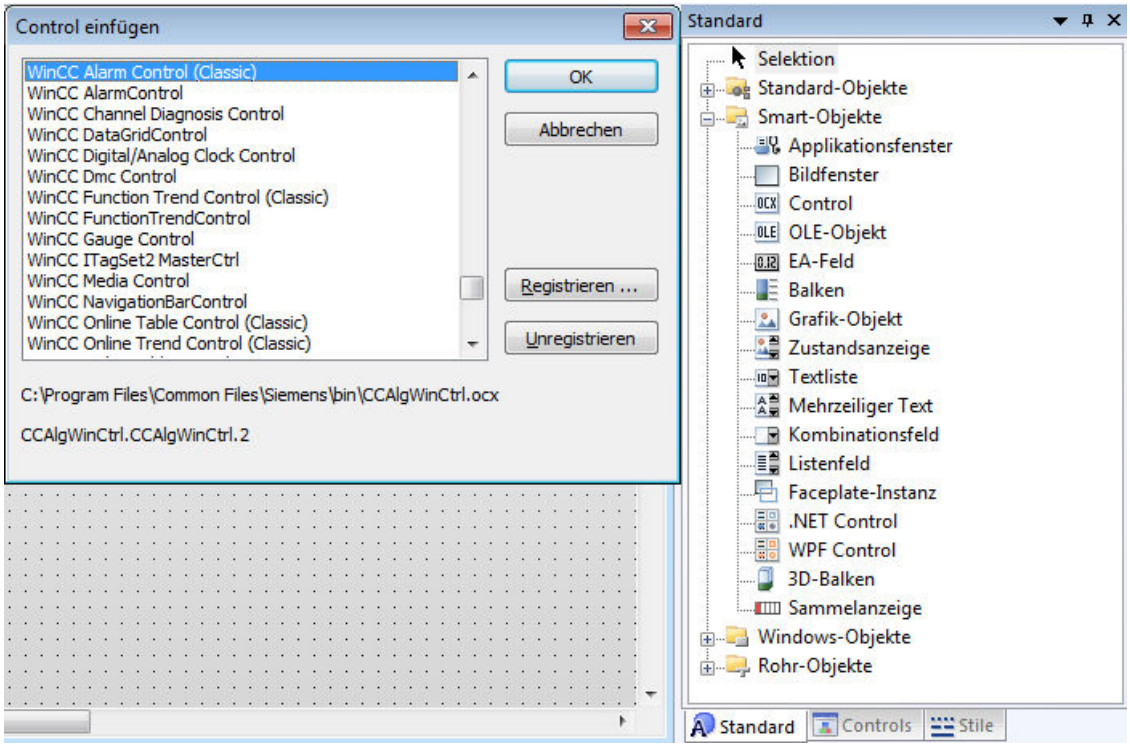
Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIObjects" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ObjectName (String)	Der Name des Objekts. Sie können das Objekt über seinen Namen in einer Auflistung ansprechen.
ProgID (String)	Das ActiveX-Control, das eingefügt werden soll.

ProgID ermitteln

Sie ermitteln die ProgID für ein ActiveX-Control, indem Sie in der "Objektpalette" des Graphics Designer auf der Registerkarte Standard unter "Smart-Objekte" das Control in das Bild einfügen. Im Dialog "Control einfügen" werden zu dem gewählten Control der Pfad und die ProgID angezeigt:



In der folgenden Tabelle sehen Sie eine Auflistung der ProgIDs von WinCC-Controls, die von WinCC installiert werden:

Name des WinCC Controls	ProgID
Siemens HMI Symbol Library	SiemensHMI.SymbolLibrary.1
WinCC AlarmControl	CCAxAlarmControl.AxAlarmControl.1
WinCC Digital/Analog Clock Control	DACLOCK.DaclockCtrl.1

Name des WinCC Controls	ProgID
WinCC FunctionTrendControl	CCAxFunctionTrendControl.AxFunctionTrendControl.1
WinCC Gauge Control	XGAUGE.XGaugeCtrl.1
WinCC Media Control	CCMediaControl.CCMediaControl.1
WinCC OnlineTableControl	CCAxOnlineTableControl.AxOnlineTableControl.1
WinCC OnlineTrendControl	CCAxOnlineTrendControl.AxOnlineTrendControl.1
WinCC Push Button Control	PBUTTON.PbuttonCtrl.1
WinCC Slider Control	SLIDER.SliderCtrl.1
WinCC RulerControl	CCAxTrendRulerControl.AxRulerControl.1
WinCC UserArchiveControl	CCAxUserArchiveControl.AxUserArchiveControl.1

Beispiel

Im folgenden Beispiel wird das ActiveX-Control "WinCC Gauge Control" in das aktive Bild eingefügt:

```
Sub AddActiveXControl()
'VBA119
Dim objActiveXControl As HMIActiveXControl
Set objActiveXControl = ActiveDocument.HMIObjects.AddActiveXControl("WinCC_Gauge",
"XGAUGE.XGaugeCtrl.1")
With ActiveDocument
.HMIObjects("WinCC_Gauge").Top = 40
.HMIObjects("WinCC_Gauge").Left = 40
End With
End Sub
```

Hinweis

Nach dem Ausführen der Methode wird der Graphics Designer nicht vollständig beendet. Die Datei "Grafexe.exe" bleibt im Speicher liegen. Um den Graphics Designer wieder zu starten, müssen Sie die Applikation "Grafexe.exe" im Taskmanager beenden.

Siehe auch

ActiveX Controls (Seite 1674)

HMIObjects-Objekt (Auflistung) (Seite 1957)

ActiveXControl-Objekt (Seite 1885)

VBA-Referenz (Seite 1735)

AddDotNetControl-Methode

Beschreibung

Fügt der Auflistung "HMIObjects" ein neues ".Net-Control"-Objekt hinzu.

Syntax

```
Expression.AddDotNetControl(ObjectName, ControlType, InGAC,
AssemblyInfo)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIObjects" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ObjectName (String)	Der Name des Objektes. Sie können das Objekt über seinen Namen in einer Auflistung ansprechen.
ControllType (String)	Der Namensraum des Objekts.
InGAC (String)	TRUE: Das Objekt ist im Global Assembly Cache registriert. FALSE: Das Objekt ist nicht im Global Assembly Cache registriert.
AssemblyInfo (String)	Wenn "InGAC=TRUE", wird die folgende Information eingegeben: Assembly Version Culture PublicKeyToken Wenn "InGAC=FALSE", wird lediglich in "Assembly" der Pfad des Objekts eingegeben.

Beispiel

Im folgenden Beispiel wird das Objekt ".NET-Control" aus dem Global Assembly Cache in das aktive Bild eingefügt:

```
'VBA851
Dim DotNetControl As HMIDotNetControl
Set DotNetControl = ActiveDocument.HMIObjects.AddDotNetControl("MyVBAControl",
"System.Windows.Forms.Label", True,"Assembly=System.Windows.Forms, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089")
```

AddFolder-Methode

Beschreibung

Legt einen neuen Ordner in der Bausteinbibliothek an. Das FolderItem-Objekt vom Typ "Folder" wird der FolderItems-Auflistung hinzugefügt.

Der so neu angelegte Ordner erhält als internen Namen "FolderX", wobei "X" für die laufende Nummer steht, beginnend bei 1. Verwenden Sie den internen Namen, um den Ordner in der FolderItems-Auflistung anzusprechen.

Syntax

Ausdruck.AddFolder(DefaultName)

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "FolderItems" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
DefaultName (String)	Der Name des Ordners, der angelegt werden soll.

Beispiel

Im folgenden Beispiel wird in der "Projekt Bibliothek" der Ordner "mein Ordner" angelegt:

```
Sub AddNewFolderToProjectLibrary()
'VBA120
Dim objProjectLib As HMISymbolLibrary
Set objProjectLib = Application.SymbolLibraries(2)
objProjectLib.FolderItems.AddFolder ("My Folder")
End Sub
```

Siehe auch

SymbolLibrary-Objekt (Seite 2035)

FolderItems-Objekt (Auflistung) (Seite 1941)

VBA-Referenz (Seite 1735)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

AddFromClipboard-Methode

Beschreibung

Kopiert ein Objekt aus der Zwischenablage in einen Ordner der Bausteinbibliothek. Das FolderItem-Objekt vom Typ "Item" wird der FolderItems-Auflistung hinzugefügt.

Hinweis

In der Zwischenablage müssen sich Objekte aus dem Graphics Designer befinden. Andere Inhalte (z.B. ASCII-Text) werden nicht eingefügt.

Syntax

Ausdruck.`AddFromClipboard(DefaultName)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "FolderItems" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
DefaultName (String)	Der Name, den das eingefügte Objekt in der Bausteinbibliothek tragen soll.

Beispiel

Im folgenden Beispiel wird das Objekt "PC" aus der "Globalen Bibliothek" in den Ordner "mein Ordner 3" der "Projekt Bibliothek" kopiert:

```
Sub CopyObjectFromGlobalLibraryToProjectLibrary()
'VBA121
Dim objGlobalLib As HMISymbolLibrary
Dim objProjectLib As HMISymbolLibrary
Set objGlobalLib = Application.SymbolLibraries(1)
Set objProjectLib = Application.SymbolLibraries(2)
objProjectLib.FolderItems.AddFolder ("My Folder3")
'
'copy object from "Global Library" to clipboard
With objGlobalLib
.FolderItems(2).Folder.Item(2).Folder.Item(1).CopyToClipboard
End With
'
'paste object from clipboard into "Project Library"
objProjectLib.FolderItems(objProjectLib.FindByDisplayName("My
Folder3").Name).Folder.AddFromClipBoard ("Copy of PC/PLC")
End Sub
```

Siehe auch

FolderItems-Objekt (Auflistung) (Seite 1941)

SymbolLibrary-Objekt (Seite 2035)

VBA-Referenz (Seite 1735)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

AddHMIObject-Methode

Beschreibung

Fügt der Auflistung "HMIObjects" ein neues Standard-, Smart- oder Windows-Objekt hinzu. Das Objekt wird in der linken, oberen Ecke des angegebenen Bildes eingefügt.

Hinweis

Verwenden Sie die AddActiveXControl-Methode, um ein ActiveXControl einzufügen.

Verwenden Sie die AddOLEObject-Methode, um ein OLE-Objekt einzufügen.

Syntax

Ausdruck.`AddHMIObject("ObjectName", "ProgID")`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIObjects" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ObjectName (String)	Der Name des Objektes. Sie können das Objekt über seinen Namen in einer Auflistung ansprechen.
ProgID (String)	Der Objekttyp, der eingefügt werden soll. Die "ProgID" erhalten Sie, indem Sie vor den VBA-Objektnamen das Präfix "HMI" stellen (z.B. "HMICircle" oder "HMIRectangle")

Beispiel

Im folgenden Beispiel wird ein Kreis in das aktive Bild eingefügt und dessen Hintergrundfarbe auf "Rot" gesetzt:

```
Sub AddCircleToActiveDocument()
'VBA122
```

```
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("VBA_Circle", "HMICircle")
objCircle.BackColor = RGB(255, 0, 0)
End Sub
```

Siehe auch

PieSegment-Objekt (Seite 1995)
TextList-Objekt (Seite 2037)
StatusDisplay-Objekt (Seite 2032)
StaticText-Objekt (Seite 2029)
Slider-Objekt (Seite 2025)
RoundRectangle-Objekt (Seite 2018)
RoundButton-Objekt (Seite 2015)
Rectangle-Objekt (Seite 2012)
PolyLine-Objekt (Seite 2001)
PictureWindow-Objekt (Seite 1992)
OptionGroup-Objekt (Seite 1989)
HMIObjects-Objekt (Auflistung) (Seite 1957)
Line-Objekt (Seite 1970)
IOField-Objekt (Seite 1959)
GraphicObject-Objekt (Seite 1943)
EllipseArc-Objekt (Seite 1929)
EllipseSegment-Objekt (Seite 1932)
Ellipse-Objekt (Seite 1926)
CircularArc-Objekt (Seite 1905)
Circle-Objekt (Seite 1902)
CheckBox-Objekt (Seite 1901)
Button-Objekt (Seite 1898)
BarGraph-Objekt (Seite 1893)
ApplicationWindow-Objekt (Seite 1891)
AddOLEObject-Methode (Seite 1794)
AddActiveXControl-Methode (Seite 1785)
VBA-Referenz (Seite 1735)

AddItem-Methode

Beschreibung

Kopiert ein Objekt aus dem angegebenen Bild in einen Ordner der Bausteinbibliothek. Das FolderItem-Objekt vom Typ "Item" wird der FolderItems-Auflistung hinzugefügt.

Syntax

Ausdruck.`Folder.AddItem "DefaultName", pHMIObject`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "FolderItems" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
DefaultName (String)	Der Name, den das eingefügte Objekt in der Bausteinbibliothek tragen soll.
pHMIObject (HMIObject)	Das Objekt, das aus dem angegebenen Bild in die Bausteinbibliothek eingefügt werden soll.

Beispiel

Im folgenden Beispiel soll ein Kreis in die "Projekt Bibliothek" kopiert werden. Dazu wird der Kreis ins aktive Bild eingefügt und in der "Projekt Bibliothek" der Ordner "mein Ordner2" angelegt:

```
Sub VBA123()
'VBA123
Dim objProjectLib As HMISymbolLibrary
Dim objCircle As HMICircle

Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle", "HMICircle")
Set objProjectLib = Application.SymbolLibraries(2)
objProjectLib.FolderItems.AddFolder ("My Folder2")
objProjectLib.FindByDisplayName("My Folder2").Folder.AddItem "ProjectLib Circle",
ActiveDocument.HMIObjects("Circle")
End Sub
```

Siehe auch

FolderItems-Objekt (Auflistung) (Seite 1941)

SymbolLibrary-Objekt (Seite 2035)

VBA-Referenz (Seite 1735)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

AddOLEObject-Methode

Beschreibung

Fügt der Auflistung "HMIObjects" ein neues OLE-Objekt hinzu. Das Objekt wird in der linken, oberen Ecke des angegebenen Bildes eingefügt.

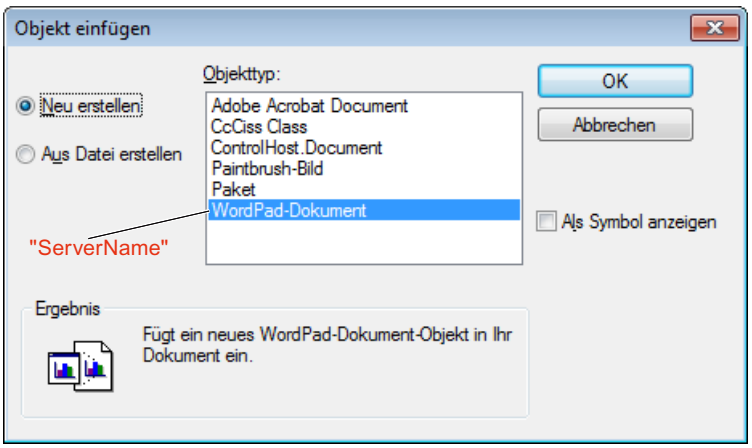
Syntax

```
Expression.AddOLEObject(ObjectName, ServerName, [CreationType],
[UseSymbol])
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIObjects" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ObjectName (String)	Der Name des Objektes. Sie können das Objekt über seinen Namen in einer Auflistung ansprechen.
ServerName (String)	Der Name der Anwendung, die das OLE-Objekt enthalten soll, oder der Dateiname mit Pfadangabe. Der Wert für "ServerName" entspricht dem "Objekttyp" im Dialog "Objekt einfügen": 

Parameter (Datentyp)	Beschreibung
CreationType (HMIOLEObjectCreation Type-)	Legt fest, ob das OLE-Objekt neu erstellt oder eine vorhandene Datei verwendet wird: • HMIOLEObjectCreationTypeDirect: Entspricht der Einstellung "Neu erstellen". Diese Einstellung wird verwendet, wenn Sie den Parameter nicht angeben. • HMIOLEObjectCreationTypeByLink: Entspricht der Einstellung "Aus Datei erstellen". Damit wird eine Kopie der Datei erstellen. Änderungen, die Sie im OLE-Objekt vornehmen, wirken sich nicht auf die Originaldatei aus. Den Dateinamen übergeben Sie mit dem Parameter "ServerName". • HMIOLEObjectCreationTypeByLinkWithReference: Wie oben, jedoch mit dem Unterschied, dass sich Änderungen im OLE-Objekt auf die Originaldatei auswirken. Den Dateinamen übergeben Sie mit dem Parameter "ServerName".
UseSymbol (Boolean)	TRUE, wenn das Standardsymbol des Dateityps verwendet wird. Ein Doppekklick auf das Symbol öffnet dann die entsprechende Anwendung. Standardeinstellung für diesen Parameter ist FALSE.

Beispiel

Im folgenden Beispiel wird ein OLE-Objekt in das aktive Bild eingefügt, das ein Wordpad-Dokument enthält:

```
Sub AddOLEObjectToActiveDocument()
'VBA124
Dim objOLEObject As HMIOLEObject
Set objOLEObject = ActiveDocument.HMIObjects.AddOLEObject("MS Wordpad Document",
"Wordpad.Document.1")
End Sub
```

Im folgenden Beispiel wird die AddOLEObject-Methode mit der Parameterangabe "HMIOLEObjectCreationTypeByLink" verwendet:

```
Sub AddOLEObjectByLink()
'VBA805
Dim objOLEObject As HMIOLEObject
Dim strFilename As String
'
'Add OLEObject by filename. In this case, the filename has to
'contain filename and path.
'Replace the definition of strFilename with a filename with path
'existing on your system
strFilename = Application.ApplicationDataPath & "Test.bmp"
Set objOLEObject = ActiveDocument.HMIObjects.AddOLEObject("OLEObject1", strFilename,
hmiOLEObjectCreationTypeByLink, False)
End Sub
```

Im folgenden Beispiel wird die AddOLEObject-Methode mit der Parameterangabe "HMIOLEObjectCreationTypeByLinkWithReference" verwendet:

```
Sub AddOLEObjectByLinkWithReference()
'VBA806
Dim objOLEObject As HMIOLEObject
Dim strFilename As String
'
'Add OLEObject by filename. In this case, the filename has to
'contain filename and path.
'Replace the definition of strFilename with a filename with path
'existing on your system
strFilename = Application.ApplicationDataPath & "Test.bmp"
Set objOLEObject = ActiveDocument.HMIObjects.AddOLEObject("OLEObject1", strFilename,
hmiOLEObjectCreationTypeByLinkWithReference, True)
End Sub
```

Siehe auch

OLEObject-Objekt (Seite 1987)

HMIObjects-Objekt (Auflistung) (Seite 1957)

VBA-Referenz (Seite 1735)

AddWPFControl-Methode

Beschreibung

Fügt der Auflistung "HMIObjects" ein neues "WPF-Control"-Objekt hinzu.

Syntax

```
Expression.AddWPFControl(ObjectName, ControlType, InGAC,
AssemblyInfo)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIObjects" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ObjectName (String)	Der Name des Objektes. Sie können das Objekt über seinen Namen in einer Auflistung ansprechen.
ControllType (String)	Der Namensraum des Objekts.

Parameter (Datentyp)	Beschreibung
InGAC (String)	TRUE: Das Objekt ist im Global Assembly Cache registriert. FALSE: Das Objekt ist nicht im Global Assembly Cache registriert.
AssemblyInfo (String)	Wenn "InGAC=TRUE", wird die folgende Information eingegeben: Assembly Version Culture PublicKeyToken Wenn "InGAC=FALSE", wird lediglich in "Assembly" der Pfad des Objekts eingegeben.

Beispiel

Im folgenden Beispiel wird das Objekt "WPF-Control" außerhalb des Global Assembly Cache in das aktive Bild eingefügt:

```
'VBA852
Dim WPFControl As HMIWPFControl
Set WPFControl = ActiveDocument.HMIObjects.AddWPFControl("MyWPFVBAControl",
"WinCCWPFControl.TestControl", False, "Assembly=Z:\TestControl\WinCCWPFControl.dll")
```

AlignBottom-Methode

Beschreibung

Richtet die ausgewählten Objekte im angegebenen Bild unten bündig aus. Die Ausrichtung orientiert sich dabei an dem Objekt, das Sie als erstes ausgewählt haben.

Syntax

Ausdruck.`AlignBottom()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte an verschiedenen Positionen ins aktuelle Bild eingefügt und anschließend unten bündig ausgerichtet:

```
Sub AlignSelectedObjectsBottom()
'VBA125
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
ActiveDocument.Selection.AlignBottom
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

AlignLeft-Methode

Beschreibung

Richtet die ausgewählten Objekte im angegebenen Bild linksbündig aus. Die Ausrichtung orientiert sich dabei an dem Objekt, das Sie als erstes ausgewählt haben.

Syntax

Ausdruck.`AlignLeft()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte an verschiedenen Positionen ins aktuelle Bild eingefügt und anschließend linksbündig ausgerichtet:

```
Sub AlignSelectedObjectsLeft()
'VBA126
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
ActiveDocument.Selection.AlignLeft
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

AlignRight-Methode

Beschreibung

Richtet die ausgewählten Objekte im angegebenen Bild rechtsbündig aus. Die Ausrichtung orientiert sich dabei an dem Objekt, das Sie als erstes ausgewählt haben.

Syntax

Ausdruck.`AlignRight()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte an verschiedenen Positionen ins aktuelle Bild eingefügt und anschließend rechtsbündig ausgerichtet:

```
Sub AlignSelectedObjectsRight()
'VBA127
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
ActiveDocument.Selection.AlignRight
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

AlignTop-Methode

Beschreibung

Richtet die ausgewählten Objekte im angegebenen Bild oben bündig aus. Die Ausrichtung orientiert sich dabei an dem Objekt, das Sie als erstes ausgewählt haben.

Syntax

Ausdruck.`AlignTop()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte an verschiedenen Positionen ins aktuelle Bild eingefügt und anschließend oben bündig ausgerichtet:

```
Sub AlignSelectedObjectsTop()
'VBA128
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
ActiveDocument.Selection.AlignTop
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

ArrangeMinimizedWindows-Methode

Beschreibung

Ordnet alle minimierten Bilder am unteren Rand des Graphics Designer an.

Syntax

Ausdruck.`ArrangeMinimizedWindows()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Application" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden alle minimierten Bilder am unteren Rand des Graphics Designer angeordnet. Damit dieses Beispiel funktioniert, müssen Sie mehrere Bilder im Graphics Designer minimiert haben:

```
Sub ArrangeMinimizedWindows()
'VBA129
Application.ArrangeMinimizedWindows
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

VBA-Referenz (Seite 1735)

BackwardOneLevel-Methode

Beschreibung

Verschiebt die selektierten Objekte in ihrer aktuellen Ebene um eine Stufe in den Hintergrund.

Syntax

Ausdruck.`BackwardOneLevel()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte im aktiven Bild eingefügt. Das zuletzt eingefügte Objekt wird dann um eine Stufe in den Hintergrund verschoben:

```
Sub MoveObjectOneLevelBackward()
'VBA173
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
```

```
.Left = 40
.Selected = False
End With
With objRectangle
.Top = 40
.Left = 40
.Width = 100
.Height = 50
.BackColor = RGB(255, 0, 255)
.Selected = True
End With
MsgBox "Objects created and selected!"
ActiveDocument.Selection.BackwardOneLevel
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

BringToFront-Methode

Beschreibung

Stellt die selektierten Objekte in ihrer aktuellen Ebene ganz in den Vordergrund.

Hinweis

Wenn die Methode "BringToFront" verwendet wird, kann sich die Reihenfolge der HMI-Objekte in der HMIObjects-Auflistung ändern.

Syntax

Ausdruck.`BringToFront()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte im aktiven Bild eingefügt. Das zuletzt eingefügte Objekt wird dann in den Vordergrund gestellt:

```
Sub MoveObjectToFront()
'VBA198
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 40
.Left = 40
.Width = 100
.Height = 50
.BackColor = RGB(255, 0, 255)
.Selected = False
End With
MsgBox "The objects circle and rectangle are created" & vbCrLf & "Only the circle is
selected!"
ActiveDocument.Selection.BringToFront
MsgBox "The selection is moved to the front."
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

CascadeWindows-Methode

Beschreibung

Ordnet alle geöffneten Bilder im Graphics Designer überlappend an.

Syntax

Ausdruck.`Methode(Parameter)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Application" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden alle geöffneten Bilder im Graphics Designer überlappend angeordnet. Damit dieses Beispiel funktioniert, müssen Sie mehrere Bilder im Graphics Designer geöffnet haben:

```
Sub CascadeWindows()
'VBA130
Application.CascadeWindows
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

Application-Objekt (Seite 1888)

CenterHorizontally-Methode

Beschreibung

Zentriert die ausgewählten Objekte im angegebenen Bild horizontal.

Syntax

Ausdruck.`CenterHorizontally()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte an verschiedenen Positionen ins aktuelle Bild eingefügt und anschließend horizontal zentriert:

```
Sub CenterSelectedObjectsHorizontally()
'VBA131
Dim objCircle As HMICircle
```

```
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
ActiveDocument.Selection.CenterHorizontally
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

CenterVertically-Methode

Beschreibung

Zentriert die ausgewählten Objekte im angegebenen Bild vertikal.

Syntax

Ausdruck.`CenterVertically()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte an verschiedenen Positionen ins aktuelle Bild eingefügt und anschließend vertikal zentriert:

```
Sub CenterSelectedObjectsVertically()
'VBA132
```

```
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
ActiveDocument.Selection.CenterVertically
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

CheckSyntax-Methode

Beschreibung

Prüft, ob die Syntax des angegebenen C-Skriptes korrekt ist.

Verwenden Sie die CheckSyntax-Methode zusammen mit der Compiled-Eigenschaft.

Syntax

Ausdruck.`CheckSyntax(CheckOK, Error)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "DynamicDialog" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
CheckOK (Boolean)	TRUE, wenn das angegebene C-Skript syntaktisch korrekt ist.
Error (String)	Der Meldungstext, der ausgegeben wird, wenn das C-Skript fehlerhaft ist.

Beispiel

--

Siehe auch

DynamicDialog-Objekt (Seite 1924)

VBA-Referenz (Seite 1735)

Close-Methode

Beschreibung

Schließt das angegebene Bild und entfernt es aus der Documents-Auflistung.

Hinweis

Nicht gespeicherte Änderungen gehen verloren.

Syntax 1

Ausdruck.`Close(FileName)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Documents" zurückgibt.

Syntax 2

Ausdruck.`Close()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Document" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
FileName (String)	Der Dateiname der zu schließenden PDL-Datei.

Beispiel

Im folgenden Beispiel wird das Bild "Test.PDL" geschlossen. Damit dieses Beispiel funktioniert, müssen Sie das Bild "Test.PDL" geöffnet haben:

```
Sub CloseDocumentUsingTheFileName()
'VBA134
Dim strFile As String
strFile = Application.ApplicationDataPath & "test.pdl"
Application.Documents.Close (strFile)
End Sub
Im folgenden Beispiel wird das aktive Bild im Graphics Designer geschlossen:
Sub CloseDocumentUsingActiveDocument()
'VBA135
ActiveDocument.Close
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

ActiveDocument-Eigenschaft (Seite 2067)

Documents-Objekt (Auflistung) (Seite 1923)

VBA-Referenz (Seite 1735)

CloseAll-Methode

Beschreibung

Schließt alle im Graphics Designer geöffneten Bilder und entfernt sie aus der Documents-Auflistung.

Hinweis

Nicht gespeicherte Änderungen gehen verloren.

Syntax

Ausdruck.`CloseAll()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Documents" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden alle im Graphics Designer geöffneten Bilder geschlossen:

```
Sub CloseAllDocuments()
'VBA136
Application.Documents.CloseAll
End Sub
```

Siehe auch

Documents-Objekt (Auflistung) (Seite 1923)

VBA-Referenz (Seite 1735)

ConvertToScript-Methode

Beschreibung

Wandelt den angegebenen Dynamik-Dialog in ein C-Skript um.

Bei der Umwandlung wird das dazugehörige DynamicDialog-Objekt gelöscht.

Hinweis

Die Umwandlung können Sie nicht rückgängig machen.

Syntax

Ausdruck.`ConvertToScript()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "DynamicDialog" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel wird ein Kreis in das aktive Bild eingefügt und der Radius mit dem Dynamik-Dialog dynamisiert. Anschließend wird der Dynamik-Dialog in ein C-Skript umgewandelt:

```
Sub ConvertDynamicDialogToScript()
'VBA137
```

```
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
'
'Create dynamic
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
'
'configure dynamic. "ResultType" defines the valuerange-type:
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.Add 50, 40
.AnalogResultInfos.Add 100, 80
.AnalogResultInfos.ElseCase = 100
MsgBox "The dynamic-dialog will be changed into a C-script."
.ConvertToScript
End With
End Sub
```

Siehe auch

DynamicDialog-Objekt (Seite 1924)

VBA-Referenz (Seite 1735)

CopySelection-Methode

Beschreibung

Kopiert alle im Bild ausgewählten Objekte in die Zwischenablage.

Syntax

Ausdruck.`CopySelection()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Document" oder "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden alle im aktiven Bild zwei Objekte eingefügt und ausgewählt. Die Auswahl wird kopiert und in ein neues Bild eingefügt:

```
Sub CopySelectionToNewDocument()
'VBA138
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim iNewDoc As Integer
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
'Instead of "ActiveDocument.CopySelection" you can also write:
'"ActiveDocument.Selection.CopySelection".
ActiveDocument.CopySelection
Application.Documents.Add hmiOpenDocumentTypeVisible
iNewDoc = Application.Documents.Count
Application.Documents(iNewDoc).PasteClipboard
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

ActiveDocument-Eigenschaft (Seite 2067)

Selection-Objekt (Auflistung) (Seite 2022)

PasteClipboard-Methode (Seite 1855)

Add-Methode (Documents-Auflistung) (Seite 1780)

Activate Methode (Seite 1776)

VBA-Referenz (Seite 1735)

CopyToClipboard-Methode

Beschreibung

Kopiert ein Objekt aus einem Ordner der Bausteinbibliothek in die Zwischenablage.

Syntax

Ausdruck.`CopyToClipboard()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein FolderItem-Objekt vom Typ "Item" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel wird das Objekt "PC" aus der "Globalen Bibliothek" in den Ordner "My Folder3" der "Projekt Bibliothek" kopiert:

```
Sub CopyObjectFromGlobalLibraryToProjectLibrary()
'VBA139
Dim objGlobalLib As HMISymbolLibrary
Dim objProjectLib As HMISymbolLibrary
Set objGlobalLib = Application.SymbolLibraries(1)
Set objProjectLib = Application.SymbolLibraries(2)
objProjectLib.FolderItems.AddFolder ("My Folder3")
'
'copy object from "Global Library" to clipboard
With objGlobalLib
.FolderItems(2).Folder.Item(2).Folder.Item(1).CopyToClipboard
End With
'
'paste object from clipboard into "Project Library"
objProjectLib.FolderItems(objProjectLib.FindByDisplayName("My
Folder3")).Folder.AddFromClipBoard ("Copy of PC/PLC")
End Sub
```

Siehe auch

SymbolLibrary-Objekt (Seite 2035)

FolderItem-Objekt (Seite 1939)

VBA-Referenz (Seite 1735)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

CreateCustomizedObject-Methode

Beschreibung

Erzeugt aus den ausgewählten Objekte im angegebenen Bild ein Anwender-Objekt. Das Anwender-Objekt müssen Sie dann im "Konfigurationsdialog" konfigurieren.

Weitere Informationen zu diesem Thema finden Sie unter "Anwender-Objekte" in dieser Dokumentation und unter "Anwender-Objekt" in der WinCC-Dokumentation .

Syntax

Ausdruck.`CreateCustomizedObject()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte an verschiedenen Positionen ins aktuelle Bild eingefügt und anschließend ein Anwender-Objekt erzeugt:

```
Sub CreateCustomizedObject()
'VBA140
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objCustObject As HMICustomizedObject
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
Set objCustObject = ActiveDocument.Selection.CreateCustomizedObject
objCustObject.ObjectName = "myCustomizedObject"
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

CustomizedObject-Objekt (Seite 1912)

VBA-Referenz (Seite 1735)

Anwender-Objekte (Seite 1687)

CreateDynamic-Methode

Beschreibung

Dynamisiert die angegebene Eigenschaft.

Syntax

Ausdruck.`CreateDynamic(DynamicType, [SourceCode])`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Property" zurückgibt.

Parameter

Sie müssen den Parameter "SourceCode" nur dann verwenden, wenn Sie die angegebene Eigenschaft mit dem Dynamik-Dialog dynamisieren wollen.

Bei allen anderen Dynamisierungsarten können Sie den Parameter weglassen.

Parameter (Datentyp)	Beschreibung
DynamicType (HMIDynamicCreationType)	Legt die Art der Dynamisierung fest: • hmiDynamicCreationTypeVariableDirect: Dynamisierung mit einer Variable. • hmiDynamicCreationTypeVariableIndirect: Dynamisierung mit einer Variable. Bei dieser Art der Dynamisierung wird nur der Name der Variablen angegeben, deren Wert zur Dynamisierung verwendet wird. • hmiDynamicCreationTypeScript: Dynamisierung mit einem Skript (C, VB). • hmiDynamicCreationTypeDynamicDialog: Dynamisierung mit dem Dynamik-Dialog.
SourceCode (String)	Legt die Funktion oder Variable fest, die zur Dynamisierung verwendet wird. Geben Sie den Variablennamen zusätzlich in Hochkommas an: "Variablenname"

Beispiel

In diesem Beispiel wird die Eigenschaft "Top" einen Kreises mit der Variable "NewDynamic" dynamisiert:

```
Sub AddDynamicAsVariableDirectToProperty()
'VBA141
Dim objVariableTrigger As HMIVariableTrigger
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("MyCircle", "HMICircle")
```

```
'Make property "Top" dynamic:
Set objVariableTrigger = objCircle.Top.CreateDynamic(hmiDynamicCreationTypeVariableDirect,
"NewDynamic")
'
'Define cycle-time
With objVariableTrigger
.CycleType = hmiCycleType_2s
End With
End Sub
```

Siehe auch

Property-Objekt (Seite 2005)

DeleteDynamic-Methode (Seite 1820)

VBA-Referenz (Seite 1735)

CreateGroup-Methode

Beschreibung

Erzeugt aus den ausgewählten Objekte im angegebenen Bild ein Gruppen-Objekt.

Weitere Informationen zu diesem Thema finden Sie unter "Gruppen-Objekte" in dieser Dokumentation und unter "Gruppen-Objekt" in der WinCC-Dokumentation .

Syntax

Ausdruck.`CreateGroup()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte an verschiedenen Positionen ins aktuelle Bild eingefügt und anschließend ein Gruppen-Objekt erzeugt:

```
Sub CreateGroup()
'VBA142
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objGroup As HMIGroup
```

```
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
Set objGroup = ActiveDocument.Selection.CreateGroup
objGroup.ObjectName = "myGroup"
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

Group-Objekt (Seite 1946)

VBA-Referenz (Seite 1735)

Gruppen-Objekte (Seite 1679)

D-M

GetDeclutterObjectSize-Methode

Beschreibung

Liest die Grenzwerte für das Ein- und Ausblenden von Objekten (Decluttering) im angegebenen Bild aus.

Syntax

Ausdruck.`GetDeclutterObjectSize(Min, Max)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Document" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
Min (Long)	Unterer Größenbereich in Pixel.
Max (Long)	Oberer Größenbereich in Pixel.

Beispiel

Im folgenden Beispiel werden die Decluttering-Grenzwerte des aktiven Bildes ausgelesen und ausgegeben:

```
Sub ReadSettingsOfPicture()
'VBA848
Dim objectsize_min As Long, objectsize_max As Long

ActiveDocument.GetDeclutterObjectSize objectsize_min, objectsize_max
MsgBox objectsize_min & " " & objectsize_max

End Sub
```

Delete-Methode

Beschreibung

Löscht das angegebene Objekt und entfernt es aus der Auflistung.

Syntax

Ausdruck.`Delete()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt der folgenden Typen zurückgibt:

- Assignment
- FolderItem
- LanguageText
- Menu
- MenuItem
- Object
- Toolbar
- ToolbarItem
- VariableTrigger
- View

Parameter

--

Beispiel

Im folgenden Beispiel wird das erste Objekt im aktiven Bild gelöscht. Damit dieses Beispiel funktioniert, müssen Sie im aktiven Bild mindestens ein Objekt angelegt haben:

```
Sub ObjectDelete()
'VBA143
ActiveDocument.HMIObjects(1).Delete
End Sub
```

Siehe auch

LanguageText-Objekt (Seite 1965)

View-Objekt (Seite 2062)

VariableTrigger-Objekt (Seite 2060)

ToolbarItem-Objekt (Seite 2043)

FolderItem-Objekt (Seite 1939)

HMIObject-Objekt (Seite 1955)

MenuItem-Objekt (Seite 1979)

Menu-Objekt (Seite 1976)

VBA-Referenz (Seite 1735)

DeleteAll-Methode

Beschreibung

Löscht alle ausgewählten Objekten im angegebenen Bild und entfernt Sie aus den Auflistungen "Selection" und "HMIObjects".

Syntax

Ausdruck.`DeleteAll()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte an verschiedenen Positionen ins aktuelle Bild eingefügt, danach ausgewählt und wieder gelöscht:

```
Sub DeleteAllSelectedObjects()
'VBA145
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
ActiveDocument.Selection.DeleteAll
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

DeleteDynamic-Methode

Beschreibung

Entfernt die Dynamik von der angegebenen Eigenschaft.

Syntax

Ausdruck.`DeleteDynamic`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Property" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel wird die mit der CreateDynamic-Methode erzeugte Dynamik wieder entfernt:

```
Sub DeleteDynamicFromObjectMeinKreis()
'VBA146
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects("MyCircle")
objCircle.Top.DeleteDynamic
End Sub
```

Siehe auch

Property-Objekt (Seite 2005)

CreateDynamic-Methode (Seite 1815)

VBA-Referenz (Seite 1735)

DeselectAll-Methode

Beschreibung

Hebt die Auswahl aller ausgewählten Objekte im angegebenen Bild auf und entfernt sie aus der Selection-Auflistung.

Syntax

Ausdruck.`DeselectAll()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte an verschiedenen Positionen ins aktuelle Bild eingefügt und ausgewählt. Danach werden alle ausgewählten Objekte wieder abgewählt:

```
Sub SelectObjectsAndDeselectThemAgain()
'VBA147
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
```

```
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects created and selected!"
ActiveDocument.Selection.DeselectAll
MsgBox "Objects deselected!"
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

Destroy-Methode

Beschreibung

Löst das angegebene Anwender-Objekt auf. Die Objekte bleiben erhalten.

Syntax

Ausdruck.`Destroy()`

Ausdruck

Ein Ausdruck, der ein Objekt vom Type "CustomizedObject" zurückgibt.

Parameter

--

Beispiel

Ein Beispiel für die Anwendung der Destroy-Methode finden Sie unter "Anwender-Objekt mit VBA bearbeiten" in dieser Dokumentation.

Siehe auch

CustomizedObject-Objekt (Seite 1912)

Destroy-Methode (Seite 1822)

Delete-Methode (Seite 1818)

CreateCustomizedObject-Methode (Seite 1813)

So bearbeiten Sie ein Anwender-Objekt mit VBA (Seite 1688)

DuplicateSelection-Methode

Beschreibung

Dupliziert die ausgewählten Objekte im angegebenen Bild. Die so erzeugten Objekte werden der HMIObjects-Auflistung hinzugefügt. Die Namen der neuen Objekte werden mit jeder Duplizierung fortlaufend nummeriert.

Wenn Sie z.B. ein Objekt namens "Kreis" duplizieren, heißt das duplizierte Objekt "Kreis1". Wenn Sie das Objekt "Kreis" noch einmal duplizieren, erhält es den Namen "Kreis2" usw.

Syntax

Ausdruck.`DuplicateSelection()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte an verschiedenen Positionen ins aktuelle Bild eingefügt und ausgewählt. Danach werden sie dupliziert:

```
Sub DuplicateSelectedObjects()
'VBA149
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
```

```
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects created and selected!"
ActiveDocument.Selection.DuplicateSelection
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

HMIObjects-Objekt (Auflistung) (Seite 1957)

VBA-Referenz (Seite 1735)

EvenlySpaceHorizontally-Methode

Beschreibung

Positioniert die ausgewählten Objekte des angegebenen Bildes horizontal im gleichen Abstand zueinander.

Syntax

Ausdruck.`EvenlySpaceHorizontally()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden drei Objekte an verschiedenen Positionen ins aktuelle Bild eingefügt und ausgewählt. Danach werden sie horizontal im gleichen Abstand zueinander positioniert:

```
Sub EvenlySpaceObjectsHorizontally()
'VBA150
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objEllipse As HMIEllipse
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
Set objEllipse = ActiveDocument.HMIObjects.AddHMIObject("sEllipse", "HMIEllipse")
```

```
With objCircle
.Top = 30
.Left = 0
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 42
.Selected = True
End With
With objEllipse
.Top = 48
.Left = 162
.BackColor = RGB(255, 0, 0)
.Selected = True
End With
MsgBox "Objects created and selected!"
ActiveDocument.Selection.EvenlySpaceHorizontally
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

Selection-Objekt (Auflistung) (Seite 2022)

EvenlySpaceVertically-Methode

Beschreibung

Positioniert die ausgewählten Objekte des angegebenen Bildes vertikal im gleichen Abstand zueinander.

Syntax

Ausdruck.`EvenlySpaceVertically()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden drei Objekte an verschiedenen Positionen ins aktuelle Bild eingefügt und ausgewählt. Danach werden sie vertikal im gleichen Abstand zueinander positioniert:

```
Sub EvenlySpaceObjectsVertically()
'VBA151
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objEllipse As HMIEllipse
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
Set objEllipse = ActiveDocument.HMIObjects.AddHMIObject("sEllipse", "HMIEllipse")
With objCircle
.Top = 30
.Left = 0
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 42
.Selected = True
End With
With objEllipse
.Top = 48
.Left = 162
.BackColor = RGB(255, 0, 0)
.Selected = True
End With
MsgBox "Objects created and selected"
ActiveDocument.Selection.EvenlySpaceVertically
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

Export-Methode

Beschreibung

Speichert das angegebene Bild als EMF-Datei.

Syntax

Ausdruck.`Export(Type, Path)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Document" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
Type (HMIImportExportType)	Legt das Format fest, in dem das exportierte Bild gespeichert wird.
Path (String)	Der Pfad, in den das Bild exportiert werden soll. Der Pfad muss angelegt sein.

Beispiel

```
Sub ExportAllPicturesAsPDL()
'VBA152
Dim iPictureCounter As Integer
Dim strPath As String

strPath = "C:\WinCC_PDL_Export\"

'Count Pictures in Graphics Designer...
For iPictureCounter = 1 To grafexe.Documents.Count
    '...and export each picture as PDL-file to specified path:
    grafexe.Documents(iPictureCounter).Export hmiImportExportTypePDL,
strPath
Next iPictureCounter
End Sub
```

Siehe auch

View-Objekt (Seite 2062)

Document-Objekt (Seite 1920)

Find-Methode

Beschreibung

Sucht im angegebenen Bild nach Objekten und gibt das Suchergebnis als Collection-Objekt zurück. Sie können nach folgenden Objekteigenschaften suchen:

- Typ
- Name
- Eigenschaft

Syntax

Ausdruck.`Find([ObjectType], [ObjectName], [PropertyName])`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIObjects" zurückgibt.

Parameter

Sie müssen mindestens einen der drei Parameter angeben.

Parameter (Datentyp)	Beschreibung
ObjectType (String)	Der Objekttyp, nach dem gesucht werden soll. Geben Sie dazu die "ProgID" des Objektes an. Die "ProgID" erhalten Sie , indem Sie vor den VBA-Objektnamen das Präfix "HMI" stellen (z.B. "HMICircle" oder "HMIRectangle")
ObjectName (String)	Der Name des Objektes, nach dem gesucht werden soll. Sie können Platzhalter (?, *) im Objektnamen verwenden, um Objekte mit ähnlichen Namen zu finden.
PropertyName (String)	Der Name der Objekteigenschaft, nach dem gesucht werden soll. Geben Sie dazu den VBA-Eigenschaftsnamen an (z.B. "BackColor" statt "Hintergrundfarbe").

Beispiel

Im folgenden Beispiel wird im aktiven Bild nach Objekten vom Typ "HMICircle" gesucht und das Suchergebnis ausgegeben:

```
Sub FindObjectsByType()
'VBA153
Dim colSearchResults As HMICollection
Dim objMember As HMIObject
Dim iResult As Integer
Dim strName As String
Set colSearchResults = ActiveDocument.HMIObjects.Find(ObjectType:="HMICircle")
For Each objMember In colSearchResults
iResult = colSearchResults.Count
strName = objMember.ObjectName
MsgBox "Found: " & CStr(iResult) & vbCrLf & "objectname: " & strName)
Next objMember
End Sub
```

Hinweis

Weitere Informationen zur Anwendung der Find-Methode finden Sie unter "Standard-, Smart- und Windows-Objekte bearbeiten" in dieser Dokumentation.

Siehe auch

Type-Eigenschaft (Seite 2392)

Name-Eigenschaft (Seite 2303)

Property-Objekt (Seite 2005)

HMIObjects-Objekt (Auflistung) (Seite 1957)

So bearbeiten Sie Standard-, Smart- Windows- und Rohr-Objekte (Seite 1667)

VBA-Referenz (Seite 1735)

FindByDisplayName-Methode

Beschreibung

Sucht in der gesamten Bausteinbibliothek nach dem angegebenen Objekt. Als Suchergebnis wird ein FolderItem-Objekt zurückgegeben.

Hinweis

Der Anzeigename des Objektes ist sprachabhängig. Beim Suchen wird nur die aktuell eingestellte Sprache berücksichtigt. Die Suche endet beim ersten gefundenen Objekt.

Syntax

Ausdruck.`FindByDisplayName(DisplayName)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "SymbolLibrary" oder die Auflistung "FolderItems" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
DisplayName (String)	Der Anzeigename des Objektes, nach dem in der Bausteinbibliothek gesucht werden soll.

Beispiel

Im folgenden Beispiel wird in der globalen Bibliothek nach dem Objekt "PC" gesucht und dessen Anzeigename ausgegeben:

```
Sub FindObjectInSymbolLibrary()
'VBA154
Dim objGlobalLib As HMISymbolLibrary
Dim objFItem As HMIFolderItem
Set objGlobalLib = Application.SymbolLibraries(1)
Set objFItem = objGlobalLib.FindByDisplayName("PC")
MsgBox objFItem.DisplayName
End Sub
```

Siehe auch

FolderItem-Objekt (Seite 1939)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

FlipHorizontally-Methode

Beschreibung

Spiegelt die ausgewählten Objekte des angegebenen Bildes entlang der horizontalen Mittelachse.

Ob eine Spiegelung zugelassen wird, hängt vom Objekttyp ab (ein OLE-Objekt kann z.B. nicht gespiegelt werden). Durch eine Spiegelung werden die Eigenschaften entsprechend angepasst. Wenn Sie z.B. ein Objekt vom Typ "StaticText" an der horizontalen Mittelachse spiegeln, ändert sich der Wert der Eigenschaft "AlignmentTop" von "0" in "2".

Syntax

Ausdruck.`FlipHorizontally()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel wird ein StaticText-Objekt in das aktive Bild eingefügt und an der horizontalen Mittelachse gespiegelt:

```
Sub FlipObjectHorizontally()
'VBA155
Dim objStaticText As HMIStaticText
Dim strPropertyName As String
Dim iPropertyValue As Integer
Set objStaticText = ActiveDocument.HMIObjects.AddHMIObject("Textfield", "HMIStaticText")
strPropertyName = objStaticText.Properties("Text").Name
With objStaticText
.Width = 120
.Text = "Sample Text"
.Selected = True
iPropertyValue = .AlignmentTop
MsgBox "Value of '" & strPropertyName & "' before flip: " & iPropertyValue
ActiveDocument.Selection.FlipHorizontally
iPropertyValue = objStaticText.AlignmentTop
MsgBox "Value of '" & strPropertyName & "' after flip: " & iPropertyValue
End With
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

FlipVertically-Methode

Beschreibung

Spiegelt die ausgewählten Objekte des angegebenen Bildes entlang der vertikalen Mittelachse.

Ob eine Spiegelung zugelassen wird, hängt vom Objekttyp ab (ein OLE-Objekt kann z.B. nicht gespiegelt werden). Durch eine Spiegelung werden die Eigenschaften entsprechend angepaßt. Wenn Sie z.B. ein Objekt vom Typ "StaticText" an der vertikalen Mittelachse spiegeln, ändert sich der Wert der Eigenschaft "AlignmentLeft" von "0" in "2".

Syntax

Ausdruck.`FlipVertically()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel wird ein StaticText-Objekt in das aktive Bild eingefügt und an der vertikalen Mittelachse gespiegelt:

```
Sub FlipObjectVertically()
'VBA156
Dim objStaticText As HMIStaticText
Dim strPropertyName As String
Dim iPropertyValue As Integer
Set objStaticText = ActiveDocument.HMIObjects.AddHMIObject("Textfield", "HMIStaticText")
strPropertyName = objStaticText.Properties("Text").Name
With objStaticText
.Width = 120
.Text = "Sample Text"
.Selected = True
.AlignmentLeft = 0
iPropertyValue = .AlignmentLeft
MsgBox "Value of '" & strPropertyName & "' before flip: " & iPropertyValue
ActiveDocument.Selection.FlipVertically
iPropertyValue = objStaticText.AlignmentLeft
MsgBox "Value of '" & strPropertyName & "' after flip: " & iPropertyValue
End With
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

ForwardOneLevel-Methode

Beschreibung

Verschiebt die selektierten Objekte in ihrer aktuellen Ebene um eine Stufe in den Vordergrund.

Syntax

Ausdruck.`ForwardOneLevel()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte im aktiven Bild eingefügt. Das zuerst eingefügte Objekt wird dann um eine Stufe in den Vordergrund verschoben:

```
Sub MoveObjectOneLevelForward()
'VBA174
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 40
.Left = 40
.Width = 100
.Height = 50
.BackColor = RGB(255, 0, 255)
.Selected = False
End With
MsgBox "Objects created and selected!"
ActiveDocument.Selection.ForwardOneLevel
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

GetItemByPath-Methode

Beschreibung

Gibt ein FolderItem-Objekt (Ordner oder Objekt) zurück, das sich im angegebenen internen Zugriffspfad in der Bausteinbibliothek befindet.

Hinweis

Den internen Zugriffspfad erhalten Sie, wenn Sie in der Bausteinbibliothek im Kontextmenü den Befehl "Pfad kopieren" wählen. Der interne Zugriffspfad des Ordners oder Objektes wird damit in die Zwischenablage kopiert.

Syntax

Ausdruck.`GetItemByPath(PathName)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "SymbolLibrary" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
PathName (String)	Der interne Zugriffspfad, in dem sich das Objekt in der Bausteinbibliothek befindet.

Beispiel

In diesem Beispiel wird ein Objekt aus globalen Bibliothek zurückgegeben und anschließend dessen Anzeigename ausgegeben:

```
Sub ShowDisplayName()
'VBA157
Dim objGlobalLib As HMISymbolLibrary
Dim objFItem As HMIFolderItem
Set objGlobalLib = Application.SymbolLibraries(1)
Set objFItem = objGlobalLib.GetItemByPath("\Folder1\Folder2\Object1")
MsgBox objFItem.DisplayName
End Sub
```

Siehe auch

SymbolLibrary-Objekt (Seite 2035)

FolderItem-Objekt (Seite 1939)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

InsertFromMenuItem-Methode

Beschreibung

Fügt ein neues Symbol in eine vorhandene benutzerdefinierte Symbolleiste ein, das einen vorhandenen Menüeintrag eines benutzerdefinierten Menüs referenziert.

Verwenden Sie diese Methode, wenn Sie zusätzlich zu einem benutzerdefinierten Menü eine Symbolleiste einfügen wollen, welche dieselben Befehle enthält.

Syntax

Ausdruck.`InsertFromMenuItem(Position, Key, pMenuItem, DefaultToolTipText)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ToolbarItems" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
Position (Long)	Legt die Position des Symbols innerhalb der benutzerdefinierten Symbolleiste fest.
Key (Variant)	Identifiziert das Symbol. Verwenden Sie für "Key" eindeutige Bezeichnungen (z.B. "tItem1_1").
pMenuItem (HMIMenuItem)	Das MenuItem-Objekt, das referenziert werden soll.
DefaultToolTipText (String)	Legt Tooltiptext für das Symbol fest, der angezeigt wird, wenn Sie die Maus über das Symbol bewegen.

Beispiel

In diesem Beispiel werden ein benutzerdefiniertes Menü und eine benutzerdefinierte Symbolleiste in das aktive Bild eingefügt. Das Symbol ruft den Menüeintrag "Hello World" aus dem benutzerdefinierten Menü auf:

```
Sub ToolbarItem_InsertFromMenuItem()
'VBA158
Dim objMenu As HMIMenu
Dim objToolbarItem As HMIToolbarItem
Dim objToolbar As HMIToolbar
Dim objMenuItem As HMIMenuItem
Set objMenu = Application.CustomMenus.InsertMenu(1, "Menu1", "TestMenu")
'
'***********************************************
'* Note:
'* The object-reference has to be unique.
'***********************************************
'
Set objMenuItem = Application.CustomMenus(1).MenuItems.InsertMenuItem(1, "MenuItem1",
"Hello World")
Application.CustomMenus(1).MenuItems(1).Macro = "HelloWorld"
Set objToolbar = Application.CustomToolbars.Add("Toolbar1")
Set objToolbarItem = Application.CustomToolbars(1).ToolbarItems.InsertFromMenuItem(1,
"ToolbarItem1", objMenuItem, "Call's Hello World of TestMenu")
End Sub

Sub HelloWorld()
MsgBox "Procedure 'HelloWorld()' is execute."
```

```
End Sub
```

Siehe auch

ToolbarItems-Objekt (Auflistung) (Seite 2046)

InsertSeparator-Methode (Seite 1839)

Add-Methode (CustomToolbars-Auflistung) (Seite 1779)

VBA-Referenz (Seite 1735)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

InsertMenu-Methode

Beschreibung

Erstellt ein neues benutzerdefiniertes Menü. Es wird zwischen anwendungs- und bildspezifischen benutzerdefinierten Menüs unterschieden:

- Anwendungsspezifisches Menü: Ist an den Graphics Designer gebunden und auch dann noch sichtbar, wenn alle Bilder im Graphics Designer geschlossen sind. Platzieren Sie den VBA-Code entweder im Dokument "GlobalTemplateDocument" oder "ProjectTemplateDocument" und verwenden Sie die Eigenschaft "Application".
- Bildspezifisches Menü: Ist an ein bestimmtes Bild gebunden und bleibt solange sichtbar, wie das Bild aktiv ist. Platzieren Sie den VBA-Code im Dokument "ThisDocument" des gewünschten Bildes und verwenden Sie die Eigenschaft "ActiveDocument".

Syntax

Ausdruck.`InsertMenu(Position, Key, DefaultLabel)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "CustomMenus" zurückgibt.

Parameter

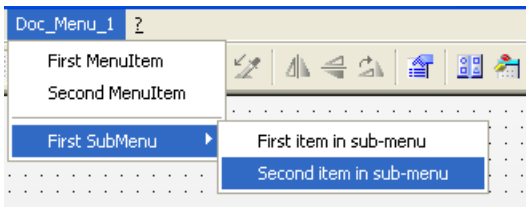
Parameter (Datentyp)	Beschreibung
Position (Long)	Legt die Position des benutzerdefinierten Menüs innerhalb der Menüleiste fest. Bildspezifische Menüs werden jedoch immer rechts von anwendungsspezifischen Menüs platziert.
Key (Variant)	Identifiziert das benutzerdefinierte Menü. Verwenden Sie für "Key" eindeutige Bezeichnungen (z.B. "DocMenu1").
DefaultLabel (String)	Der Name des benutzerdefinierten Menüs.

Beispiel

Im folgenden Beispiel wird im aktiven Bild ein benutzerdefiniertes Menü mit zwei Menüeinträgen und einem Untermenü mit zwei Einträgen angelegt. Das Untermenü wird durch eine Trennlinie optisch abgesetzt:

```
Sub CreateDocumentMenus()
'VBA159
Dim objDocMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Dim objSubMenu As HMIMenuItem
'
Set objDocMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DocMenu1", "Doc_Menu_1")
'
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(1, "dmItem1_1", "First MenuItem")
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(2, "dmItem1_2", "Second MenuItem")
'
'Insert a dividing rule into custumized menu:
Set objMenuItem = objDocMenu.MenuItems.InsertSeparator(3, "dSeparator1_3")
'
Set objSubMenu = objDocMenu.MenuItems.InsertSubMenu(4, "dSubMenu1_4", "First SubMenu")
'
Set objMenuItem = objSubMenu.SubMenu.InsertMenuItem(5, "dmItem1_5", "First item in sub-
menu")
Set objMenuItem = objSubMenu.SubMenu.InsertMenuItem(6, "dmItem1_6", "Second item in sub-
menu")
End Sub
```

Die Abbildung zeigt die erzeugte Menüstruktur.



Siehe auch

Menus-Objekt (Auflistung) (Seite 1977)

InsertSubmenu-Methode (Seite 1840)

InsertSeparator-Methode (Seite 1839)

InsertMenuItem-Methode (Seite 1838)

VBA-Referenz (Seite 1735)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

InsertMenuItem-Methode

Beschreibung

Fügt einen neuen Menüeintrag in ein benutzerdefiniertes Menü ein.

Syntax

Ausdruck.`InsertMenuItem(Position, Key, DefaultLabel)`

Ausdruck

Erforderlich. Ein Ausdruck der ein Objekt vom Typ "MenuItems" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
Position (Long)	Legt die Position des Untermenüs innerhalb des benutzerdefinierten Menüs fest.
Key (Variant)	Identifiziert das Untermenü. Verwenden Sie für "Key" eindeutige Bezeichnungen (z.B. "dSubMenu1_4").
DefaultLabel (String)	Legt den Namen des Untermenüs fest.

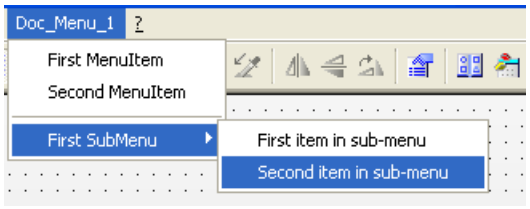
Beispiel

Im folgenden Beispiel wird im aktiven Bild ein benutzerdefiniertes Menü mit zwei Menüeinträgen und einem Untermenü mit zwei Einträgen angelegt. Das Untermenü wird durch eine Trennlinie optisch abgesetzt:

```
Sub CreateDocumentMenus()
'VBA160
Dim objDocMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Dim objSubMenu As HMIMenuItem
'
Set objDocMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DocMenu1", "Doc_Menu_1")
'
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(1, "dmItem1_1", "First MenuItem")
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(2, "dmItem1_2", "Second MenuItem")
'
'Insert a dividing rule into customized menu:
Set objMenuItem = objDocMenu.MenuItems.InsertSeparator(3, "dSeparator1_3")
'
Set objSubMenu = objDocMenu.MenuItems.InsertSubMenu(4, "dSubMenu1_4", "First SubMenu")
'
Set objMenuItem = objSubMenu.SubMenu.InsertMenuItem(5, "dmItem1_5", "First item in sub-
menu")
```

```
Set objMenuItem = objSubMenu.SubMenu.InsertMenuItem(6, "dmItem1_6", "Second item in sub-
menu")
End Sub
```

Die Abbildung zeigt die Menüstruktur:



Siehe auch

MenuItems-Objekt (Auflistung) (Seite 1982)

MenuItem-Objekt (Seite 1979)

InsertSubmenu-Methode (Seite 1840)

InsertSeparator-Methode (Seite 1839)

InsertMenu-Methode (Seite 1836)

VBA-Referenz (Seite 1735)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

InsertSeparator-Methode

Beschreibung

Fügt eine Trennlinie in ein benutzerdefiniertes Menü oder eine benutzerdefinierte Symbolleiste ein.

Syntax

Ausdruck.`InsertSeparator(Position, Key)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "MenuItems" oder "ToolbarItems" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
Position (Long)	Legt die Position der Trennlinie innerhalb des benutzerdefinierten Menüs oder der benutzerdefinierten Symbolleiste fest.
Key (Variant)	Identifiziert die Trennlinie. Verwenden Sie für "Key" eindeutige Bezeichnungen (z.B. "tSeparator1_2")

Beispiel

Im folgenden Beispiel wird im aktiven Bild eine benutzerdefinierte Symbolleiste zwei Symbole angelegt, die durch eine Trennlinie getrennt sind:

```
Sub AddDocumentSpecificCustomToolbar()
'VBA161
Dim objToolbar As HMIToolbar
Dim objToolbarItem As HMIToolbarItem
Set objToolbar = ActiveDocument.CustomToolbars.Add("DocToolbar")
'Add toolbar-item to userdefined toolbar
Set objToolbarItem = objToolbar.ToolbarItems.InsertToolbarItem(1, "tItem1_1", "First
symbol-icon")
Set objToolbarItem = objToolbar.ToolbarItems.InsertToolbarItem(3, "tItem1_3", "Second
symbol-icon")
'
'Insert dividing rule between first and second symbol-icon
Set objToolbarItem = objToolbar.ToolbarItems.InsertSeparator(2, "tSeparator1_2")
End Sub
```

Siehe auch

ToolbarItems-Objekt (Auflistung) (Seite 2046)

MenuItems-Objekt (Auflistung) (Seite 1982)

InsertToolbarItem-Methode (Seite 1842)

VBA-Referenz (Seite 1735)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

InsertSubmenu-Methode

Beschreibung

Fügt ein Untermenü in ein vorhandenes benutzerdefiniertes Menü ein.

Syntax

Ausdruck.`InsertSubmenu(Position, Key, DefaultLabel)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "MenuItem" zurückgibt.

Parameter

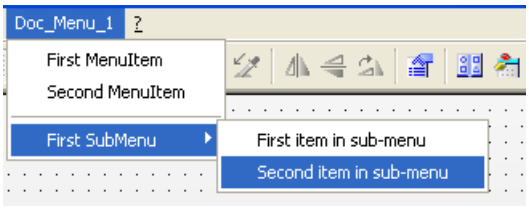
Parameter (Datentyp)	Beschreibung
Position (Long)	Legt die Position des Untermenüs innerhalb des benutzerdefinierten Menüs fest.
Key (Variant)	Identifiziert das Untermenü. Verwenden Sie für "Key" eindeutige Bezeichnungen (z.B. "dSubMenu1_4").
DefaultLabel (String)	Legt den Namen des Untermenüs fest.

Beispiel

Im folgenden Beispiel wird im aktiven Bild ein benutzerdefiniertes Menü mit zwei Menüeinträgen und einem Untermenü mit zwei Einträgen angelegt. Das Untermenü wird durch eine Trennlinie optisch abgesetzt:

```
Sub CreateDocumentMenus()
'VBA162
Dim objDocMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Dim objSubMenu As HMIMenuItem
'
Set objDocMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DocMenu1", "Doc_Menu_1")
'
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(1, "dmItem1_1", "First MenuItem")
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(2, "dmItem1_2", "Second MenuItem")
'
'Insert a dividing rule into customized menu:
Set objMenuItem = objDocMenu.MenuItems.InsertSeparator(3, "dSeparator1_3")
'
Set objSubMenu = objDocMenu.MenuItems.InsertSubMenu(4, "dSubMenu1_4", "First SubMenu")
'
Set objMenuItem = objSubMenu.SubMenu.InsertMenuItem(5, "dmItem1_5", "First item in sub-
menu")
Set objMenuItem = objSubMenu.SubMenu.InsertMenuItem(6, "dmItem1_6", "Second item in sub-
menu")
End Sub
```

Die Abbildung zeigt die Menüstruktur:



Siehe auch

MenuItem-Objekt (Seite 1979)

InsertSeparator-Methode (Seite 1839)

InsertMenuItem-Methode (Seite 1838)

InsertMenu-Methode (Seite 1836)

VBA-Referenz (Seite 1735)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

InsertToolbarItem-Methode

Beschreibung

Fügt ein neues Symbol in eine vorhandene benutzerdefinierte Symbolleiste ein.

Syntax

Ausdruck.`InsertToolbarItem(Position, Key, DefaultToolTipText)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "ToolbarItems" zurückgibt.

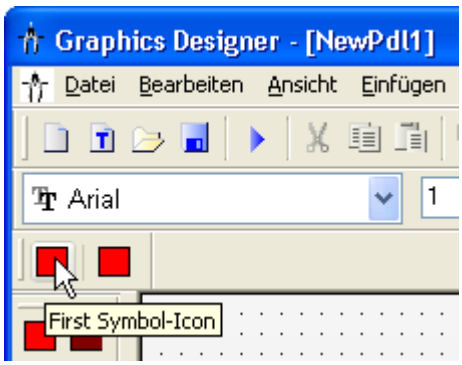
Parameter

Parameter (Datentyp)	Beschreibung
Position (Long)	Legt die Position des Symbols innerhalb der benutzerdefinierten Symbolleiste fest.
Key (Variant)	Identifiziert das Symbol. Verwenden Sie für "Key" eindeutige Bezeichnungen (z.B. "tItem1_1").
DefaultToolTipText (String)	Legt Tooltiptext für das Symbol fest, der angezeigt wird, wenn Sie die Maus über das Symbol bewegen.

Beispiel

Im folgenden Beispiel wird im aktiven Bild eine benutzerdefinierte Symbolleiste zwei Symbole angelegt, die durch eine Trennlinie getrennt sind:

```
Sub AddDocumentSpecificCustomToolbar()
'VBA163
Dim objToolbar As HMIToolbar
Dim objToolbarItem As HMIToolbarItem
Set objToolbar = ActiveDocument.CustomToolbars.Add("DocToolbar")
'Add toolbar-item to userdefined toolbar
Set objToolbarItem = objToolbar.ToolbarItems.InsertToolbarItem(1, "tItem1_1", "First
symbol-icon")
Set objToolbarItem = objToolbar.ToolbarItems.InsertToolbarItem(3, "tItem1_3", "Second
symbol-icon")
'
'Insert dividing rule between first and second symbol-icon
Set objToolbarItem = objToolbar.ToolbarItems.InsertSeparator(2, "tSeparator1_2")
End Sub
```



Siehe auch

ToolbarItems-Objekt (Auflistung) (Seite 2046)

InsertSeparator-Methode (Seite 1839)

Add-Methode (CustomToolbars-Auflistung) (Seite 1779)

VBA-Referenz (Seite 1735)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

IsCSLayerVisible-Methode

Beschreibung

Gibt TRUE zurück, wenn die angegebene CS Ebene sichtbar ist.

Syntax

Ausdruck.`IsCSLayerVisible(Index)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Document" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
Index (Variant)	Legt die CS Ebene fest. Wertebereich von 1 bis 32. Ebene0 entspricht dem Indexwert "1".

Beispiel

Im folgenden Beispiel wird festgestellt, ob die CS Ebene 1 in der Kopie des aktiven Bildes sichtbar ist und das Ergebnis ausgegeben:

```
Sub IsCSLayerVisible()
'VBA164
Dim objView As HMIView
Dim strLayerName As String
Dim iLayerIdx As Integer
Set objView = ActiveDocument.Views(1)
objView.Activate
iLayerIdx = 2
strLayerName = ActiveDocument.Layers(iLayerIdx).Name
If objView.IsCSLayerVisible(iLayerIdx) = True Then
MsgBox "CS " & strLayerName & " is visible"
Else
MsgBox "CS " & strLayerName & " is invisible"
End If
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

VBA-Referenz (Seite 1735)

Ebenen mit VBA bearbeiten (Seite 1659)

IsRTLayerVisible-Methode

Beschreibung

Gibt TRUE zurück, wenn die angegebene RT Ebene sichtbar ist.

Syntax

Ausdruck.`IsRTLayerVisible(Index)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Document" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
Index (Variant)	Legt die RT Ebene fest. Wertebereich von 1 bis 32. Ebene0 entspricht dem Indexwert "1".

Beispiel

Im folgenden Beispiel wird festgestellt, ob die RT Ebene 1 sichtbar ist und das Ergebnis ausgegeben:

```
Sub RTLayerVisibility()
'VBA165
Dim strLayerName As String
Dim iLayerIdx As Integer
iLayerIdx = 2
strLayerName = ActiveDocument.Layers(iLayerIdx).Name
If ActiveDocument.IsRTLayerVisible(iLayerIdx) = True Then
MsgBox "RT " & strLayerName & " is visible"
Else
MsgBox "RT " & strLayerName & " is invisible"
End If
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

VBA-Referenz (Seite 1735)

Ebenen mit VBA bearbeiten (Seite 1659)

Item-Methode

Beschreibung

Gibt ein Element aus einer Auflistung zurück.

Syntax

Ausdruck.`Item(Index)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
Index (Variant)	Der Name oder die Indexnummer eines Elements der Auflistung. Als Namen können Sie den Objektnamen verwenden, als Indexnummer einen numerischen Ausdruck (von 1 bis zum Wert der Count-Eigenschaft der Auflistung). Wenn der übergebene Wert mit keinem Element in der Auflistung übereinstimmt, tritt ein Fehler auf.

Beispiel

Hinweis

Die Item-Methode ist die Standardmethode für Auflistungen. Deshalb liefern die beiden folgenden Beispiele das gleiche Ergebnis.

Im folgenden Beispiel wird der Name des ersten Bildes im Graphics Designer ausgegeben:

```
Sub ShowDocumentNameLongVersion()
'VBA166
Dim strDocName As String
strDocName = Application.Documents.Item(3).Name
MsgBox strDocName
End Sub

Sub ShowDocumentNameShortVersion()
'VBA167
Dim strDocName As String
strDocName = Application.Documents(3).Name
MsgBox strDocName
End Sub
```

Siehe auch

VariableStateValues-Objekt (Auflistung) (Seite 2058)
Count-Eigenschaft (Seite 2153)
Views-Objekt (Auflistung) (Seite 2064)
VariableTriggers-Objekt (Auflistung) (Seite 2061)
ToolbarItems-Objekt (Auflistung) (Seite 2046)
Toolbars-Objekt (Auflistung) (Seite 2041)
SymbolLibraries-Objekt (Auflistung) (Seite 2036)
Selection-Objekt (Auflistung) (Seite 2022)
Properties-Objekt (Auflistung) (Seite 2004)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
MenuItems-Objekt (Auflistung) (Seite 1982)
Menus-Objekt (Auflistung) (Seite 1977)
Layers-Objekt (Auflistung) (Seite 1969)
LanguageTexts-Objekt (Auflistung) (Seite 1966)
LanguageFonts-Objekt (Auflistung) (Seite 1963)
GroupedObjects-Objekt (Auflistung) (Seite 1950)
FolderItems-Objekt (Auflistung) (Seite 1941)
Events-Objekt (Auflistung) (Seite 1936)
Documents-Objekt (Auflistung) (Seite 1923)
DataLanguages-Objekt (Auflistung) (Seite 1915)
ConnectionPoints-Objekt (Auflistung) (Seite 1910)
AnalogResultInfos-Objekt (Auflistung) (Seite 1887)
Actions-Objekt (Auflistung) (Seite 1884)
VBA-Referenz (Seite 1735)

ItemByLcid-Methode

Beschreibung

Wählt die Sprache aus, für die Sie die Schrifteinstellungen vornehmen wollen. Lese-Zugriff.

Hinweis

Sie können nur Sprachen auswählen, in denen Sie bereits projektiert haben.

Syntax

Ausdruck.`ItemByLcid(LangID)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "LanguageFonts" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
LangID (Long)	Die Sprachkennung. Die Liste mit den Sprachkennungen finden Sie z. B. in der Datei "Languages.csv", die Sie im Index der WinCC-Dokumentation finden.

Beispiel

Im folgenden Beispiel werden für eine Schaltfläche die Schriftattribute für Französisch und Englisch gesetzt. Gegenüber Englisch erscheint in Französisch auf der Schaltfläche eine kleinere Schrift mit konstanter Laufweite (Courier New, 12pt):

```
Sub ExampleForLanguageFonts()
'VBA168
Dim objLangFonts As HMILanguageFonts
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
objButton.Text = "Hello"
Set objLangFonts = objButton.LDFonts
'
'To make fontsettings for english:
With objLangFonts.ItemByLCID(1033)
.Family = "Times New Roman"
.Bold = False
.Italic = True
.Underlined = False
.Size = 14
End With
'
'To make fontsettings for french:
With objLangFonts.ItemByLCID(1036)
.Family = "Courier New"
.Bold = True
.Italic = False
.Underlined = True
.Size = 12
End With

End Sub
```

Siehe auch

LanguageFonts-Objekt (Auflistung) (Seite 1963)

LoadDefaultConfig-Methode

Beschreibung

Lädt die Datei, in der die Voreinstellungen für Objekte gespeichert werden. Die PDD-Datei befindet sich im Verzeichnis "GraCS" des aktuellen Projektes.

Syntax

Ausdruck.`LoadDefaultConfig(FileName)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Application" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
FileName (String)	Der Dateiname der PDD-Datei, die geladen werden soll.

Beispiel

Im folgenden Beispiel wird die Datei "Test.PDD" geladen. Damit dieses Beispiel funktioniert, müssen Sie die Datei zuvor gespeichert haben. Dazu können Sie die SaveDefaultConfig-Methode verwenden:

```
Sub LoadDefaultConfig()
'VBA169
Application.LoadDefaultConfig ("Test.PDD")
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

SaveDefaultConfig-Methode (Seite 1866)

VBA-Referenz (Seite 1735)

MoveOneLayerDown-Methode

Beschreibung

Verschiebt das ausgewählte Objekt im angegebenen Bild in die nächstniedrigere Ebene.

Syntax

Ausdruck.`MoveOneLayerDown()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel wird ein Kreis im aktiven Bild in der dritten Ebene eingefügt und dann in die nächstniedrigere Ebene verschoben:

```
Sub MoveObjectOneLayerDown()
'VBA170
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
.Layer = 3
MsgBox "Circle is inserted into layer" & Str(.Layer)
ActiveDocument.Selection.MoveOneLayerDown
MsgBox "Circle is moved into layer" & Str(.Layer)
End With
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

MoveOneLayerUp-Methode

Beschreibung

Verschiebt das ausgewählte Objekt im angegebenen Bild in die nächsthöhere Ebene.

Syntax

Ausdruck.`MoveOneLayerUp()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel wird ein Kreis im aktiven Bild in der dritten Ebene eingefügt und dann in die nächsthöhere Ebene verschoben:

```
Sub MoveObjectOneLayerUp()
'VBA171
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
.Layer = 3
MsgBox "Circle is inserted into layer" & Str(.Layer)
ActiveDocument.Selection.MoveOneLayerUp
MsgBox "Circle is moved into layer" & Str(.Layer)
End With
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

MoveSelection-Methode

Beschreibung

Verschiebt ein oder mehrere ausgewählte Objekte im Bild um die angegeben Koordinaten.

Hinweis

Wenn Sie ein oder mehrere ausgewählte Objekte an eine neue Position setzen wollen, verwenden Sie die Eigenschaften "Left" und "Top".

Syntax

Ausdruck.`MoveSelection(PosX, PosY)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Document" oder "Selection" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
PosX (Long)	Die Anzahl der Pixel, um welche die Auswahl horizontal bewegt werden soll.
PosY (Long)	Die Anzahl der Pixel, um welche die Auswahl vertikal bewegt werden soll.

Beispiel

Im folgenden Beispiel werden zwei Objekte an verschiedenen Positionen ins aktuelle Bild eingefügt und ausgewählt. Danach wird die Auswahl um 30 Pixel nach rechts und um 40 nach unten verschoben:

```
Sub MoveSelectionToNewPostion()
'VBA172
Dim nPosX As Long
Dim nPosY As Long
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
```

```
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
nPosX = 30
nPosY = 40
ActiveDocument.MoveSelection nPosX, nPosY
End Sub
```

Siehe auch

Top-Eigenschaft (Seite 2388)

Left-Eigenschaft (Seite 2267)

Document-Objekt (Seite 1920)

VBA-Referenz (Seite 1735)

O-Z

Open-Methode

Beschreibung

Öffnet ein vorhandenes Bild im Graphics Designer und fügt es der Documents-Auflistung hinzu.

Syntax

Ausdruck.`Open(FileName, [HMIOpenDocumentType])`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Documents" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
FileName (String)	Der Dateiname der zu öffnenden PDL-Datei. Wenn Sie die PDL-Datei nicht im "GraCS"-Verzeichnis des geöffneten Projektes abgespeichert haben, müssen Sie zusätzlich den Pfad mit angeben.
HMIOpenDocumentType (HMIDocumentType)	Legt fest, wie das Bild geöffnet wird: • HMIDocumentTypeVisible: Öffnet das Bild zur direkten Bearbeitung. Dies ist die Standardeinstellung, wenn Sie den Parameter nicht angeben. • HMIDocumentTypeInvisible: Öffnet das Bild unsichtbar, d.h. es wird nicht im Graphics Designer angezeigt. Sie können das Bild nur über die Documents-Auflistung ansprechen und mit der Hide-Eigeschaft wieder auf sichtbar setzen.

Beispiel

Im folgenden Beispiel wird das Bild "Test" geöffnet. Damit dieses Beispiel funktioniert, müssen Sie zuvor ein Bild mit dem Namen "Test" im Verzeichnis "GraCS" des geöffneten Projektes abgespeichert haben:

```
Sub OpenDocument()
'VBA175
Application.Documents.Open "Test.PDL", hmiOpenDocumentTypeVisible
End Sub
```

Siehe auch

Hide-Eigenschaft (Seite 2214)

Documents-Objekt (Auflistung) (Seite 1923)

VBA-Referenz (Seite 1735)

PasteClipboard-Methode

Beschreibung

Fügt den Inhalt der Zwischenablage in das angegebene Bild ein.

Hinweis

In der Zwischenablage müssen sich Objekte aus dem Graphics Designer befinden. Andere Inhalte (z.B. ASCII-Text) werden nicht eingefügt.

Syntax

Ausdruck.`PasteClipboard()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Document" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden alle im aktiven Bild ausgewählten Objekte in die Zwischenablage kopiert und danach in einem neuen Bild wieder eingefügt. Damit dieses Beispiel funktioniert, müssen Sie im aktiven Bild mindestens ein Objekt ausgewählt haben:

```
Sub CopySelectionToNewDocument()
'VBA176
Dim iNewDoc As String
ActiveDocument.CopySelection
Application.Documents.Add hmiOpenDocumentTypeVisible
iNewDoc = Application.Documents.Count
Application.Documents(iNewDoc).PasteClipboard
End Sub
```

Siehe auch

ActiveDocument-Eigenschaft (Seite 2067)

Document-Objekt (Seite 1920)

CopySelection-Methode (Seite 1811)

Add-Methode (Documents-Auflistung) (Seite 1780)

Activate Methode (Seite 1776)

VBA-Referenz (Seite 1735)

PrintDocument-Methode

Beschreibung

Druckt die angegebene Kopie des Bildes mit den aktuellen Druckereinstellungen.

Syntax

Ausdruck.`PrintDocument()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "View" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel wird vom aktiven Bild eine Kopie erzeugt und diese dann aktiviert und gedruckt:

```
Sub CreateAndPrintView()
'VBA177
Dim objView As HMIView
Set objView = ActiveDocument.Views.Add
objView.Activate
objView.PrintDocument
End Sub
```

Siehe auch

View-Objekt (Seite 2062)

VBA-Referenz (Seite 1735)

PrintProjectDocumentation-Methode

Beschreibung

Druckt die Projektdokumentation des aktuellen Bildes mit allen darin enthaltenen Objekten und ihren Eigenschaften über das Berichtssystem in WinCC (Report Designer) aus.

Die Druckeinstellungen (z.B. Seitenbereich) müssen Sie zuvor im Dialog "Druckauftrageigenschaften" eingestellt haben. Wählen Sie dazu im Graphics Designer den Menübefehl "Datei" > "Projektdokumentation einrichten".

Hinweis

Die Projektdokumentation wird auf dem im Report Designer eingestellten Drucker ausgegeben. Das Drucklayout können Sie mit dem Report Designer nach Ihren Wünschen gestalten.

Syntax

Ausdruck.`PrintProjectDocumentation()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Document" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel wird die Projektdokumentation für das aktive Bild ausgedruckt:

```
Sub ToPrintProjectDocumentation()
'VBA178
ActiveDocument.PrintProjectDocumentation
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

VBA-Referenz (Seite 1735)

Remove-Methode

Beschreibung

Entfernt ein Objekt aus einer Auswahl von Objekten oder aus einem Gruppen-Objekt.

Syntax

Ausdruck.`Remove(Index)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "GroupedObjects" oder "Selection" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
Index (Variant)	Der Name oder die Indexnummer des Objektes, das entfernt werden soll. Als Namen können Sie den Objektnamen verwenden, als Indexnummer einen numerischen Ausdruck (von 1 bis zum Wert der Count-Eigenschaft der Auflistung). Wenn der übergebene Wert mit keinem Element in der Auflistung übereinstimmt, tritt ein Fehler auf.

Beispiel

Im folgenden Beispiel werden zunächst drei Objekte ins aktive Bild eingefügt und ausgewählt. Danach wird ein Objekt aus der Auswahl entfernt und die verbleibenden Objekte gruppiert. Aus dem Gruppen-Objekt wird dann das erste Objekt entfernt:

```
Sub RemoveObjectFromGroup()
'VBA179
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objEllipse As HMIEllipse
Dim objGroup As HMIGroup
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
Set objEllipse = ActiveDocument.HMIObjects.AddHMIObject("sEllipse", "HMIEllipse")
With objCircle
.Top = 30
.Left = 0
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 42
.Selected = True
End With
With objEllipse
.Top = 48
.Left = 162
.Width = 40
.BackColor = RGB(255, 0, 0)
.Selected = True
End With
MsgBox "Objects selected!"
Set objGroup = ActiveDocument.Selection.CreateGroup
```

```
MsgBox "Group-object is created."
objGroup.GroupedHMIObjects.Remove ("sEllipse")
MsgBox "The ellipse is removed from group-object."
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

GroupedObjects-Objekt (Auflistung) (Seite 1950)

VBA-Referenz (Seite 1735)

Rotate-Methode

Beschreibung

Dreht das ausgewählte Objekt im angegebenen Bild im Uhrzeigersinn um 90°.

Syntax

Ausdruck.`Rotate()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte ins aktive Bild eingefügt und diese dann gruppiert. Das Gruppen-Objekt wird dann einmal gedreht:

```
Sub RotateGroupObject()
'VBA180
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objGroup As HMIGroup
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
With objRectangle
.Top = 30
.Left = 30
.Width = 80
.Height = 40
.Selected = True
```

```
End With
With objCircle
.Top = 30
.Left = 30
.BackColor = RGB(255, 255, 255)
.Selected = True
End With
MsgBox "Objects selected!"
Set objGroup = ActiveDocument.Selection.CreateGroup
MsgBox "Group-object created."
objGroup.Selected = True
ActiveDocument.Selection.Rotate
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

Selection-Objekt (Auflistung) (Seite 2022)

SameHeight-Methode

Beschreibung

Setzt die Eigenschaft "Height" bei allen ausgewählten Objekten im angegebenen Bild auf den kleinsten vorhandenen Wert.

Syntax

Ausdruck.`SameHeight()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden drei Objekte von unterschiedlicher Größe ins aktive Bild eingefügt. Danach werden alle Objekte ausgewählt und auf dieselbe Höhe gesetzt:

```
Sub ApplySameHeightToSelectedObjects()
'VBA181
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
```

```
Dim objEllipse As HMIEllipse
Dim objGroup As HMIGroup
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
Set objEllipse = ActiveDocument.HMIObjects.AddHMIObject("sEllipse", "HMIEllipse")
With objCircle
.Top = 30
.Left = 0
.Height = 15
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 42
.Height = 40
.Selected = True
End With
With objEllipse
.Top = 48
.Left = 162
.Width = 40
.Height = 120
.BackColor = RGB(255, 0, 0)
.Selected = True
End With
MsgBox "Objects selected!"
ActiveDocument.Selection.SameHeight
End Sub
```

Siehe auch

Height-Eigenschaft (Seite 2213)

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

SameWidth-Methode

Beschreibung

Setzt die Eigenschaft "Width" bei allen ausgewählten Objekten im angegebenen Bild auf den kleinsten vorhandenen Wert.

Syntax

Ausdruck.`SameWidth()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden drei Objekte von unterschiedlicher Größe ins aktive Bild eingefügt. Danach werden alle Objekte ausgewählt und auf dieselbe Breite gesetzt:

```
Sub ApplySameWidthToSelectedObjects()
'VBA182
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objEllipse As HMIEllipse
Dim objGroup As HMIGroup
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
Set objEllipse = ActiveDocument.HMIObjects.AddHMIObject("sEllipse", "HMIEllipse")
With objCircle
.Top = 30
.Left = 0
.Width = 15
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 42
.Width = 40
.Selected = True
End With
With objEllipse
.Top = 48
.Left = 162
.Width = 120
.BackColor = RGB(255, 0, 0)
.Selected = True
End With
MsgBox "Objects selected!"
ActiveDocument.Selection.SameWidth
End Sub
```

Siehe auch

Width-Eigenschaft (Seite 2486)

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

SameWidthAndHeight-Methode

Beschreibung

Setzt die Eigenschaften "Height" und "Width" bei allen ausgewählten Objekten im angegebenen Bild auf den kleinsten vorhandenen Wert.

Syntax

Ausdruck.`SameWidthAndHeight()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden drei Objekte von unterschiedlicher Größe ins aktive Bild eingefügt. Danach werden alle Objekte ausgewählt und auf dieselbe Höhe gesetzt:

```
Sub ApplySameWidthAndHeightToSelectedObjects()
'VBA183
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objEllipse As HMIEllipse
Dim objGroup As HMIGroup
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
Set objEllipse = ActiveDocument.HMIObjects.AddHMIObject("sEllipse", "HMIEllipse")
With objCircle
.Top = 30
.Left = 0
.Height = 15
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 42
.Width = 25
.Height = 40
.Selected = True
End With
With objEllipse
.Top = 48
.Left = 162
.Width = 40
.Height = 120
```

```
.BackColor = RGB(255, 0, 0)
.Selected = True
End With
MsgBox "Objects selected!"
ActiveDocument.Selection.SameWidthAndHeight
End Sub
```

Siehe auch

Width-Eigenschaft (Seite 2486)

Height-Eigenschaft (Seite 2213)

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

Save-Methode

Beschreibung

Speichert das angegebene Bild unter seinem aktuellen Namen.

Syntax

Ausdruck.`Save()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Document" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel wird das aktive Bild im Graphics Designer gespeichert:

```
Sub SaveDocument()
'VBA184
ActiveDocument.Save
End Sub
```

Siehe auch

ActiveDocument-Eigenschaft (Seite 2067)

Document-Objekt (Seite 1920)

VBA-Referenz (Seite 1735)

SaveAll-Methode

Beschreibung

Speichert alle im Graphics Designer geöffneten Bilder unter ihrem aktuellen Namen.

Syntax

Ausdruck.`SaveAll()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Documents" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden alle geöffneten Bilder im Graphics Designer gespeichert:

```
Sub SaveAllDocuments()
'VBA185
Application.Documents.SaveAll
End Sub
```

Siehe auch

Documents-Objekt (Auflistung) (Seite 1923)

VBA-Referenz (Seite 1735)

SaveAs-Methode

Beschreibung

Speichert das angegebene Bild unter einem neuen Namen.

Soll ein bereits vorhandenes Bild überschrieben werden, muss vor dem Aufruf der SaveAs-Methode sichergestellt sein, dass dieses Bild überschrieben werden darf. Dazu müssen Sie

die LockedByCreatorID-Eigenschaft beim zu überschreibenden Bild abfragen. Ansonsten wird ein Fehler in VBA ausgelöst.

Syntax

Ausdruck.`SaveAs(FileName)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Document" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
FileName (String)	Der Dateiname, unter dem das Bild gespeichert werden soll.

Beispiel

Im folgenden Beispiel wird das aktive Bild unter dem Namen "Test2.PDL" gespeichert.

```
Sub SaveDocumentAs()
'VBA186
ActiveDocument.SaveAs ("Test2.PDL")
End Sub
```

Siehe auch

LockedByCreatorID-Eigenschaft (Seite 2275)

ActiveDocument-Eigenschaft (Seite 2067)

Document-Objekt (Seite 1920)

VBA-Referenz (Seite 1735)

SaveDefaultConfig-Methode

Beschreibung

Speichert die Voreinstellungen für Objekte in einer PDD-Datei. Die Datei wird im Verzeichnis "GraCS" des aktuellen Projektes gespeichert.

Syntax

Ausdruck.`SaveDefaultConfig(FileName)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Application" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
FileName (String)	Der Dateiname der PDD-Datei.

Beispiel

Im folgenden Beispiel werden die Voreinstellungen für Objekte in der Datei "Test.PDD" gespeichert.

```
Sub SaveDefaultConfig()
'VBA187
Application.SaveDefaultConfig ("Test.PDD")
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

LoadDefaultConfig-Methode (Seite 1849)

VBA-Referenz (Seite 1735)

SelectAll-Methode

Beschreibung

Wählt alle Objekte im angegebenen Bild aus und fügt sie der Selection-Auflistung hinzu.

Syntax

Ausdruck.`SelectAll()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden drei Objekte ins aktive Bild eingefügt und anschließend ausgewählt:

```
Sub SelectAllObjectsInActiveDocument()
'VBA188
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objEllipse As HMIEllipse
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
Set objEllipse = ActiveDocument.HMIObjects.AddHMIObject("sEllipse", "HMIEllipse")
With objCircle
.Top = 30
.Left = 0
.Height = 15
End With
With objRectangle
.Top = 80
.Left = 42
.Width = 25
.Height = 40
End With
With objEllipse
.Top = 48
.Left = 162
.Width = 40
.Height = 120
.BackColor = RGB(255, 0, 0)
End With
ActiveDocument.Selection.SelectAll
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

SendToBack-Methode

Beschreibung

Stellt die selektierten Objekte in ihrer aktuellen Ebene ganz in den Hintergrund.

Hinweis

Wenn die Methode "SendToBack" verwendet wird, kann sich die Reihenfolge der HMI-Objekte in der HMIObjects-Auflistung ändern.

Syntax

Ausdruck.`SendToBack()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Selection" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden zwei Objekte im aktiven Bild eingefügt. Das zuerst eingefügte Objekt wird dann in den Hintergrund gestellt:

```
Sub SendObjectToBack()
'VBA197
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = False
End With
With objRectangle
.Top = 40
.Left = 40
.Width = 100
.Height = 50
.BackColor = RGB(255, 0, 255)
.Selected = True
End With
MsgBox "The objects circle and rectangle are created" & vbCrLf & "Only the rectangle is
selected!"
ActiveDocument.Selection.SendToBack
MsgBox "The selection is moved to the back."
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

VBA-Referenz (Seite 1735)

SetCSLayerVisible-Methode

Beschreibung

Blendet die angegebene CS Ebene ein oder aus.

Syntax

Ausdruck.`SetCSLayerVisible(Index, Val)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "View" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
Index (Variant)	Legt die CS Ebene fest, die ein- oder ausgeblendet werden soll. WeCSebereich von 1 bis 32.
Val (Boolean)	TRUE, wenn die angegebene CS Ebene sichtbar sein soll.

Beispiel

Im folgenden Beispiel wird die zweite CS Ebene in der Kopie des aktiven Bildes ausgeblendet:

```
Sub SetCSLayerVisible()
'VBA189
Dim objView As HMIView
Set objView = ActiveDocument.Views.Add
objView.Activate
objView.SetCSLayerVisible 2, False
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

VBA-Referenz (Seite 1735)

Ebenen mit VBA bearbeiten (Seite 1659)

SetOpenContext-Methode

Beschreibung

Die SetOpenContext-Methode wird dazu verwendet, das Passwort eines passwortgeschützten Bildes (Prozessbild oder Faceplate-Typ) zu hinterlegen und anschließend das Bild zu öffnen.

Syntax

Ausdruck.`SetOpenContext (Password)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Documents" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
Password (String)	Das Passwort des vorhandenen Bildes.

Beispiel

Im folgenden Beispiel werden mehrere Bilder ("A.pdl", "B.pdl" und "C.pdl") mit demselben Passwort "Test123" geöffnet. Um die Bilder zu öffnen, müssen Sie zuvor das Passwort der Bilder angeben. Damit das Passwort nicht weiter zugänglich bleibt, beenden Sie die SetOpenContext-Methode mit einem Leer-String "".

```
Sub OpenProtectedPicture()
'VBAxxx
Documents.SetOpenContext ("Test123")
Documents.Open ("A.pdl")
Documents.Open ("B.pdl")
Documents.Open ("C.pdl")
Documents.SetOpenContext ("")
End Sub
```

SetDeclutterObjectSize-Methode

Beschreibung

Legt den Größenbereich für das Ein-/Ausblenden von Objekten im angegebenen Bild fest. Wenn Objekthöhe und Objektbreite außerhalb des angegebenen Größenbereichs liegen, dann werden die Objekte ausgeblendet.

Die Eigenschaft "ObjectSizeDecluttering" muss auf TRUE gesetzt sein.

Syntax

Ausdruck.SetDeclutterObjectSize(Min, Max)

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Document" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
Min (Long)	Unterer Größenbereich in Pixel.
Max (Long)	Oberer Größenbereich in Pixel.

Beispiel

Im folgenden Beispiel werden im aktiven Bild die Einstellungen der untersten Ebene konfiguriert:

```
Sub ConfigureSettingsOfLayer()
'VBA190
Dim objLayer As HMILayer
Set objLayer = ActiveDocument.Layers(1)
With objLayer
'Configure "Layer 0"
.MinZoom = 10
.MaxZoom = 100
.Name = "Configured with VBA"
End With
'Define decluttering of objects:
With ActiveDocument
.LayerDecluttering = True
.ObjectSizeDecluttering = True
.SetDeclutterObjectSize 50, 100
End With
End Sub
```

Siehe auch

ObjectSizeDecluttering-Eigenschaft (Seite 2309)

Document-Objekt (Seite 1920)

VBA-Referenz (Seite 1735)

SetRTLayerVisible-Methode

Beschreibung

Blendet die angegebene RT Ebene ein oder aus.

Syntax

Ausdruck.`SetRTLayerVisible(Index, Val)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Document" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
Index (Variant)	Legt die RT Ebene fest, die ein- oder ausgeblendet werden soll. Wertebereich von 1 bis 32.
Val (Boolean)	TRUE, wenn die angegebene RT Ebene sichtbar sein soll.

Beispiel

Im folgenden Beispiel wird die erste RT Ebene im aktiven Bild ausgeblendet:

```
Sub SetRTLayerVisibleWithVBA()
'VBA191
ActiveDocument.SetRTLayerVisible 1, False
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

VBA-Referenz (Seite 1735)

Ebenen mit VBA bearbeiten (Seite 1659)

ShowPropertiesDialog-Methode

Beschreibung

Öffnet den Dialog "Objekteigenschaften".

Syntax

Ausdruck.`ShowPropertiesDialog()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Application" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel wird der Dialog "Objekteigenschaften" geöffnet:

```
Sub ShowPropertiesDialog()
'VBA192
Application.ShowPropertiesDialog
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)
VBA-Referenz (Seite 1735)

ShowSymbolLibraryDialog-Methode

Beschreibung

Öffnet die Bausteinbibliothek.

Syntax

Ausdruck.`ShowSymbolLibraryDialog()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Application" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel wird die Bausteinbibliothek geöffnet:

```
Sub ShowSymbolLibraryDialog()
'VBA193
Application.ShowSymbolLibraryDialog
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

VBA-Referenz (Seite 1735)

ShowTagDialog-Methode

Beschreibung

Öffnet den Dialog "Variablen".

Syntax

Ausdruck.`ShowTagDialog()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Application" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel wird der Dialog "Variablen" geöffnet:

```
Sub ShowTagDialog()
'VBA194
Application.ShowTagDialog
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

VBA-Referenz (Seite 1735)

TileWindowsHorizontally-Methode

Beschreibung

Ordnet alle geöffneten Bilder im Graphics Designer horizontal an.

Syntax

Ausdruck.`Methode()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Application" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden alle geöffneten Bilder im Graphics Designer horizontal angeordnet. Damit dieses Beispiel funktioniert, müssen Sie mehrere Bilder im Graphics Designer geöffnet haben:

```
Sub TileWindowsHorizontally()
'VBA195
Application.TileWindowsHorizontally
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

VBA-Referenz (Seite 1735)

TileWindowsVertically-Methode

Beschreibung

Ordnet alle geöffneten Bilder im Graphics Designer vertikal an.

Syntax

Ausdruck.`Methode()`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Application" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden alle geöffneten Bilder im Graphics Designer vertikal angeordnet. Damit dieses Beispiel funktioniert, müssen Sie mehrere Bilder im Graphics Designer geöffnet haben:

```
Sub TileWindowsVertically()
'VBA196
Application.TileWindowsVertically
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

VBA-Referenz (Seite 1735)

Ungroup-Methode

Beschreibung

Löst ein Gruppen-Objekt auf. Die Objekte bleiben erhalten.

Syntax

Ausdruck.`Ungroup(Parameter)`

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "Group" zurückgibt.

Parameter

--

Beispiel

Im folgenden Beispiel werden drei Objekte im aktiven Bild angelegt und aus diesen ein Gruppen-Objekt erzeugt. Danach wird das Gruppen-Objekt bewegt und wieder aufgelöst.

```
Sub DissolveGroup()
'VBA199
Dim objCircle As HMICircle
```

```
Dim objRectangle As HMIRectangle
Dim objEllipse As HMIEllipse
Dim objGroup As HMIGroup
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
Set objEllipse = ActiveDocument.HMIObjects.AddHMIObject("sEllipse", "HMIEllipse")
With objCircle
.Top = 30
.Left = 0
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 42
.Selected = True
End With
With objEllipse
.Top = 48
.Left = 162
.Width = 40
.BackColor = RGB(255, 0, 0)
.Selected = True
End With
MsgBox "Objects selected!"
Set objGroup = ActiveDocument.Selection.CreateGroup
MsgBox "Group-object is created."
With objGroup
.Left = 120
.Top = 300
MsgBox "Group-object is moved."
.UnGroup
MsgBox "Group is dissolved."
End With
End Sub
```

Siehe auch

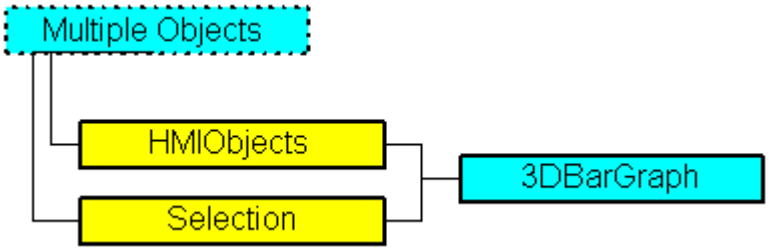
Group-Objekt (Seite 1946)
CreateGroup-Methode (Seite 1816)
VBA-Referenz (Seite 1735)
Gruppen-Objekte (Seite 1679)

3.5.1.7 Objekte und Auflistungen

0-9, A-C

3DBarGraph-Objekt

Beschreibung



Stellt das Objekt "3D-Balken" dar. Das 3DBarGraph-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.

VBA-Objektbezeichnung

HMI3DBarGraph

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "3D-Balken" im Bild anzulegen:

```
Sub Add3DBarGraph()
'VBA200
Dim obj3DBarGraph As HMI3DBarGraph
Set obj3DBarGraph = ActiveDocument.HMIObjects.AddHMIObject("3DBar", "HMI3DBarGraph")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub Edit3DBarGraph()
'VBA201
Dim obj3DBarGraph As HMI3DBarGraph
Set obj3DBarGraph = ActiveDocument.HMIObjects("3DBar")
obj3DBarGraph.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

```
Sub ShowNameOfFirstSelectedObject()
'VBA202
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Objekteigenschaften

Das Objekt 3D-Balken verfügt über folgende Eigenschaften:

- AngleAlpha
- AngleBeta
- Axe
- BorderColor
- Background
- BarDepth
- BarHeight
- BarWidth
- BaseX
- BaseY
- BorderColor
- BorderStyle
- BorderWidth
- Direction
- Height
- Layer00Checked
- Layer00Color
- Layer00Value
- Layer01Checked
- Layer01Color
- Layer01Value
- Layer02Checked
- Layer02Color
- Layer02Value
- Layer03Checked
- Layer03Color

- Layer03Value
- Layer04Checked
- Layer04Color
- Layer04Value
- Layer05Checked
- Layer05Color
- Layer05Value
- Layer06Checked
- Layer06Color
- Layer06Value
- Layer07Checked
- Layer07Color
- Layer07Value
- Layer08Checked
- Layer08Color
- Layer08Value
- Layer09Checked
- Layer09Color
- Layer09Value
- Layer10Checked
- Layer10Color
- Layer10Value
- Layer
- Left
- LightEffect
- Max
- Min
- Name
- Operation
- PasswordLevel
- PredefinedAngles
- Process
- ToolTipText
- Top
- Visible

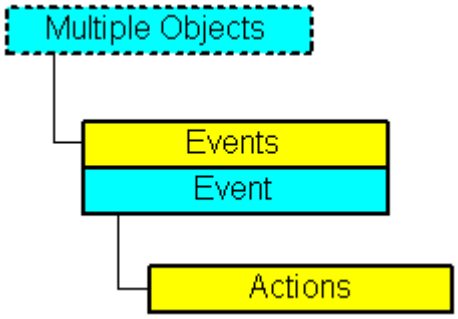
- Width
- ZeroPointValue

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)
Layer09Color-Eigenschaft (Seite 2255)
Layer01Checked-Eigenschaft (Seite 2237)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
ZeroPointValue-Eigenschaft (Seite 2491)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
Process-Eigenschaft (Seite 2337)
PredefinedAngels-Eigenschaft (Seite 2336)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Min-Eigenschaft (Seite 2300)
Max-Eigenschaft (Seite 2283)
LightEffect-Eigenschaft (Seite 2268)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Layer10Value-Eigenschaft (Seite 2258)
Layer10Color-Eigenschaft (Seite 2257)
Layer10Checked-Eigenschaft (Seite 2256)
Layer09Value-Eigenschaft (Seite 2256)
Layer09Checked-Eigenschaft (Seite 2254)
Layer08Value-Eigenschaft (Seite 2254)
Layer08Color-Eigenschaft (Seite 2253)
Layer08Checked-Eigenschaft (Seite 2252)
Layer07Value-Eigenschaft (Seite 2251)
Layer07Color-Eigenschaft (Seite 2251)
Layer07Checked-Eigenschaft (Seite 2250)
Layer06Value-Eigenschaft (Seite 2249)
Layer06Color-Eigenschaft (Seite 2248)
Layer06Checked-Eigenschaft (Seite 2248)
Layer05Value-Eigenschaft (Seite 2247)
Layer05Color-Eigenschaft (Seite 2246)
Layer05Checked-Eigenschaft (Seite 2246)

Actions-Objekt (Auflistung)

Beschreibung



Stellt eine Auflistung von Aktionen dar, die an ein Ereignis projektiert sind.

VBA-Objektbezeichnung

HMIActions

Verwendung

Verwenden Sie die AddAction-Methode, um an ein Ereignis ein oder mehrere Aktionen zu projektieren. In diesem Beispiel werden ein Button und ein Kreis in das aktive Bild eingefügt. In Runtime vergrößert sich der Kreisradius mit jedem Klick auf den Button:

```
Sub CreateVBActionToClickedEvent()
'VBA203
Dim objButton As HMIButton
Dim objCircle As HMICircle
Dim objVBScript As HMIScriptInfo
Dim strVBCode As String
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_VB", "HMICircle")
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objCircle
.Top = 100
.Left = 100
.BackColor = RGB(255, 0, 0)
End With
With objButton
.Top = 10
.Left = 10
.Text = "Increase Radius"
End With
'define event and assign sourcecode to it:
Set objVBScript = objButton.Events(1).Actions.AddAction(hmiActionCreationTypeVBScript)
strVBCode = "Dim myCircle" & vbCrLf & "Set myCircle = "
strVBCode = strVBCode & "HMIRuntime.ActiveScreen.ScreenItems(""Circle_VB"")"
strVBCode = strVBCode & vbCrLf & "myCircle.Radius = myCircle.Radius + 5"
With objVBScript
```

```
.SourceCode = strVBCode
End With
End Sub
```

Siehe auch

AddAction-Methode (Seite 1784)

Projektierung von ereignisgesteuerten Aktionen mit VBA (Seite 1704)

Parent-Eigenschaft (Seite 2317)

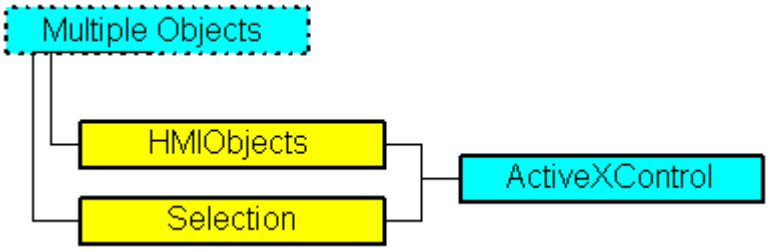
Item-Eigenschaft (Seite 2225)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

ActiveXControl-Objekt

Beschreibung



Stellt das Objekt ActiveX Control dar. Das ActiveXControl-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.

VBA-Objektbezeichnung

HMIActiveXControl

Verwendung

Verwenden Sie z.B. die AddActiveXControl-Methode, um ein ActiveX Control in ein Bild einzufügen. Im folgenden Beispiel wird das ActiveX Control "WinCC Gauge Control" in das aktive Bild eingefügt:

```
Sub AddActiveXControl()
'VBA204
```

```
Dim objActiveXControl As HMIActiveXControl
Set objActiveXControl = ActiveDocument.HMIObjects.AddActiveXControl("WinCC_Gauge",
"XGAUGE.XGaugeCtrl.1")
With ActiveDocument
.HMIObjects("WinCC_Gauge").Top = 40
.HMIObjects("WinCC_Gauge").Left = 40
End With
End Sub
```

Siehe auch

ServerName-Eigenschaft (Seite 2364)

AddActiveXControl-Methode (Seite 1785)

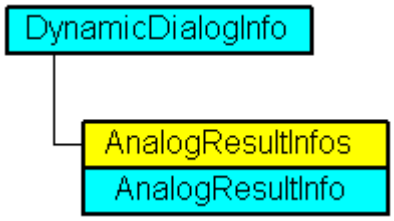
VBA-Referenz (Seite 1735)

ActiveX Controls (Seite 1674)

ProgID-Eigenschaft (Seite 2338)

AnalogResultInfo-Objekt

Beschreibung



Stellt einen analogen Wertebereich und den dazugehörigen Eigenschaftswert im Dynamik-Dialog dar. Das AnalogResultInfo-Objekt ist Element der AnalogResultInfos-Auflistung.

VBA-Objektbezeichnung

HMIAnalogResultInfo

Verwendung

Verwenden Sie das AnalogResultInfo-Objekt, um einen einzelnen Wertebereich und Eigenschaftswert zurückzugeben. Ein ausführliches Beispiel finden Sie unter "AnalogResultInfos-Objekt (Auflistung)" in dieser Dokumentation.

Siehe auch

AnalogResultInfos-Objekt (Auflistung) (Seite 1887)

Delete-Methode (Seite 1818)

Value-Eigenschaft (Seite 2408)

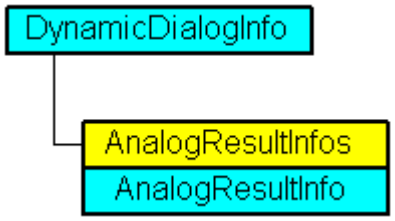
RangeTo-Eigenschaft (Seite 2347)

Parent-Eigenschaft (Seite 2317)

Application-Eigenschaft (Seite 2079)

AnalogResultInfos-Objekt (Auflistung)

Beschreibung



Eine Auflistung von AnalogResultInfo-Objekten, die alle analogen Wertebereiche und den dazugehörigen Eigenschaftswert im Dynamik-Dialog enthalten.

VBA-Objektbezeichnung

HMIAnalogResultInfos

Verwendung

Verwenden Sie die Add-Methode, um einen neuen Wertebereich im Dynamik-Dialog hinzuzufügen. Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert, ein Variablenname vergeben und drei analoge Wertebereiche angelegt:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA206
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.Add 50, 40
.AnalogResultInfos.Add 100, 80
.AnalogResultInfos.ElseCase = 100
End With
```

```
End Sub
```

Verwenden Sie AnalogResultInfos, um die AnalogResultInfos-Auflistung zurückzugeben. In diesem Beispiel werden die im Beispiel oben angelegten Wertebereiche ausgegeben:

```
Sub ShowAnalogResultInfosOfCircleRadius()
'VBA207
Dim colAResultInfos As HMIAnalogResultInfos
Dim objAResultInfo As HMIAnalogResultInfo
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Dim iAnswer As Integer
Dim varRange As Variant
Dim varValue As Variant
Set objCircle = ActiveDocument.HMIObjects("Circle_A")
Set objDynDialog = objCircle.Radius.Dynamic
Set colAResultInfos = objDynDialog.AnalogResultInfos
For Each objAResultInfo In colAResultInfos
varRange = objAResultInfo.RangeTo
varValue = objAResultInfo.value
iAnswer = MsgBox("Ranges of values from Circle_A-Radius:" & vbCrLf & "Range of value to: "
& varRange & vbCrLf & "Value of property: " & varValue, vbOKCancel)
If vbCancel = iAnswer Then Exit For
Next objAResultInfo
End Sub
```

Siehe auch

Add-Methode (AnalogResultInfos-Auflistung) (Seite 1777)

Parent-Eigenschaft (Seite 2317)

Item-Eigenschaft (Seite 2225)

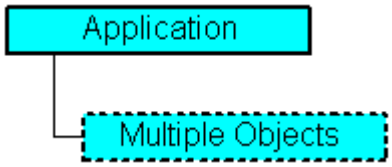
ElseCase-Eigenschaft (Seite 2169)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

Application-Objekt

Beschreibung



Stellt den Editor Graphics Designer dar. Das Application-Objekt enthält Eigenschaften und Methoden, die Objekte der Hauptebene zurückgeben. Zum Beispiel gibt die ActiveDocument ein Document-Objekt zurück.

VBA-Objektbezeichnung

HMIApplication

Verwendung

Verwenden Sie Application, um das Application-Objekt zurückzugeben. Im folgende Beispiel wird die Version der Anwendung ausgegeben:

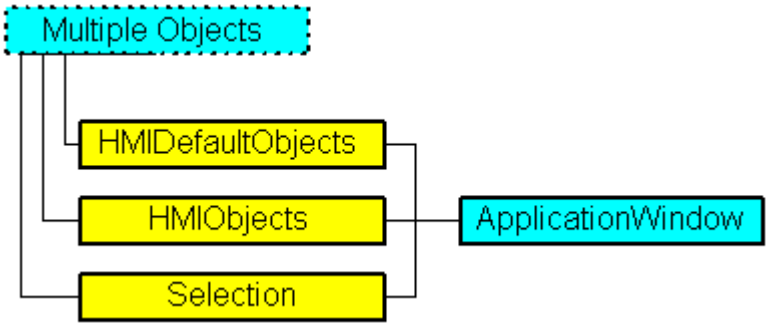
```
Sub ShowApplicationVersion()
'VBA208
MsgBox Application.Version
End Sub
```

Siehe auch

ShowTagDialog-Methode (Seite 1875)
CurrentDesktopLanguage-Eigenschaft (Seite 2156)
TileWindowsVertically-Methode (Seite 1876)
TileWindowsHorizontally-Methode (Seite 1876)
ShowSymbolLibraryDialog-Methode (Seite 1874)
ShowPropertiesDialog-Methode (Seite 1873)
SaveDefaultConfig-Methode (Seite 1866)
LoadDefaultConfig-Methode (Seite 1849)
CascadeWindows-Methode (Seite 1804)
ArrangeMinimizedWindows-Methode (Seite 1801)
Activate Methode (Seite 1776)
VBA-Referenz (Seite 1735)
WindowState-Eigenschaft (Seite 2490)
Visible-Eigenschaft (Seite 2484)
Version-Eigenschaft (Seite 2483)
VBE-Eigenschaft (Seite 2482)
VBAVersion-Eigenschaft (Seite 2482)
SymbolLibraries-Eigenschaft (Seite 2376)
ProjectType-Eigenschaft (Seite 2340)
ProjectName-Eigenschaft (Seite 2339)
ProfileName-Eigenschaft (Seite 2338)
Parent-Eigenschaft (Seite 2317)
Name-Eigenschaft (Seite 2303)
IsConnectedToProject-Eigenschaft (Seite 2223)
Documents-Eigenschaft (Seite 2167)
DefaultHMIObjects-Eigenschaft (Seite 2161)
CustomToolbars-Eigenschaft (Seite 2158)
CustomMenus-Eigenschaft (Seite 2158)
CurrentDataLanguage-Eigenschaft (Seite 2155)
ConfigurationFileName-Eigenschaft (Seite 2153)
AvailableDataLanguages-Eigenschaft (Seite 2084)
ApplicationDataPath-Eigenschaft (Seite 2080)
Application-Eigenschaft (Seite 2079)
ActiveDocument-Eigenschaft (Seite 2067)

ApplicationWindow-Objekt

Beschreibung



Stellt das Objekt "Applikationsfenster" dar. Das ApplicationWindow-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIApplicationWindow

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Applikationsfenster" im Bild anzulegen:

```
Sub AddApplicationWindow()
'VBA209
Dim objApplicationWindow As HMIApplicationWindow
Set objApplicationWindow = ActiveDocument.HMIObjects.AddHMIObject("AppWindow",
"HMIApplicationWindow")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditApplicationWindow()
'VBA210
Dim objApplicationWindow As HMIApplicationWindow
Set objApplicationWindow = ActiveDocument.HMIObjects("AppWindow")
objApplicationWindow.Sizeable = True
End Sub
```

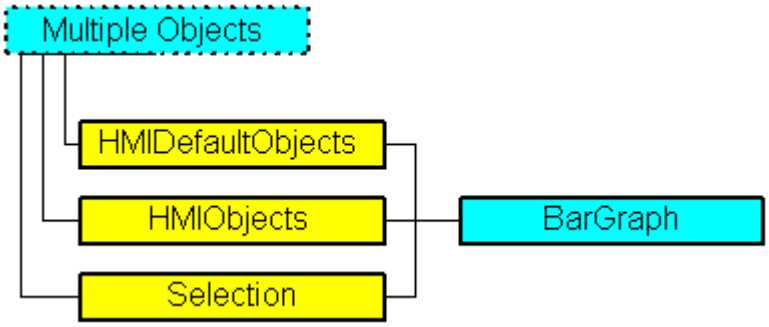
```
Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:
Sub ShowNameOfFirstSelectedObject()
'VBA211
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

Caption-Eigenschaft (Seite 2125)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
WindowBorder-Eigenschaft (Seite 2488)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
Sizeable-Eigenschaft (Seite 2369)
OnTop-Eigenschaft (Seite 2311)
Name-Eigenschaft (Seite 2303)
Moveable-Eigenschaft (Seite 2303)
MaximizeButton-Eigenschaft (Seite 2284)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
CloseButton-Eigenschaft (Seite 2137)
Application-Eigenschaft (Seite 2079)

BarGraph-Objekt

Beschreibung



Stellt das Objekt "Balken" dar. Das BarGraph-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smarten.

VBAbezeichnung

HMIBarGraph

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Balken" im Bild anzulegen:

```
Sub AddBarGraph()
'VBA212
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditBarGraph()
'VBA213
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects("Bar1")
objBarGraph.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

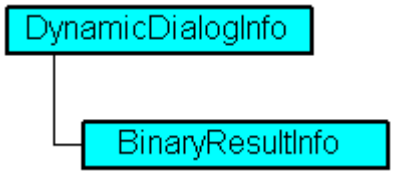
```
Sub ShowNameOfFirstSelectedObject()
'VBA214
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

TypeWarningHigh-Eigenschaft (Seite 2399)
Max-Eigenschaft (Seite 2283)
FillColor-Eigenschaft (Seite 2177)
BorderStyle-Eigenschaft (Seite 2115)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
ZeroPointValue-Eigenschaft (Seite 2491)
ZeroPoint-Eigenschaft (Seite 2490)
Width-Eigenschaft (Seite 2486)
WarningLow-Eigenschaft (Seite 2485)
WarningHigh-Eigenschaft (Seite 2484)
Visible-Eigenschaft (Seite 2484)
TypeWarningLow-Eigenschaft (Seite 2400)
TypeToleranceLow-Eigenschaft (Seite 2398)
TypeToleranceHigh-Eigenschaft (Seite 2398)
TypeLimitLow5-Eigenschaft (Seite 2397)
TypeLimitLow4-Eigenschaft (Seite 2396)
TypeLimitHigh5-Eigenschaft (Seite 2395)
TypeLimitHigh4-Eigenschaft (Seite 2394)
TypeAlarmLow-Eigenschaft (Seite 2394)
TypeAlarmHigh-Eigenschaft (Seite 2393)
Trend-Eigenschaft (Seite 2390)
TrendColor-Eigenschaft (Seite 2391)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
ToleranceLow-Eigenschaft (Seite 2386)
ToleranceHigh-Eigenschaft (Seite 2385)
ScalingType-Eigenschaft (Seite 2356)
Scaling-Eigenschaft (Seite 2355)
ScaleTicks-Eigenschaft (Seite 2354)
ScaleColor-Eigenschaft (Seite 2354)
RightComma-Eigenschaft (Seite 2350)
Process-Eigenschaft (Seite 2337)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)

BinaryResultInfo-Objekt

Beschreibung



Stellt die beiden binären (booleschen) Wertebereiche und die dazugehörigen Eigenschaftswerte im Dynamik-Dialog dar.

VBA-Objekbezeichnung

HMIBinaryResultInfo

Verwendung

Verwenden Sie BinaryResultInfo, um das BinaryResultInfo-Objekt zurückzugeben. Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert, ein Variablenname vergeben und den beiden binären Wertebereichen die dazugehörigen Eigenschaftswerte zugeordnet:

```
Sub AddDynamicDialogToCircleRadiusTypeBinary()
'VBA215
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_C", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeBool
.BinaryResultInfo.NegativeValue = 20
.BinaryResultInfo.PositiveValue = 40
End With
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

PositiveValue-Eigenschaft (Seite 2335)

Parent-Eigenschaft (Seite 2317)

NegativeValue-Eigenschaft (Seite 2305)

Application-Eigenschaft (Seite 2079)

BitResultInfo-Objekt

Beschreibung

Stellt die beiden Wertebereiche für Bit gesetzt/nicht gesetzt und die dazugehörigen Eigenschaftswerte im Dynamik-Dialog dar.

VBA-Objektbezeichnung

HMIBitResultInfo

Verwendung

Verwenden Sie BitResultInfo, um ein BitResultInfo-Objekt zurückzugeben. Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert, ein Variablenname vergeben sowie das zu setzende Bit definiert und den Zuständen "gesetzt"/"nicht gesetzt" die dazugehörigen Eigenschaftswerte zugeordnet:

```
Sub AddDynamicDialogToCircleRadiusTypeBit()
'VBA216
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_B", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeBit
.BitResultInfo.BitNumber = 1
.BitResultInfo.BitSetValue = 40
.BitResultInfo.BitNotSetValue = 80
End With
End Sub
```

Siehe auch

Delete-Methode (Seite 1818)

VBA-Referenz (Seite 1735)

BitSetValue-Eigenschaft (Seite 2105)

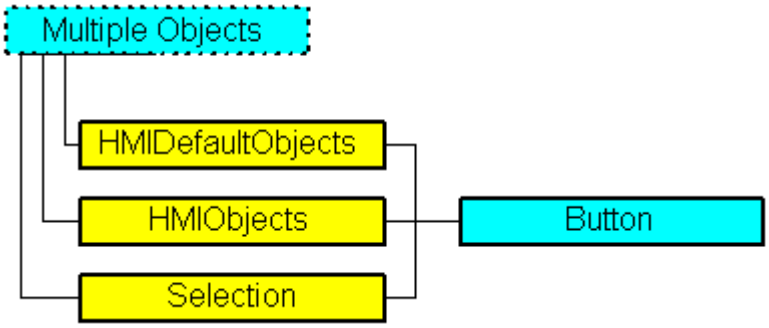
BitNumber-Eigenschaft (Seite 2104)

BitNotSetValue-Eigenschaft (Seite 2103)

Application-Eigenschaft (Seite 2079)

Button-Objekt

Beschreibung



Stellt das Objekt "Button" dar. Das Button-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smarten.

VBAbezeichnung

HMIButton

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Button" im Bild anzulegen:

```
Sub AddButton()
'VBA217
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button", "HMIButton")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditButton()
'VBA218
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects("Button")
objButton.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

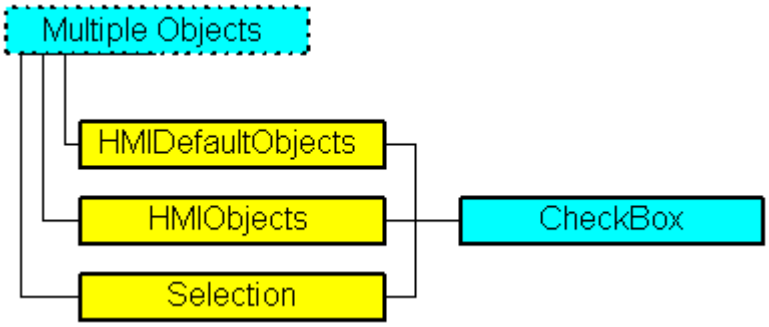
```
Sub ShowNameOfFirstSelectedObject()
'VBA219
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

ForeFlashColorOn-Eigenschaft (Seite 2207)
BorderColorBottom-Eigenschaft (Seite 2110)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
Text-Eigenschaft (Seite 2384)
PictureUp-Eigenschaft (Seite 2330)
PictureDown-Eigenschaft (Seite 2328)
PasswordLevel-Eigenschaft (Seite 2319)
Orientation-Eigenschaft (Seite 2315)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Hotkey-Eigenschaft (Seite 2216)
Height-Eigenschaft (Seite 2213)
ForeFlashColorOff-Eigenschaft (Seite 2206)
ForeColor-Eigenschaft (Seite 2206)
FontUnderline-Eigenschaft (Seite 2205)
FontSize-Eigenschaft (Seite 2204)
FontName-Eigenschaft (Seite 2203)
FontItalic-Eigenschaft (Seite 2203)
FontBold-Eigenschaft (Seite 2200)
FlashRateForeColor-Eigenschaft (Seite 2197)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashRateBackColor-Eigenschaft (Seite 2193)
FlashForeColor-Eigenschaft (Seite 2188)
FlashBorderColor-Eigenschaft (Seite 2186)
FlashBackColor-Eigenschaft (Seite 2185)
FillStyle-Eigenschaft (Seite 2181)
FillingIndex-Eigenschaft (Seite 2179)
Filling-Eigenschaft (Seite 2178)
FillColor-Eigenschaft (Seite 2177)

CheckBox-Objekt

Beschreibung



Stellt das Objekt "Check-Box" dar. Das CheckBox-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smarten.

VBAbezeichnung

HMICheckBox

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Check-Box" im Bild anzulegen:

```
Sub AddCheckBox()
'VBA220
Dim objCheckBox As HMICheckBox
Set objCheckBox = ActiveDocument.HMIObjects.AddHMIObject("CheckBox", "HMICheckBox")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditCheckBox()
'VBA221
Dim objCheckBox As HMICheckBox
Set objCheckBox = ActiveDocument.HMIObjects("CheckBox")
objCheckBox.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

```
Sub ShowNameOfFirstSelectedObject()
'VBA222
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

HMIObjects-Objekt (Auflistung) (Seite 1957)

HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)

AddHMIObject-Methode (Seite 1791)

VBA-Referenz (Seite 1735)

Objekte mit VBA bearbeiten (Seite 1662)

BitSetValue-Eigenschaft (Seite 2105)

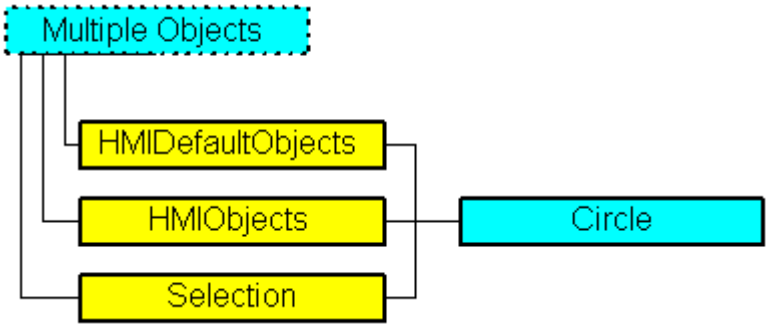
BitNumber-Eigenschaft (Seite 2104)

BitNotSetValue-Eigenschaft (Seite 2103)

Application-Eigenschaft (Seite 2079)

Circle-Objekt

Beschreibung



Stellt das Objekt "Kreis" dar. Das Circle-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMICircle

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Kreis" im Bild anzulegen:

```
Sub AddCircle()
'VBA223
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle", "HMICircle")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditCircle()
'VBA224
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects("Circle")
objCircle.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

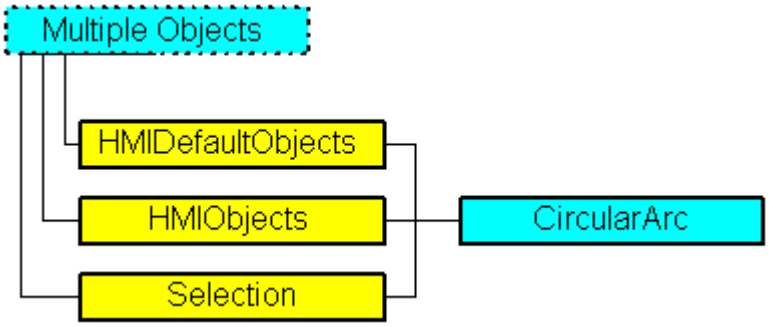
```
Sub ShowNameOfFirstSelectedObject()
'VBA225
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

FillColor-Eigenschaft (Seite 2177)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
Radius-Eigenschaft (Seite 2345)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashRateBackColor-Eigenschaft (Seite 2193)
FlashBorderColor-Eigenschaft (Seite 2186)
FlashBackColor-Eigenschaft (Seite 2185)
FillStyle-Eigenschaft (Seite 2181)
FillingIndex-Eigenschaft (Seite 2179)
Filling-Eigenschaft (Seite 2178)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
BorderColor-Eigenschaft (Seite 2108)
BorderBackColor-Eigenschaft (Seite 2107)
BackFlashColorOn-Eigenschaft (Seite 2094)
BackFlashColorOff-Eigenschaft (Seite 2093)
BackColor-Eigenschaft (Seite 2088)

CircularArc-Objekt

Beschreibung



Stellt das Objekt "Kreisbogen" dar. Das CiruclarArc-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMICircularArc

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Kreisbogen" im Bild anzulegen:

```
Sub AddCiruclarArc()
'VBA226
Dim objCiruclarArc As HMICircularArc
Set objCiruclarArc = ActiveDocument.HMIObjects.AddHMIObject("CircularArc",
"HMICircularArc")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditCiruclarArc()
'VBA227
Dim objCiruclarArc As HMICircularArc
Set objCiruclarArc = ActiveDocument.HMIObjects("CircularArc")
objCiruclarArc.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

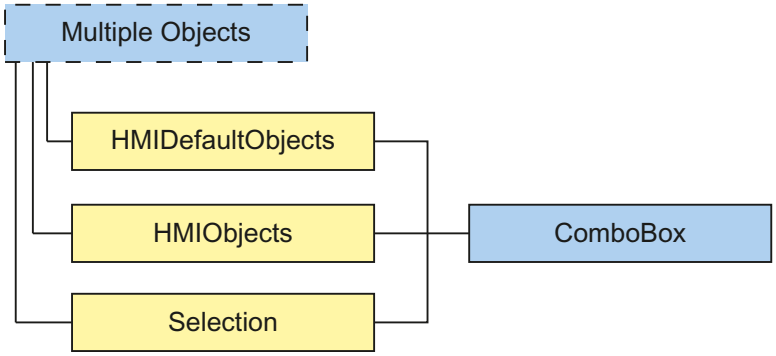
```
Sub ShowNameOfFirstSelectedObject()
'VBA228
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

HMIObjects-Objekt (Auflistung) (Seite 1957)
BorderBackColor-Eigenschaft (Seite 2107)
Selection-Objekt (Auflistung) (Seite 2022)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
StartAngle-Eigenschaft (Seite 2373)
Radius-Eigenschaft (Seite 2345)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashBorderColor-Eigenschaft (Seite 2186)
EndAngle-Eigenschaft (Seite 2171)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
BorderColor-Eigenschaft (Seite 2108)

ComboBox-Objekt

Beschreibung



Stellt das Objekt "Kombinationsfeld" dar. Das ComboBox-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Smart-, Windows- und Rohr-Objekten.

VBA-Objektbezeichnung

HMIComboBox

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Kombinationsfeld" im Bild anzulegen:

```
Sub AddComboBox()
'VBA822
Dim objComboBox As HMIComboBox
Set objComboBox = ActiveDocument.HMIObjects.AddHMIObject("Kombinationsfeld", "HMIComboBox")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditComboBox()
'VBA850
Dim objComboBox As HMIComboBox
Set objComboBox = ActiveDocument.HMIObjects("Kombinationsfeld")
objComboBox.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

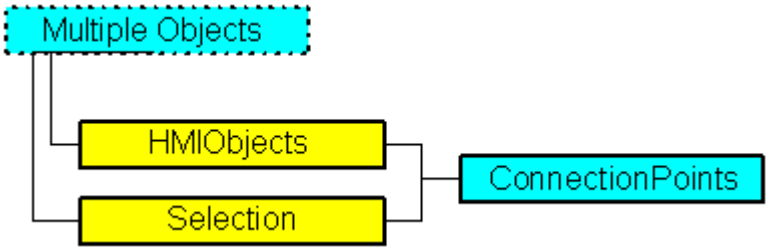
```
Sub ShowNameOfFirstSelectedObject()
'VBA824
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

ObjectName-Eigenschaft (Seite 2307)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Top-Eigenschaft (Seite 2388)
Width-Eigenschaft (Seite 2486)
Height-Eigenschaft (Seite 2213)
NumberLines-Eigenschaft (Seite 2306)
ForeColor-Eigenschaft (Seite 2206)
BorderColor-Eigenschaft (Seite 2108)
BorderBackColor-Eigenschaft (Seite 2107)
BackColor-Eigenschaft (Seite 2088)
BorderStyle-Eigenschaft (Seite 2115)
BorderWidth-Eigenschaft (Seite 2117)
FillColor-Eigenschaft (Seite 2177)
FillStyle-Eigenschaft (Seite 2181)
FontName-Eigenschaft (Seite 2203)
FontSize-Eigenschaft (Seite 2204)
FontBold-Eigenschaft (Seite 2200)
FontItalic-Eigenschaft (Seite 2203)
FontUnderline-Eigenschaft (Seite 2205)
AlignmentLeft-Eigenschaft (Seite 2075)
GlobalShadow-Eigenschaft (Seite 2209)
Index-Eigenschaft (Seite 2219)
Text-Eigenschaft (Seite 2384)
Operation-Eigenschaft (Seite 2312)
PasswordLevel-Eigenschaft (Seite 2319)
Visible-Eigenschaft (Seite 2484)
TooltipText-Eigenschaft (Seite 2387)
OperationMessage-Eigenschaft (Seite 2313)
OperationReport-Eigenschaft (Seite 2314)
SelText-Eigenschaft (Seite 2363)
SelIndex-Eigenschaft (Seite 2363)

ConnectionPoints-Objekt (Auflistung)

Beschreibung



Die Auflistung gibt die Anzahl der Punkte des angegebenen Objektes zurück, an denen der Verbinder angehängt werden kann.

VBA-Objektbezeichnung

HMIConnectionPoints

Objekteigenschaften

Das Objekt ConnectionPoints verfügt über folgende Eigenschaften:

- Application
- Count
- Item
- Parent

Beispiel 1

In diesem Beispiel wird ein Rechteck eingefügt und die Anzahl der Verbindungspunkte des Rechtecks ausgegeben:

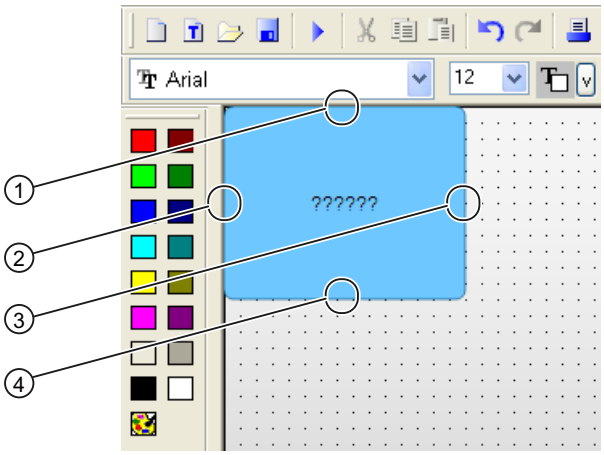
```
Sub CountConnectionPoints()
'VBA229
Dim objRectangle As HMIRectangle
Dim objConnPoints As HMIConnectionPoints
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
Set objConnPoints = ActiveDocument.HMIObjects("Rectangle1").ConnectionPoints
MsgBox "Rectangle1 has " & objConnPoints.Count & " connectionpoints."
End Sub
```

Beispiel 2:

In diesem Beispiel wird ein Textfeld eingefügt und auf die Verbindungspunkte über "ConnectionPoints.Item" zugegriffen. Die Koordinaten der Verbindungspunkte werden in einem Ausgabefenster angezeigt.

```
Sub GetConnectionPoints()
'VBA825
Dim xPos As Long
Dim yPos As Long
Dim objConnPoints As HMIConnectionPoints

Set objDoc = Application.ActiveDocument
Set objObject = objDoc.HMIObjects.AddHMIObject("Text","HMIStaticText")
Set objConnPoints = ActiveDocument.HMIObjects("Text").ConnectionPoints

For i = 1 To objConnPoints.Count
    xPos = objObject.ConnectionPoints.Item(i)(0)
    yPos = objObject.ConnectionPoints.Item(i)(1)
    MsgBox "Coordinates " & i & ". ConnectionPoint:" & Chr(13) & "x: " & xPos & Chr(13) &
"y: " & yPos
Next

End Sub
```

Das folgende Bild zeigt die Position der 4 Verbindungspunkte des Textfeldes:



Hinweis

Wenn Sie die Verbindungspunkte eines Verbinders mit VBA ansprechen, beginnt der Verbindungspunktindex mit "1".

Wenn Sie in der grafischen Oberfläche die Verbindungspunkte im Eigenschaftsfenster des Verbinders festlegen, beginnt der Verbindungspunktindex mit "0".

Die Indexnummern z.B. des unteren Verbindungspunktes im Bild sind wie folgt vergeben:

- VBA: Index = 3
- grafische Oberläche: Index = 2

Siehe auch

Parent-Eigenschaft (Seite 2317)

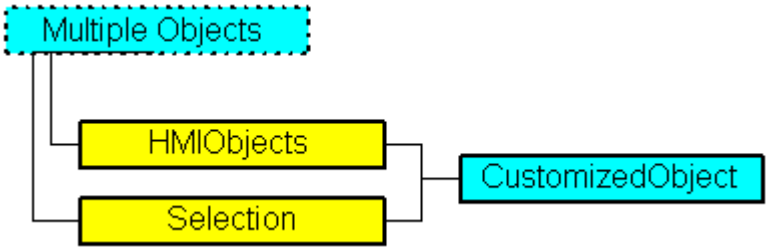
Item-Eigenschaft (Seite 2225)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

CustomizedObject-Objekt

Beschreibung



Stellt das Objekt "Anwender-Objekt" dar. Das CustomizedObject-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.

Beim CustomizedObject-Objekt sind nur die Objekteigenschaften verfügbar, die Sie im "Konfigurationsdialog" des Anwender-Objektes ausgewählt haben.

Hinweis

Sie können das CustomizedObject-Objekt nicht mit VBA konfigurieren.

Weitere Informationen zum Anwender-Objekt finden Sie in der WinCC-Dokumentation unter "Anwender-Objekt".

VBA-Objektbezeichnung

HMICustomizedObject

Verwendung

Verwenden Sie die CreateCustomizedObject-Methode mit der Selection-Auflistung, um ein neues Objekt "Anwender-Objekt" im Bild anzulegen:

```
Sub CreateCustomizedObject()
'VBA230
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objCustomizedObject As HMICustomizedObject
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_B", "HMIRectangle")
With objCircle
.Left = 10
.Top = 10
.Selected = True
End With
With objRectangle
.Left = 50
.Top = 50
.Selected = True
End With
MsgBox "objects created and selected!"
Set objCustomizedObject = ActiveDocument.Selection.CreateCustomizedObject
objCustomizedObject.ObjectName = "Customer-Object"
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditCustomizedObject()
'VBA231
Dim objCustomizedObject As HMICustomizedObject
Set objCustomizedObject = ActiveDocument.HMIObjects("Customer-Object")
MsgBox objCustomizedObject.ObjectName
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

HMIObjects-Objekt (Auflistung) (Seite 1957)

Destroy-Methode (Seite 1822)

Delete-Methode (Seite 1818)

CreateCustomizedObject-Methode (Seite 1813)

So bearbeiten Sie ein Anwender-Objekt mit VBA (Seite 1688)

VBA-Referenz (Seite 1735)

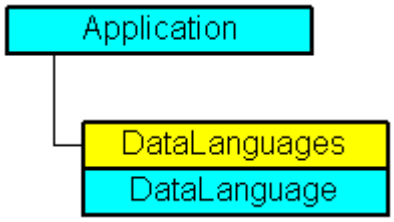
Anwender-Objekte (Seite 1687)

Objekte mit VBA bearbeiten (Seite 1662)

D-I

DataLanguage-Objekt

Beschreibung



Repräsentiert die installierte Projektiersprache, die mit ihrem Namen und der Sprachenkennung identifiziert wird. Das DataLanguage-Objekt ist Element der DataLanguages-Auflistung.

Die Liste mit den Sprachkennungen finden Sie in der WinCC-Dokumentation (Index > Language Code). Den dort angegebenen Hexadezimalwert müssen Sie in seinen entsprechenden Dezimalwert umwandeln.

VBA-Objektbezeichnung

HMIDataLanguage

Verwendung

Verwenden Sie die DataLanguages-Eigenschaft um ein einzelnes DataLanguage-Objekt zurückzugeben. Im folgenden Beispiel wird die erste installierte Projektiersprache ausgegeben:

```
Sub ShowFirstObjectOfCollection()
'VBA232
Dim strName As String
strName = ActiveDocument.Application.AvailableDataLanguages(1).LanguageName
MsgBox strName
End Sub
```

Siehe auch

DataLanguages-Objekt (Auflistung) (Seite 1915)

VBA-Referenz (Seite 1735)

Sprachabhängige Projektierung mit VBA (Seite 1626)

Parent-Eigenschaft (Seite 2317)

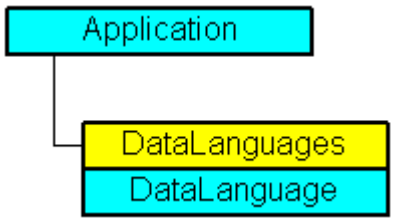
LanguageName-Eigenschaft (Seite 2232)

LanguageID-Eigenschaft (Seite 2232)

Application-Eigenschaft (Seite 2079)

DataLanguages-Objekt (Auflistung)

Beschreibung



Eine Auflistung von DataLanguage-Objekten, die alle installierten Projektiersprachen darstellen.

VBA-Objektbezeichnung

HMIDataLanguages

Verwendung

Verwenden Sie die AvailableDataLanguages-Eigenschaft, um die DataLanguages-Auflistung zurückzugeben. Im folgenden Beispiel werden die installierten Projektiersprachen ausgegeben:

```
Sub ShowDataLanguage()
'VBA233
Dim colDataLanguages As HMIDataLanguages
Dim objDataLanguage As HMIDataLanguage
Dim strLanguages As String
Dim iCount As Integer
iCount = 0
Set colDataLanguages = Application.AvailableDataLanguages
For Each objDataLanguage In colDataLanguages
If "" <> strLanguages Then strLanguages = strLanguages & "/"
strLanguages = strLanguages & objDataLanguage.LanguageName & " "
'Every 15 items of datalanguages output in a messagebox
If 0 = iCount Mod 15 And 0 <> iCount Then
MsgBox strLanguages
strLanguages = ""
End If
iCount = iCount + 1
Next objDataLanguage
MsgBox strLanguages
End Sub
```

Siehe auch

Sprachabhängige Projektierung mit VBA (Seite 1626)

DataLanguage-Objekt (Seite 1914)

Item-Methode (Seite 1845)

VBA-Referenz (Seite 1735)

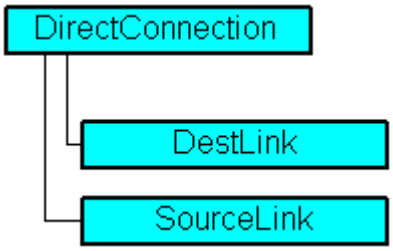
Parent-Eigenschaft (Seite 2317)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

DestLink-Objekt

Beschreibung



Stellt das Ziel bei der Direktverbindung dar.

VBA-Objektbezeichung

HMIDestLink

Verwendung

Verwenden Sie die DestinationLink-Eigenschaft, um das DestLink-Objekt zurückzugeben. Im folgenden Beispiel wird in Runtime bei einem Mausklick auf den Button die x-Position von "Rectangle_A" in die y-Position von "Rectangle_B" kopiert:

```
Sub DirectConnection()
'VBA234
Dim objButton As HMIButton
Dim objRectangleA As HMIRectangle
Dim objRectangleB As HMIRectangle
Dim objEvent As HMIEvent
Dim objDirConnection As HMIDirectConnection
Set objRectangleA = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_A", "HMIRectangle")
Set objRectangleB = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_B", "HMIRectangle")
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objRectangleA
.Top = 100
.Left = 100
End With
With objRectangleB
.Top = 250
.Left = 400
.BackColor = RGB(255, 0, 0)
End With
With objButton
.Top = 10
.Left = 10
.Width = 90
.Height = 50
.Text = "SetPosition"
End With
```

```
'
'Directconnection is initiated on mouseclick:
Set objDirConnection =
objButton.Events(1).Actions.AddAction(hmiActionCreationTypeDirectConnection)
With objDirConnection
'Sourceobject: Property "Top" of "Rectangle_A"
.SourceLink.Type = hmiSourceTypeProperty
.SourceLink.ObjectName = "Rectangle_A"
.SourceLink.AutomationName = "Top"
'
'Targetobject: Property "Left" of "Rectangle_B"
.DestinationLink.Type = hmiDestTypeProperty
.DestinationLink.ObjectName = "Rectangle_B"
.DestinationLink.AutomationName = "Left"
End With
End Sub
```

Siehe auch

DirectConnection-Objekt (Seite 1918)

VBA-Referenz (Seite 1735)

Type-Eigenschaft (Seite 2392)

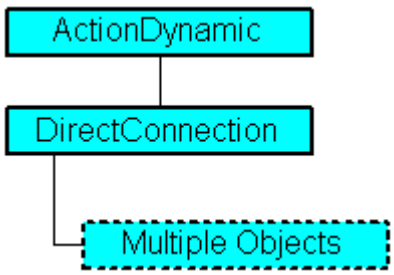
ObjectName-Eigenschaft (Seite 2307)

AutomationName-Eigenschaft (Seite 2082)

DestinationLink-Eigenschaft (Seite 2162)

DirectConnection-Objekt

Beschreibung



Stellt die Direktverbindung dar.

VBA-Objektbezeichnung

HMIDirectConnection

Verwendung

Verwenden Sie die DestinationLink- und SourceLink-Eigenschaften, um Quelle und Ziel der Direktverbindung zu projektieren. Im folgenden Beispiel wird in Runtime bei einem Mausklick auf den Button die x-Position von "Rectangle_A" in die y-Position von "Rectangle_B" kopiert:

```
Sub DirectConnection()
'VBA235
Dim objButton As HMIButton
Dim objRectangleA As HMIRectangle
Dim objRectangleB As HMIRectangle
Dim objEvent As HMIEvent
Dim objDirConnection As HMIDirectConnection
Set objRectangleA = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_A", "HMIRectangle")
Set objRectangleB = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_B", "HMIRectangle")
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objRectangleA
.Top = 100
.Left = 100
End With
With objRectangleB
.Top = 250
.Left = 400
.BackColor = RGB(255, 0, 0)
End With
With objButton
.Top = 10
.Left = 10
.Width = 90
.Height = 50
.Text = "SetPosition"
End With
'
'Directconnection is initiated on mouseclick:
Set objDirConnection =
objButton.Events(1).Actions.AddAction(hmiActionCreationTypeDirectConnection)
With objDirConnection
'Sourceobject: Property "Top" of "Rectangle_A"
.SourceLink.Type = hmiSourceTypeProperty
.SourceLink.ObjectName = "Rectangle_A"
.SourceLink.AutomationName = "Top"
'
'Targetobject: Property "Left" of "Rectangle_B"
.DestinationLink.Type = hmiDestTypeProperty
.DestinationLink.ObjectName = "Rectangle_B"
.DestinationLink.AutomationName = "Left"
End With
End Sub
```

Siehe auch

DestinationLink-Eigenschaft (Seite 2162)

SourceLink-Objekt (Seite 2028)

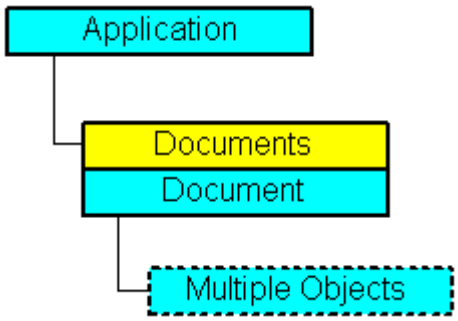
DestLink-Objekt (Seite 1917)

VBA-Referenz (Seite 1735)

SourceLink-Eigenschaft (Seite 2370)

Document-Objekt

Beschreibung



Stellt ein Bild im Graphics Designer dar. Das Document-Objekt ist Element der Documents-Auflistung.

VBA-Objektbezeichnung

HMIDocument

Verwendung

Verwenden Sie Documents(Index), um ein einzelnes Document-Objekt zurückzugeben. Im folgenden Beispiel wird der Dateiname des ersten Bildes angezeigt:

```
Sub ShowFirstObjectOfCollection()
'VBA236
Dim strName As String
strName = Application.Documents(3).Name
MsgBox strName
End Sub
```

Sie können auch das Objekt "Me" verwenden, wenn Sie das aktuelle Dokument ansprechen wollen:

```
Sub ShowDocumentName()
'VBA812
Dim obj As Document
set obj = Me
MsgBox obj.Name
End Sub
```

Verwenden Sie z.B. die SaveAs-Methode, um das Bild unter einem anderen Namen zu speichern. Im folgenden Beispiel wird das erste Bild unter dem Namen "Sicherungskopie Bild1" gespeichert:

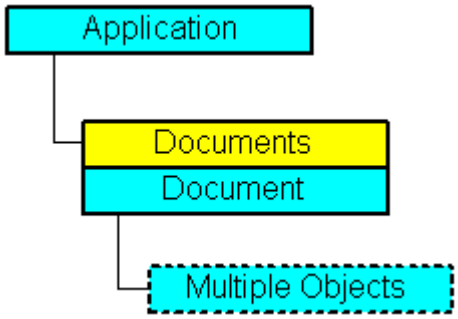
```
Sub SaveDocumentAs()
'VBA237
Application.Documents(3).SaveAs ("CopyOfPicture1")
End Sub
```

Siehe auch

Bilder mit VBA bearbeiten (Seite 1656)
GridHeight-Eigenschaft (Seite 2210)
Documents-Objekt (Auflistung) (Seite 1923)
SetRTLayerVisible-Methode (Seite 1873)
SaveAs-Methode (Seite 1865)
Save-Methode (Seite 1864)
PrintProjectDocumentation-Methode (Seite 1856)
PasteClipboard-Methode (Seite 1855)
MoveSelection-Methode (Seite 1852)
IsRTLayerVisible-Methode (Seite 1844)
Export-Methode (Seite 1826)
CopySelection-Methode (Seite 1811)
Close-Methode (Seite 1808)
VBA-Referenz (Seite 1735)
ExtendedZoomingEnable-Eigenschaft (Seite 2175)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Views-Eigenschaft (Seite 2483)
UpdateCycle-Eigenschaft (Seite 2404)
TabOrderOtherAction-Eigenschaft (Seite 2380)
TabOrderMouse-Eigenschaft (Seite 2379)
TabOrderKeyboard-Eigenschaft (Seite 2378)
TabOrderAllHMIObjects-Eigenschaft (Seite 2377)
SnapToGrid-Eigenschaft (Seite 2370)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
Selection-Eigenschaft (Seite 2363)
Properties-Eigenschaft (Seite 2341)
Path-Eigenschaft (Seite 2320)
PasswordLevel-Eigenschaft (Seite 2319)
Parent-Eigenschaft (Seite 2317)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
LockedByCreatorID-Eigenschaft (Seite 2275)
LastChange-Eigenschaft (Seite 2234)
HMIObjects-Eigenschaft (Seite 2215)
Hide-Eigenschaft (Seite 2214)
Height-Eigenschaft (Seite 2213)
GridWidth-Eigenschaft (Seite 2211)
GridColor-Eigenschaft (Seite 2210)
Grid-Eigenschaft (Seite 2209)

Documents-Objekt (Auflistung)

Beschreibung



VBA-Objektbezeichnung

HMIDocuments

Verwendung

Hinweis

Verwenden Sie die Eigenschaft "ActiveDocument", wenn Sie auf das aktive Bild verweisen wollen.

Verwenden Sie die Documents-Eigenschaft, um die Documents-Auflistung zurückzugeben. Im folgenden Beispiel werden die Namen aller geöffneten Bilder ausgegeben:

```
Sub ShowDocuments()
'VBA238
Dim colDocuments As Documents
Dim objDocument As Document
Set colDocuments = Application.Documents
For Each objDocument In colDocuments
MsgBox objDocument.Name
Next objDocument
End Sub
```

Verwenden Sie die Add-Methode, um ein neues Document-Objekt zur Documents-Auflistung hinzuzufügen. Im folgenden Beispiel wird ein neues Bild angelegt:

```
Sub AddNewDocument()
'VBA239
Dim objDocument As Document
Set objDocument = Application.Documents.Add
End Sub
```

Siehe auch

Add-Methode (Seite 1776)

Document-Objekt (Seite 1920)

SaveAll-Methode (Seite 1865)

Open-Methode (Seite 1853)

Item-Methode (Seite 1845)

CloseAll-Methode (Seite 1809)

Close-Methode (Seite 1808)

VBA-Referenz (Seite 1735)

Bilder mit VBA bearbeiten (Seite 1656)

Parent-Eigenschaft (Seite 2317)

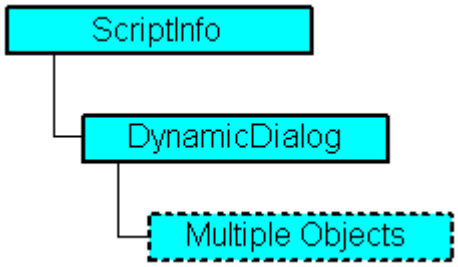
Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

ActiveDocument-Eigenschaft (Seite 2067)

DynamicDialog-Objekt

Beschreibung



Stellt den Dynamik-Dialog dar. Mit dem Dynamik-Dialog können Sie Eigenschaften von Bildern und Objekten in Abhängigkeit von verschiedenen Wertebereichen dynamisieren.

Den Wertebereich legen Sie mit der ResultType-Eigenschaft fest.

VBA-Objektbezeichnung

HMIDynamicDialog

Verwendung

Verwenden Sie das DynamicDialog-Objekt, um eine Objekteigenschaft mit dem Dynamik-Dialog zu dynamisieren. Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert, ein Variablenname vergeben und drei analoge Wertebereiche angelegt:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA240
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.Trigger.VariableTriggers.Add "NewDynamic2", hmiVariableCycleType_5s
.AnalogResultInfos.Add 50, 40
.AnalogResultInfos.Add 100, 80
.AnalogResultInfos.ElseCase = 100
End With
End Sub
```

Siehe auch

Delete-Methode (Seite 1818)

ConvertToScript-Methode (Seite 1810)

CheckSyntax-Methode (Seite 1807)

VariableStateValues-Eigenschaft (Seite 2479)

VariableStateChecked-Eigenschaft (Seite 2478)

Trigger-Eigenschaft (Seite 2391)

SourceCode-Eigenschaft (Seite 2372)

ScriptType-Eigenschaft (Seite 2357)

ResultType-Eigenschaft (Seite 2349)

Parent-Eigenschaft (Seite 2317)

Compiled-Eigenschaft (Seite 2152)

BitResultInfo-Eigenschaft (Seite 2104)

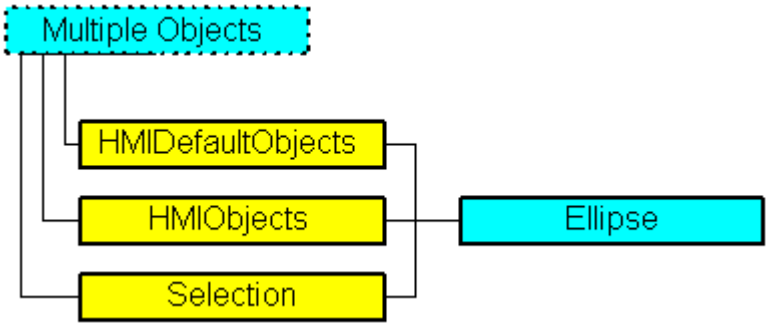
BinaryResultInfo-Eigenschaft (Seite 2103)

Application-Eigenschaft (Seite 2079)

AnalogResultInfos-Eigenschaft (Seite 2077)

Ellipse-Objekt

Beschreibung



Stellt das Objekt "Ellipse" dar. Das Ellipse-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIEllipse

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Ellipse" im Bild anzulegen:

```
Sub AddEllipse()
'VBA241
Dim objEllipse As HMIEllipse
Set objEllipse = ActiveDocument.HMIObjects.AddHMIObject("Ellipse", "HMIEllipse")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditEllipse()
'VBA242
Dim objEllipse As HMIEllipse
Set objEllipse = ActiveDocument.HMIObjects("Ellipse")
objEllipse.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

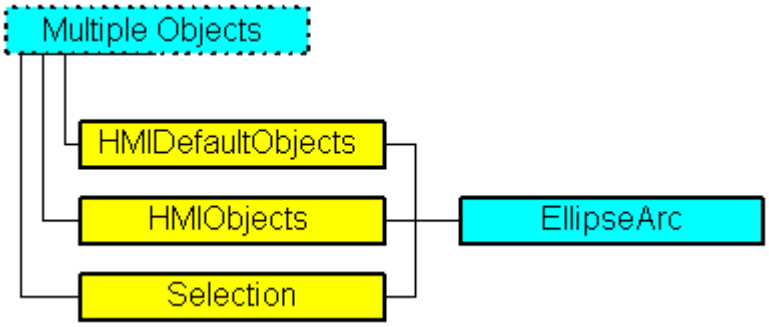
```
Sub ShowNameOfFirstSelectedObject()
'VBA243
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

FillingIndex-Eigenschaft (Seite 2179)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
RadiusWidth-Eigenschaft (Seite 2346)
RadiusHeight-Eigenschaft (Seite 2345)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashRateBackColor-Eigenschaft (Seite 2193)
FlashBorderColor-Eigenschaft (Seite 2186)
FlashBackColor-Eigenschaft (Seite 2185)
FillStyle-Eigenschaft (Seite 2181)
Filling-Eigenschaft (Seite 2178)
FillColor-Eigenschaft (Seite 2177)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
BorderColor-Eigenschaft (Seite 2108)
BorderBackColor-Eigenschaft (Seite 2107)
BackFlashColorOn-Eigenschaft (Seite 2094)
BackFlashColorOff-Eigenschaft (Seite 2093)
BackColor-Eigenschaft (Seite 2088)

EllipseArc-Objekt

Beschreibung



Stellt das Objekt "Ellipsenbogen" dar. Das EllipseArc-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIEllipseArc

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Ellipsenbogen" im Bild anzulegen:

```
Sub AddEllipseArc()
'VBA244
Dim objEllipseArc As HMIEllipseArc
Set objEllipseArc = ActiveDocument.HMIObjects.AddHMIObject("EllipseArc", "HMIEllipseArc")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditEllipseArc()
'VBA245
Dim objEllipseArc As HMIEllipseArc
Set objEllipseArc = ActiveDocument.HMIObjects("EllipseArc")
objEllipseArc.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

```
Sub ShowNameOfFirstSelectedObject()
'VBA246
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Verwenden Sie "HMIDefaultObjects(Index)", um ein Objekt aus der HMIDefaultObjects-Auflistung zurückzugeben:

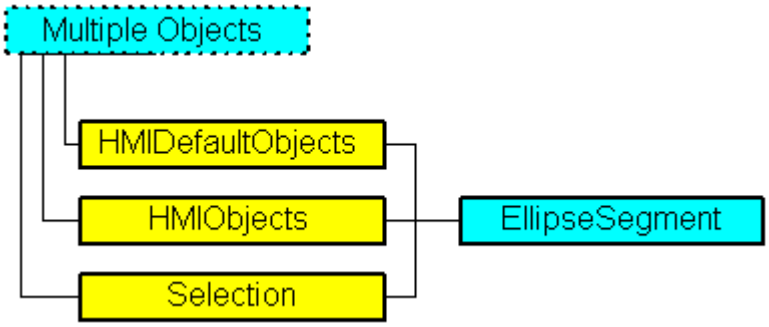
```
Sub EditDefaultPropertiesOfEllipseArc()
'VBA247
Dim objEllipseArc As HMIEllipseArc
Set objEllipseArc = Application.DefaultHMIObjects("HMIEllipseArc")
objEllipseArc.BorderColor = RGB(255, 255, 0)
'create new "EllipseArc"-object
Set objEllipseArc = ActiveDocument.HMIObjects.AddHMIObject("EllipseArc2", "HMIEllipseArc")
End Sub
```

Siehe auch

TooltipText-Eigenschaft (Seite 2387)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
StartAngle-Eigenschaft (Seite 2373)
RadiusWidth-Eigenschaft (Seite 2346)
RadiusHeight-Eigenschaft (Seite 2345)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashBorderColor-Eigenschaft (Seite 2186)
EndAngle-Eigenschaft (Seite 2171)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
BorderColor-Eigenschaft (Seite 2108)
BorderBackColor-Eigenschaft (Seite 2107)

EllipseSegment-Objekt

Beschreibung



Stellt das Objekt "Ellipsensegment" dar. Das EllipseSegment-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIEllipseSegment

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Ellipsensegment" im Bild anzulegen:

```
Sub AddEllipseSegment()
'VBA248
Dim objEllipseSegment As HMIEllipseSegment
Set objEllipseSegment = ActiveDocument.HMIObjects.AddHMIObject("EllipseSegment",
"HMIEllipseSegment")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditEllipseSegment()
'VBA249
Dim objEllipseSegment As HMIEllipseSegment
Set objEllipseSegment = ActiveDocument.HMIObjects("EllipseSegment")
objEllipseSegment.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

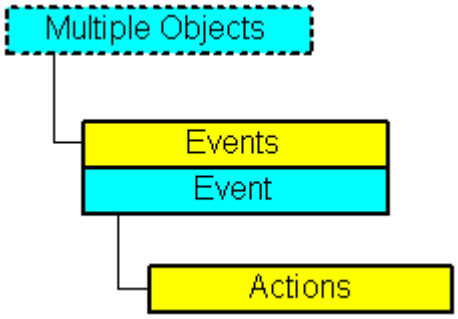
```
Sub ShowNameOfFirstSelectedObject()
'VBA250
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

TooltipText-Eigenschaft (Seite 2387)
BackFlashColorOn-Eigenschaft (Seite 2094)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
StartAngle-Eigenschaft (Seite 2373)
RadiusWidth-Eigenschaft (Seite 2346)
RadiusHeight-Eigenschaft (Seite 2345)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashRateBackColor-Eigenschaft (Seite 2193)
FlashBorderColor-Eigenschaft (Seite 2186)
FlashBackColor-Eigenschaft (Seite 2185)
FillStyle-Eigenschaft (Seite 2181)
FillingIndex-Eigenschaft (Seite 2179)
Filling-Eigenschaft (Seite 2178)
FillColor-Eigenschaft (Seite 2177)
EndAngle-Eigenschaft (Seite 2171)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
BorderColor-Eigenschaft (Seite 2108)
BorderBackColor-Eigenschaft (Seite 2107)
BackFlashColorOff-Eigenschaft (Seite 2093)
BackColor-Eigenschaft (Seite 2088)

Event-Objekt

Beschreibung



Stellt das Ereignis dar, das in Runtime eine oder mehrere Aktionen (z.B. Direktverbindung) auslöst. Ein Ereignis kann an ein Objekt und eine Eigenschaft projekiert werden.

VBA-Objektbezeichnung

HMIEvent

Verwendung

Verwenden Sie die AddAction-Methode, um eine Aktion an ein Ereignis zu projektieren. In diesem Beispiel soll bei Radiusänderung in Runtime eine C-Aktion ausgelöst werden:

```
Sub AddActionToPropertyTypeCScript()
'VBA251
Dim objEvent As HMIEvent
Dim objCScript As HMIScriptInfo
Dim objCircle As HMICircle
'Create circle in the picture. If property "Radius" is changed,
'a C-action is added:
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_AB", "HMICircle")
Set objEvent = objCircle.Radius.Events(1)
Set objCScript = objEvent.Actions.AddAction(hmiActionCreationTypeCScript)
End Sub
```

Siehe auch

Application-Eigenschaft (Seite 2079)

Delete-Methode (Seite 1818)

AddAction-Methode (Seite 1784)

VBA-Referenz (Seite 1735)

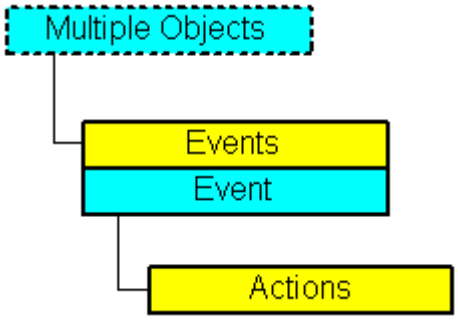
Parent-Eigenschaft (Seite 2317)

EventType-Eigenschaft (Seite 2173)

Actions-Eigenschaft (Seite 2066)

Events-Objekt (Auflistung)

Beschreibung



Eine Auflistung von Event-Objekten, die alle Ereignisse darstellen, die an ein Objekt projektiert sind. Welches Ereignis projektiert werden soll, legen Sie mit der Item-Methode fest:

- Sie projektieren mit VBA eine Aktion an eine Eigenschaft, indem Sie die Eigenschaft "Events(1)" verwenden, wobei der Index "1" für das Ereignis "bei Änderung" steht.
- Sie projektieren mit VBA eine Aktion an ein Objekt, indem Sie die Eigenschaft "Events(Index)" verwenden, wobei "Index" für das auslösende Ereignis steht (siehe Tabelle):

Index	EventType (abhängig vom verwendeten Objekt)
0	hmiEventTypeNotDefined
1	hmiEventTypeMouseClick
2	hmiEventTypeMouseLButtonDown
3	hmiEventTypeMouseLButtonUp
4	hmiEventTypeMouseRButtonDown
5	hmiEventTypeMouseRButtonUp
6	hmiEventTypeKeyboardDown
7	hmiEventTypeKeyboardUp
8	hmiEventTypeFocusEnter

Index	EventType (abhängig vom verwendeten Objekt)
9	hmiEventTypeObjectChange
10	hmiEventTypePictureOpen

VBA-Objektbezeichnung

HMIEvents

Verwendung

Verwenden Sie die Item-Methode, um ein einzelnes Event-Objekt zurückzugeben. In diesem Beispiel werden die Ereignisnamen und -typen aller Objekte im aktiven Bild ausgegeben. Damit dieses Beispiel funktioniert, fügen Sie in das aktive Bild einige Objekte ein und projektieren Sie verschiedene Ereignisse.

```
Sub ShowEventsOfAllObjectsInActiveDocument()
'VBA252
Dim colEvents As HMIEvents
Dim objEvent As HMIEvent
Dim iMax As Integer
Dim iIndex As Integer
Dim iAnswer As Integer
Dim strEventName As String
Dim strObjectName As String
Dim varEventType As Variant
iIndex = 1
iMax = ActiveDocument.HMIObjects.Count
For iIndex = 1 To iMax
Set colEvents = ActiveDocument.HMIObjects(iIndex).Events
strObjectName = ActiveDocument.HMIObjects(iIndex).ObjectName
For Each objEvent In colEvents
strEventName = objEvent.EventName
varEventType = objEvent.EventType
iAnswer = MsgBox("Objectname: " & strObjectName & vbCrLf & "Eventtype: " & varEventType &
vbCrLf & "Eventname: " & strEventName, vbOKCancel)
If vbCancel = iAnswer Then Exit For
Next objEvent
If vbCancel = iAnswer Then Exit For
Next iIndex
End Sub
```

Siehe auch

Item-Methode (Seite 1845)

VBA-Referenz (Seite 1735)

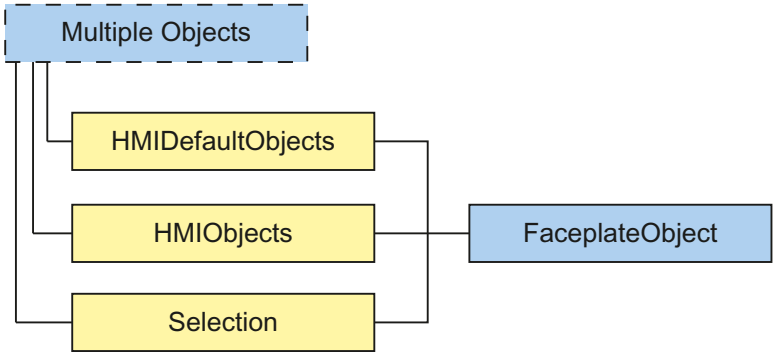
Parent-Eigenschaft (Seite 2317)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

FaceplateObject-Objekt

Beschreibung



Stellt das Objekt "Faceplate-Instanz" dar. Das FaceplateObject-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Smart-, Windows- und Rohr-Objekten.

VBA-Objektbezeichnung

HMIFaceplateObject

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Faceplate-Instanz" im Bild anzulegen:

```
Sub AddFaceplateInstance()
'VBA826
Dim objFaceplateInstance As HMIFaceplateObject
Set objFaceplateInstance = ActiveDocument.HMIObjects.AddHMIObject("Faceplate-Instanz",
"HMIFaceplateObject")
objFaceplateInstance.Properties.Item(3).value = "Faceplate1.fpt"
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditFaceplateInstance()
'VBA827
Dim objFaceplateInstance As HMIFaceplateObject
Set objFaceplateInstance = ActiveDocument.HMIObjects("Faceplate-Instanz")
objFaceplateInstance.visible = True
```

```
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

```
Sub ShowNameOfFirstSelectedObject()
'VBA828
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

ObjectName-Eigenschaft (Seite 2307)

Layer-Eigenschaft (Seite 2234)

Left-Eigenschaft (Seite 2267)

Top-Eigenschaft (Seite 2388)

Width-Eigenschaft (Seite 2486)

Height-Eigenschaft (Seite 2213)

Operation-Eigenschaft (Seite 2312)

PasswordLevel-Eigenschaft (Seite 2319)

Visible-Eigenschaft (Seite 2484)

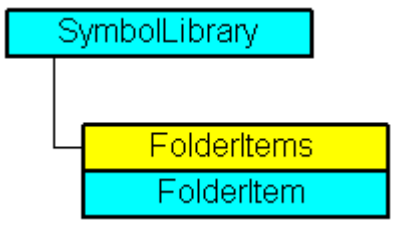
TooltipText-Eigenschaft (Seite 2387)

ScalingMode-Eigenschaft (Seite 2356)

FaceplateType-Eigenschaft (Seite 2176)

FolderItem-Objekt

Beschreibung



Repräsentiert einen Ordner oder ein Objekt in der Bausteinbibliothek. Das FolderItem-Objekt vom Typ "Folder" ist Element der FolderItems-Auflistung. Das FolderItem-Objekt vom Typ "Item" ist Element der Folder-Auflistung.

VBA-Objektbezeichnung

HMIFolderItem

Verwendung

Verwenden Sie die FolderItems-Eigenschaft, um die FolderItems-Auflistung zurückzugeben. Im folgenden Beispiel werden die Ordnernamen der "Globalen Bibliothek" ausgegeben:

```
Sub ShowFolderItemsOfGlobalLibrary()
'VBA253
Dim colFolderItems As HMIFolderItems
Dim objFolderItem As HMIFolderItem
Set colFolderItems = Application.SymbolLibraries(1).FolderItems
For Each objFolderItem In colFolderItems
MsgBox objFolderItem.Name
Next objFolderItem
End Sub
```

Verwenden Sie die CopyToClipboard-Methode, um ein Objekt "FolderItem" vom Typ "Item" in die Zwischenablage zu kopieren. Im folgenden Beispiel wird das Objekt "PC" in die Zwischenablage kopiert:

```
Sub CopyFolderItemToClipboard()
'VBA254
Dim objGlobalLib As HMISymbolLibrary
Set objGlobalLib = Application.SymbolLibraries(1)
objGlobalLib.FolderItems("Folder2").Folder("Folder2").Folder.Item("Object1").CopyToClipboa
rd
End Sub
```

Siehe auch

Type-Eigenschaft (Seite 2392)

FolderItems-Objekt (Auflistung) (Seite 1941)

Delete-Methode (Seite 1818)

CopyToClipboard-Methode (Seite 1812)

So fügen Sie ein Objekt aus der Bausteinbibliothek mit VBA in ein Bild ein (Seite 1654)

So bearbeiten Sie die Bausteinbibliothek mit VBA (Seite 1651)

VBA-Referenz (Seite 1735)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

Parent-Eigenschaft (Seite 2317)

Name-Eigenschaft (Seite 2303)

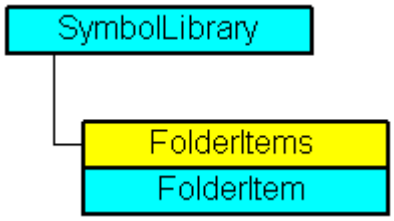
LDNames-Eigenschaft (Seite 2262)

Folder-Eigenschaft (Seite 2198)

Application-Eigenschaft (Seite 2079)

FolderItems-Objekt (Auflistung)

Beschreibung



Eine Auflistung von FolderItem-Objekten, die alle Ordner und Objekte in der Bausteinbibliothek darstellen.

VBA-Objektbezeichnung

HMIFolderItems

Verwendung

Verwenden Sie die FolderItems-Eigenschaft, um die FolderItems-Auflistung zurückzugeben. Im folgenden Beispiel werden die Ordnernamen der "Globalen Bibliothek" ausgegeben:

```
Sub ShowFolderItemsOfGlobalLibrary()
'VBA255
Dim colFolderItems As HMIFolderItems
```

```
Dim objFolderItem As HMIFolderItem
Set colFolderItems = Application.SymbolLibraries(1).FolderItems
For Each objFolderItem In colFolderItems
MsgBox objFolderItem.Name
Next objFolderItem
End Sub
```

Verwenden Sie die z.B. die AddFolder-Methode, um einen neuen Ordner in der Bausteinbibliothek anzulegen. Im folgenden Beispiel wird in der "Projekt Bibliothek" der Ordner "Projektordner" angelegt:

```
Sub AddNewFolderToProjectLibrary()
'VBA256
Dim objProjectLib As HMISymbolLibrary
Set objProjectLib = Application.SymbolLibraries(2)
objProjectLib.FolderItems.AddFolder ("My Folder")
End Sub
```

Siehe auch

AddItem-Methode (Seite 1793)

SymbolLibrary-Objekt (Seite 2035)

FolderItem-Objekt (Seite 1939)

Item-Methode (Seite 1845)

AddFromClipboard-Methode (Seite 1790)

AddFolder-Methode (Seite 1789)

So fügen Sie ein Objekt aus der Bausteinbibliothek mit VBA in ein Bild ein (Seite 1654)

So bearbeiten Sie die Bausteinbibliothek mit VBA (Seite 1651)

VBA-Referenz (Seite 1735)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

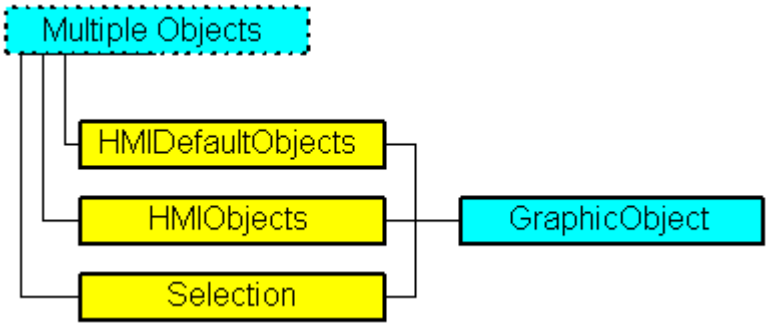
Parent-Eigenschaft (Seite 2317)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

GraphicObject-Objekt

Beschreibung



Stellt das Objekt "Grafik-Objekt" dar. Das GraphicObject-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIGraphicObject

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Grafik-Objekt" im Bild anzulegen:

```
Sub AddGraphicObject()
'VBA257
Dim objGraphicObject As HMIGraphicObject
Set objGraphicObject = ActiveDocument.HMIObjects.AddHMIObject("Graphic-Object",
"HMIGraphicObject")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditGraphicObject()
'VBA258
Dim objGraphicObject As HMIGraphicObject
Set objGraphicObject = ActiveDocument.HMIObjects("Graphic-Object")
objGraphicObject.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

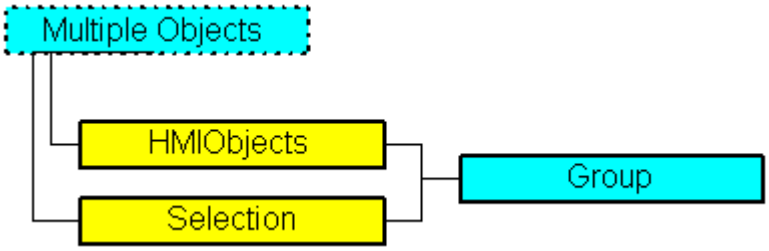
```
Sub ShowNameOfFirstSelectedObject()
'VBA259
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name of the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

Left-Eigenschaft (Seite 2267)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
PicUseTransColor-Eigenschaft (Seite 2332)
PictureName-Eigenschaft (Seite 2329)
PicTransColor-Eigenschaft (Seite 2327)
PicReferenced-Eigenschaft (Seite 2326)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashRateBackColor-Eigenschaft (Seite 2193)
FlashBorderColor-Eigenschaft (Seite 2186)
FlashBackColor-Eigenschaft (Seite 2185)
FillStyle-Eigenschaft (Seite 2181)
FillingIndex-Eigenschaft (Seite 2179)
Filling-Eigenschaft (Seite 2178)
FillColor-Eigenschaft (Seite 2177)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
BorderColor-Eigenschaft (Seite 2108)
BorderBackColor-Eigenschaft (Seite 2107)
BackFlashColorOn-Eigenschaft (Seite 2094)
BackFlashColorOff-Eigenschaft (Seite 2093)
BackColor-Eigenschaft (Seite 2088)

Group-Objekt

Beschreibung



Stellt das Objekt "Gruppen-Objekt" dar. Das Group-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.

Das Gruppen-Objekt wird aus ausgewählten Objekten im Bild erzeugt. Die Objekte im Group-Objekt werden zusätzlich in der Auflistung "GroupedHMIObjects" gespeichert, wobei die Indexnummern neu vergeben werden.

Sie haben auf die Objekteigenschaften aller Objekte im Group-Objekt uneingeschränkten Zugriff.

Weitere Informationen zum Gruppen-Objekt finden Sie in der WinCC-Dokumentation unter "Gruppen-Objekt".

VBA-Objektbezeichnung

HMIGroup

Verwendung

Verwenden Sie die CreateGroup-Methode mit der Selection-Auflistung, um ein neues Objekt "Gruppen-Objekt" im Bild anzulegen:

```
Sub DoCreateGroup()
'VBA260
Dim objGroup As HMIGroup
Set objGroup = ActiveDocument.Selection.CreateGroup
objGroup.ObjectName = "Group-Object"
End Sub
```

Verwenden Sie folgende Methoden, um ein vorhandenes Group-Objekt zu bearbeiten:

- Methode "Add(Index)": Fügt ein neues Objekt zum Gruppen-Objekt hinzu.
- Methode "Remove(Index)": Entfernt ein Objekt aus dem Gruppen-Objekt.
- Methode "UnGroup()": Löst das Gruppen-Objekt auf (Gruppierung aufheben).
- Methode "Delete()": Löscht das Gruppen-Objekt und die darin enthaltenen Objekte.

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditGroup()
'VBA261
Dim objGroup As HMIGroup
Set objGroup = ActiveDocument.HMIObjects("Group-Object")
MsgBox objGroup.ObjectName
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

HMIObjects-Objekt (Auflistung) (Seite 1957)

GroupedObjects-Objekt (Auflistung) (Seite 1950)

Ungroup-Methode (Seite 1877)

Remove-Methode (Seite 1857)

Delete-Methode (Seite 1818)

Add-Methode (GroupedObjects-Auflistung) (Seite 1781)

So bearbeiten Objekte im Gruppen-Objekt mit VBA (Seite 1684)

So bearbeiten Sie Gruppen-Objekte mit VBA (Seite 1681)

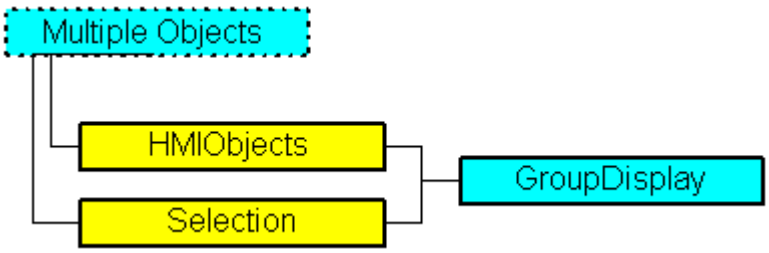
VBA-Referenz (Seite 1735)

Gruppen-Objekte (Seite 1679)

Objekte mit VBA bearbeiten (Seite 1662)

GroupDisplay-Objekt

Beschreibung



Stellt das Objekt "Sammelanzeige" dar. Das GroupDisplay-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.

VBA-Objektbezeichnung

HMIGroupDisplay

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Sammelanzeige" im Bild anzulegen:

```
Sub AddGroupDisplay()
'VBA262
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("Groupdisplay",
"HMIGroupDisplay")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditGroupDisplay()
'VBA263
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects("Groupdisplay")
objGroupDisplay.BackColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

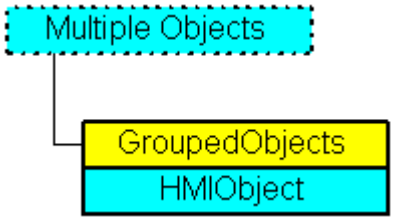
```
Sub ShowNameOfFirstSelectedObject()
'VBA264
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

MCText-Eigenschaft (Seite 2297)
Height-Eigenschaft (Seite 2213)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
UserValue4-Eigenschaft (Seite 2408)
UserValue3-Eigenschaft (Seite 2407)
UserValue2-Eigenschaft (Seite 2406)
UserValue1-Eigenschaft (Seite 2405)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
SignificantMask-Eigenschaft (Seite 2367)
SameSize-Eigenschaft (Seite 2353)
Relevant-Eigenschaft (Seite 2349)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
MessageClass-Eigenschaft (Seite 2300)
MCKQTextFlash-Eigenschaft (Seite 2296)
MCKQTextColorOn-Eigenschaft (Seite 2296)
MCKQTextColorOff-Eigenschaft (Seite 2295)
MCKQBackFlash-Eigenschaft (Seite 2294)
MCKQBackColorOn-Eigenschaft (Seite 2294)
MCKQBackColorOff-Eigenschaft (Seite 2293)
MCKOTextFlash-Eigenschaft (Seite 2292)
MCKOTextColorOn-Eigenschaft (Seite 2292)
MCKOTextColorOff-Eigenschaft (Seite 2291)
MCKOBackFlash-Eigenschaft (Seite 2290)
MCKOBackColorOn-Eigenschaft (Seite 2290)
MCKOBackColorOff-Eigenschaft (Seite 2289)
MCGUTextFlash-Eigenschaft (Seite 2288)
MCGUTextColorOn-Eigenschaft (Seite 2288)
MCGUTextColorOff-Eigenschaft (Seite 2287)
MCGUBackFlash-Eigenschaft (Seite 2286)
MCGUBackColorOn-Eigenschaft (Seite 2286)
MCGUBackColorOff-Eigenschaft (Seite 2285)

GroupedObjects-Objekt (Auflistung)

Beschreibung



Eine Auflistung von HMIObject-Objekten, die alle Objekte im Gruppen-Objekt darstellen.

VBA-Objektbezeichnung

HMIGroupedObjects

Verwendung

Verwenden Sie die GroupedHMIObjects-Eigenschaft, um die GroupedObjects-Auflistung zurückzugeben. Im folgenden Beispiel werden alle Objekte des ersten Gruppen-Objektes im aktuellen Bild ausgegeben. Das Gruppen-Objekt mit dem Namen "Gruppe1" müssen Sie zuvor angelegt haben:

```
Sub ShowGroupedObjectsOfFirstGroup()
'VBA265
Dim colGroupedObjects As HMIGroupedObjects
Dim objObject As HMIObject
Set colGroupedObjects = ActiveDocument.HMIObjects("Group1").GroupedHMIObjects
For Each objObject In colGroupedObjects
MsgBox objObject.ObjectName
Next objObject
End Sub
```

Verwenden Sie z.B. die Remove-Methode, um ein Objekt aus dem Gruppen-Objekt zu entfernen. Im folgenden Beispiel wird das erste aus Objekt aus dem Gruppen-Objekt "Gruppe1" entfernt:

```
Sub RemoveObjectFromGroup()
'VBA266
Dim objGroup As HMIGroup
Set objGroup = ActiveDocument.HMIObjects("Group1")
objGroup.GroupedHMIObjects.Remove (1)
End Sub
```

Siehe auch

Item-Methode (Seite 1845)

Group-Objekt (Seite 1946)

Remove-Methode (Seite 1857)

Add-Methode (GroupedObjects-Auflistung) (Seite 1781)

So bearbeiten Sie Gruppen-Objekte mit VBA (Seite 1681)

VBA-Referenz (Seite 1735)

Gruppen-Objekte (Seite 1679)

Parent-Eigenschaft (Seite 2317)

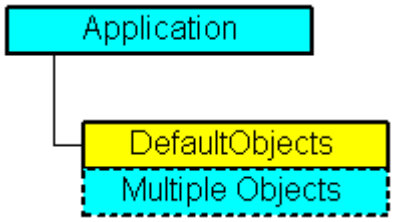
GroupedHMIObjects-Eigenschaft (Seite 2212)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

HMIDefaultObjects-Objekt (Auflistung)

Beschreibung



Eine Auflistung von folgenden HMIObject-Objekten:

Objekt	VBA-Objektbezeichnung
Linie	HMILine
Polygon	HMIPolygon
Polygonzug	HMIPolyLine
Ellipse	HMIEllipse
Kreis	HMICircle
Ellipsensegment	HMIEllipseSegment
Kreissegment	HMIPieSegment
Ellipsenbogen	HMIEllipseArc
Kreisbogen	HMICircularArc
Rechteck	HMIRectangle
Rundrechteck	HMIRoundRectangle
Applikationsfenster	HMIApplicationWindow
Bildfenster	HMIPictureWindow
Statischer Text	HMIStaticText

Objekt	VBA-Objektbezeichnung
EA-Feld	HMIIOField
Button	HMIButton
Check-Box	HMICheckBox
Radio-Box	HMIOptionGroup
Rundbutton	HMIRoundButton
Balken	HMIBarGraph
Slider-Objekt	HMISlider
Grafik-Objekt	HMIGraphicObject
Zustandsanzeige	HMIStatusDisplay
Textliste	HMITextList
Verbinder	HMIObjConnection
Mehrzeiliger Text	HMIMultiLineEdit
Kombinationsfeld	HMIComboBox
Listenfeld	HMIListBox
Polygonrohr	HMITubePolyline
T-Stück	HMITubeTeeObject
Doppel-T-Stück	HMITubeDoubleTeeObject
Rohrbogen	HMITubeArcObject
3D-Balken	HMI3DBarGraph
Sammelanzeige	HMIGroupDisplay
Faceplate-Instanz	HMIFaceplateObject

VBA-Objektbezeichnung

HMIDefaultObjects

Verwendung

Verwenden Sie die DefaultHMIObjects-Eigenschaft, um die Vorbelegungen der Eigenschaftswerte der enthaltenen Objekte zu ändern. In diesem Beispiel werden alle in der Auflistung enthaltenen Objekte ausgegeben:

```
Sub ShowDefaultObjects()
'VBA267
Dim strType As String
Dim strName As String
Dim strMessage As String
Dim iMax As Integer
Dim iIndex As Integer
iMax = Application.DefaultHMIObjects.Count
iIndex = 1
For iIndex = 1 To iMax
With Application.DefaultHMIObjects(iIndex)
strType = .Type
strName = .ObjectName
```

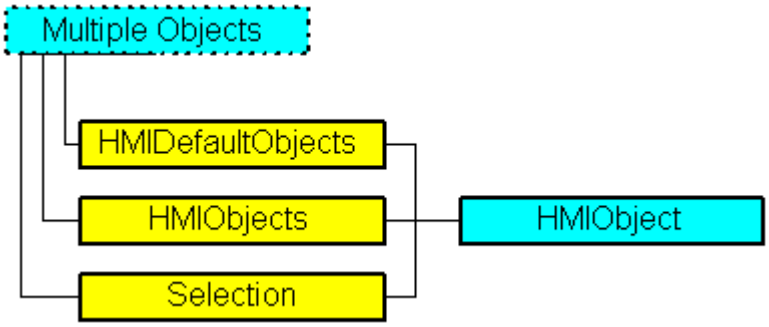
```
strMessage = strMessage & "Element: " & iIndex & " / Objecttype: " & strType & " /
Objectname: " & strName
End With
If 0 = iIndex Mod 10 Then
MsgBox strMessage
strMessage = ""
Else
strMessage = strMessage & vbCrLf & vbCrLf
End If
Next iIndex
MsgBox "Element: " & iIndex & vbCrLf & "Objecttype: " & strType & vbCrLf & "Objectname: "
& strName
End Sub
```

Siehe auch

Button-Objekt (Seite 1898)
TextList-Objekt (Seite 2037)
StatusDisplay-Objekt (Seite 2032)
StaticText-Objekt (Seite 2029)
Slider-Objekt (Seite 2025)
RoundRectangle-Objekt (Seite 2018)
RoundButton-Objekt (Seite 2015)
Rectangle-Objekt (Seite 2012)
PolyLine-Objekt (Seite 2001)
Polygon-Objekt (Seite 1998)
PieSegment-Objekt (Seite 1995)
PictureWindow-Objekt (Seite 1992)
OptionGroup-Objekt (Seite 1989)
Line-Objekt (Seite 1970)
IOField-Objekt (Seite 1959)
EllipseSegment-Objekt (Seite 1932)
EllipseArc-Objekt (Seite 1929)
Ellipse-Objekt (Seite 1926)
CircularArc-Objekt (Seite 1905)
Circle-Objekt (Seite 1902)
CheckBox-Objekt (Seite 1901)
BarGraph-Objekt (Seite 1893)
ApplicationWindow-Objekt (Seite 1891)
Item-Methode (Seite 1845)
VBA-Referenz (Seite 1735)
Parent-Eigenschaft (Seite 2317)
Count-Eigenschaft (Seite 2153)
DefaultHMIObjects-Eigenschaft (Seite 2161)
Application-Eigenschaft (Seite 2079)

HMIObject-Objekt

Beschreibung



Stellt ein Objekt aus der Objektpalette des Graphics Designer dar. Das HMIObject-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

Dieses Objekt enthält die Objekteigenschaften, die für alle Standard-, Smart- und Windows-Objekte gelten (u.a. Width, Height, Top und Left).

VBA-Objektbezeichnung

HMIObject

Verwendung

Verwenden Sie z.B. HMIObjects(Index), um ein einzelnes HMIObject-Objekt zurückzugeben. Als "Index" können Sie entweder die Indexnummer oder den Namen des Objektes verwenden. Im folgenden Beispiel wird der Objektname des ersten Objektes im aktiven Bild ausgegeben:

```
Sub ShowFirstObjectOfCollection()
'VBA268
Dim strName As String
strName = ActiveDocument.HMIObjects(1).ObjectName
MsgBox strName
End Sub
```

Verwenden Sie die Delete-Methode, um ein Objekt aus der HMIObjects-Auflistung zu entfernen. Im folgenden Beispiel wird das erste Objekt im aktiven Bild entfernt:

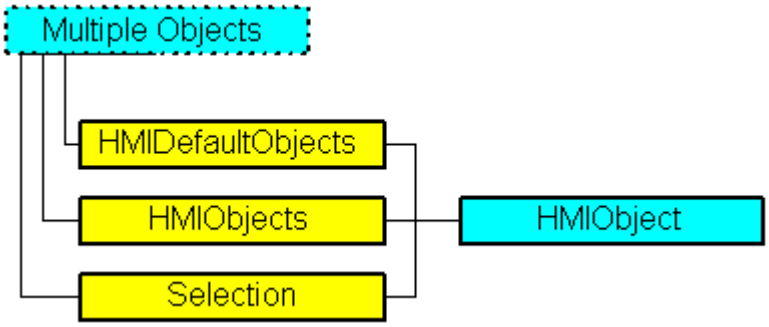
```
Sub DeleteObject()
'VBA269
ActiveDocument.HMIObjects(1).Delete
End Sub
```

Siehe auch

Name-Eigenschaft (Seite 2303)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
Delete-Methode (Seite 1818)
VBA-Referenz (Seite 1735)
Standard-, Smart-, Windows- und Rohr-Objekte (Seite 1665)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Type-Eigenschaft (Seite 2392)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
TabOrderAlpha-Eigenschaft (Seite 2378)
TabOrderSwitch-Eigenschaft (Seite 2376)
Selected-Eigenschaft (Seite 2362)
Properties-Eigenschaft (Seite 2341)
PasswordLevel-Eigenschaft (Seite 2319)
Parent-Eigenschaft (Seite 2317)
Operation-Eigenschaft (Seite 2312)
Left-Eigenschaft (Seite 2267)
LDTooltipTexts-Eigenschaft (Seite 2265)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
GroupParent-Eigenschaft (Seite 2212)
Events-Eigenschaft (Seite 2171)
Application-Eigenschaft (Seite 2079)

HMIObjects-Objekt (Auflistung)

Beschreibung



Eine Auflistung von HMIObject-Objekten, die alle Objekte im Bild darstellen.

VBA-Objektbezeichnung

HMIObjects

Hinweis

Die Reihenfolge der HMI-Objekte in der HMIObjects-Auflistung kann sich durch Hinzufügen und/oder Löschen von HMI-Objekten ändern.

Die Reihenfolge der Auflistung kann sich auch ändern, wenn HMI-Objekte in der aktuellen Auflistung bearbeitet werden. Das Verhalten kann auftreten, wenn die Eigenschaft "Layers" geändert wird und/oder wenn die Methoden "SendToBack" und "BringToFront" verwendet werden.

Verwendung

Verwenden Sie die HMIObjects-Eigenschaft, um die HMIObjects-Auflistung zurückzugeben. Im folgenden Beispiel werden alle Objektnamen des aktiven Bildes ausgegeben:

```
Sub ShowObjectsOfDocument()
'VBA270
Dim colObjects As HMIObjects
Dim objObject As HMIObject
Set colObjects = ActiveDocument.HMIObjects
For Each objObject In colObjects
MsgBox objObject.ObjectName
Next objObject
End Sub
```

Verwenden Sie z.B. die AddHMIObject-Methode, um ein neues Objekt im Bild anzulegen. Im folgenden Beispiel wird ein Kreis in das aktive Bild eingefügt:

```
Sub AddCircle()
'VBA271
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_1", "HMICircle")
End Sub
```

Verwenden Sie z.B. die Find-Methode, um nach einem oder mehreren Objekten im Bild zu suchen. Im folgenden Beispiel wird im aktiven Bild nach Objekten vom Typ "HMICircle" gesucht:

```
Sub FindObjectsByType()
'VBA272
Dim colSearchResults As HMICollection
Dim objMember As HMIObject
Dim iResult As Integer
Dim strName As String
Set colSearchResults = ActiveDocument.HMIObjects.Find(ObjectType:="HMICircle")
For Each objMember In colSearchResults
iResult = colSearchResults.Count
strName = objMember.ObjectName
MsgBox "Found: " & CStr(iResult) & vbCrLf & "Objectname: " & strName
Next objMember
End Sub
```

Siehe auch

Count-Eigenschaft (Seite 2153)

HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)

Selection-Objekt (Auflistung) (Seite 2022)

Item-Methode (Seite 1845)

Find-Methode (Seite 1827)

AddOLEObject-Methode (Seite 1794)

AddHMIObject-Methode (Seite 1791)

AddActiveXControl-Methode (Seite 1785)

So bearbeiten Sie Standard-, Smart- Windows- und Rohr-Objekte (Seite 1667)

VBA-Referenz (Seite 1735)

Standard-, Smart-, Windows- und Rohr-Objekte (Seite 1665)

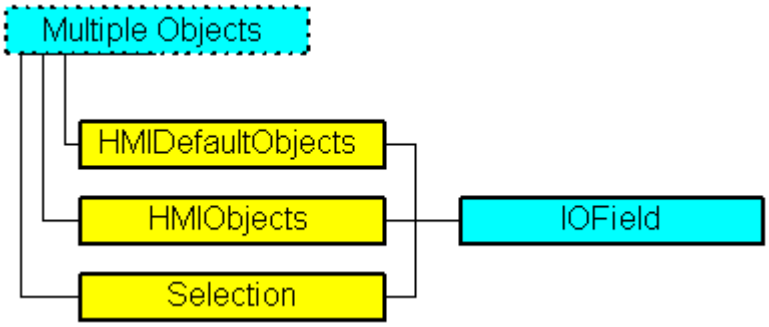
Objekte mit VBA bearbeiten (Seite 1662)

Parent-Eigenschaft (Seite 2317)

Application-Eigenschaft (Seite 2079)

IOField-Objekt

Beschreibung



Stellt das Objekt "EA-Feld" dar. Das IOField-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIIOField

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "EA-Feld" im Bild anzulegen:

```
Sub AddIOField()
'VBA273
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IO-Field", "HMIIOField")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditIOField()
'VBA274
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects("IO-Field")
objIOField.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

```
Sub ShowNameOfFirstSelectedObject()
'VBA275
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Verwenden Sie "HMIDefaultObjects(Index)", um ein Objekt aus der HMIDefaultObjects-Auflistung zurückzugeben:

```
Sub EditDefaultPropertiesOfIOField()
'VBA276
Dim objIOField As HMIIOField
Set objIOField = Application.DefaultHMIObjects("HMIIOField")
objIOField.BorderColor = RGB(255, 255, 0)
End Sub
```

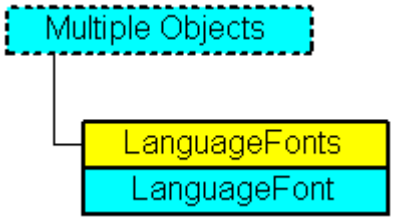
Siehe auch

LimitMin-Eigenschaft (Seite 2273)
ClearOnNew-Eigenschaft (Seite 2136)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
PasswordLevel-Eigenschaft (Seite 2319)
OutputValue-Eigenschaft (Seite 2316)
OutputFormat-Eigenschaft (Seite 2315)
Orientation-Eigenschaft (Seite 2315)
OperationReport-Eigenschaft (Seite 2314)
OperationMessage-Eigenschaft (Seite 2313)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
LimitMax-Eigenschaft (Seite 2272)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
HiddenInput-Eigenschaft (Seite 2215)
Height-Eigenschaft (Seite 2213)
ForeFlashColorOn-Eigenschaft (Seite 2207)
ForeFlashColorOff-Eigenschaft (Seite 2206)
ForeColor-Eigenschaft (Seite 2206)
FontUnderline-Eigenschaft (Seite 2205)
FontSize-Eigenschaft (Seite 2204)
FontName-Eigenschaft (Seite 2203)
FontItalic-Eigenschaft (Seite 2203)
FontBold-Eigenschaft (Seite 2200)
FlashRateForeColor-Eigenschaft (Seite 2197)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashRateBackColor-Eigenschaft (Seite 2193)
FlashForeColor-Eigenschaft (Seite 2188)
FlashBorderColor-Eigenschaft (Seite 2186)
FlashBackColor-Eigenschaft (Seite 2185)
FillStyle-Eigenschaft (Seite 2181)

L-Q

LanguageFont-Objekt

Beschreibung



Enthält die Schrifteinstellungen der Projektiersprache. Das LanguageFont-Objekt ist Element der LanguageFonts-Auflistung.

VBA-Objektbezeichnung

HMILanguageFont

Verwendung

Verwenden Sie die LDFonts(Index), um ein einzelnes LanguageFont-Objekt zurückzugeben. Im folgenden Beispiel wird ein Button-Objekt angelegt und der Name der ersten projektierten Schriftart ausgegeben:

```
Sub ShowFirstObjectOfCollection()
'VBA277
Dim strName As String
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button", "HMIButton")
strName = objButton.LDFonts(1).Family
MsgBox strName
End Sub
```

Objekteigenschaften

Das Objekt LanguageFont verfügt über folgende Eigenschaften:

Siehe auch

LanguageFonts-Objekt (Auflistung) (Seite 1963)

VBA-Referenz (Seite 1735)

Underlined-Eigenschaft (Seite 2402)

Size-Eigenschaft (Seite 2368)

Parent-Eigenschaft (Seite 2317)

LanguageID-Eigenschaft (Seite 2232)

Italic-Eigenschaft (Seite 2224)

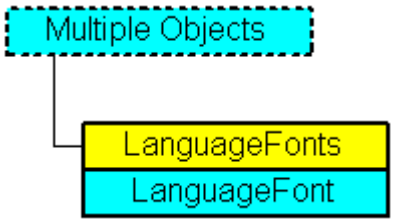
FontFamily-Eigenschaft (Seite 2201)

Bold-Eigenschaft (Seite 2106)

Application-Eigenschaft (Seite 2079)

LanguageFonts-Objekt (Auflistung)

Beschreibung



Eine Auflistung von LanguageFont-Objekten, die alle sprachabhängigen Schriften in einem Objekt darstellen.

VBA-Objektbezeichnung

HMILanguageFonts

Verwendung

Verwenden Sie die LDFonts-Eigenschaft, um die LanguageFonts-Auflistung zurückzugeben. Im folgenden Beispiel werden die Sprachkennungen der projektierten Schriftarten ausgegeben:

```
Sub ShowLanguageFont()
'VBA278
Dim colLanguageFonts As HMILanguageFonts
Dim objLanguageFont As HMILanguageFont
Dim objButton As HMIButton
Dim iMax As Integer
```

```
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
Set colLanguageFonts = objButton.LDFonts
iMax = colLanguageFonts.Count
For Each objLanguageFont In colLanguageFonts
MsgBox "Planned fonts: " & iMax & vbCrLf & "Language-ID: " & objLanguageFont.LanguageID
Next objLanguageFont
End Sub
```

Verwenden Sie die ItemByLcid-Methode, um die Sprache festzulegen, für welche die Schrifteinstellungen vorgenommen werden sollen. Im folgenden Beispiel werden für einen Button die Schriftattribute für Französisch und Englisch gesetzt.

Hinweis

Damit das Beispiel funktioniert, müssen Sie bereits in den jeweiligen Sprachen projektiert haben.

```
Sub ExampleForLanguageFonts()
'VBA279
Dim colLangFonts As HMILanguageFonts
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
objButton.Text = "DefText"
Set colLangFonts = objButton.LDFonts

'Adjust fontsettings for french:
With colLangFonts.ItemByLCID(1036)
.Family = "Courier New"
.Bold = True
.Italic = False
.Underlined = True
.Size = 12
End With
'Adjust fontsettings for english:
With colLangFonts.ItemByLCID(1033)
.Family = "Times New Roman"
.Bold = False
.Italic = True
.Underlined = False
.Size = 14
End With
End Sub
```

Siehe auch

LanguageFont-Objekt (Seite 1962)

ItemByLcid-Methode (Seite 1847)

Item-Methode (Seite 1845)

VBA-Referenz (Seite 1735)

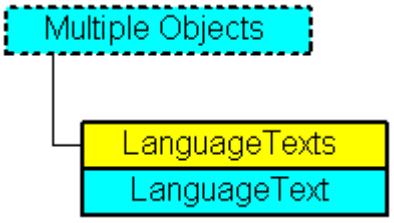
Parent-Eigenschaft (Seite 2317)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

LanguageText-Objekt

Beschreibung



Enthält die mehrsprachigen Beschriftungen eines Objektes. Das LanguageText-Objekt ist Element der LanguageTexts-Auflistung.

VBA-Objektbezeichnung

HMILanguageText

Verwendung

Im folgenden Beispiel wird dem Button "myButton" eine deutsche und englische Beschriftung zugewiesen:

```
Sub AddLanguagesToButton()
'VBA280
Dim objLabelText As HMILanguageText
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
'
'Add text in actual datalanguage:
objButton.Text = "Actual-Language Text"
'
'Add english text:
Set objLabelText = ActiveDocument.HMIObjects("myButton").LDTexts.Add(1033, "English Text")
End Sub
```

Siehe auch

LanguageTexts-Objekt (Auflistung) (Seite 1966)

Delete-Methode (Seite 1818)

VBA-Referenz (Seite 1735)

Parent-Eigenschaft (Seite 2317)

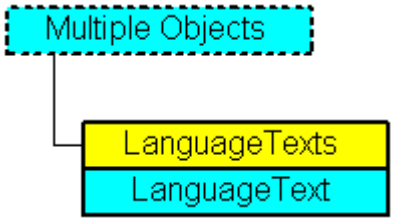
LanguageID-Eigenschaft (Seite 2232)

DisplayText-Eigenschaft (Seite 2166)

Application-Eigenschaft (Seite 2079)

LanguageTexts-Objekt (Auflistung)

Beschreibung



Eine Auflistung von LanguageText-Objekten, die alle mehrsprachigen Texte in einem Objekt darstellen.

VBA-Objektbezeichnung

HMILanguageTexts

Verwendung

Verwenden Sie eine der folgenden Eigenschaften, um die LanguageTexts-Auflistung zurückzugeben:

- LDLabelTexts-Eigenschaft
- LDNames-Eigenschaft
- LDStatusTexts-Eigenschaft
- LDTexts-Eigenschaft
- LDToolTipTexts-Eigenschaft

Ein Beispiel für die Verwendung der LanguageTexts-Auflistung finden Sie unter "LDStatusTexts-Eigenschaft" in dieser Dokumentation.

Verwenden Sie die Add-Methode, um einem Objekt fremdsprachige Texte hinzuzufügen. Im folgenden Beispiel wird dem Button "myButton" eine deutsche und englische Beschriftung zugewiesen:

```
Sub AddLanguagesToButton()
'VBA281
Dim objLabelText As HMILanguageText
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
'
'Add text in actual datalanguage:
objButton.Text = "Actual-Language Text"
'
'Add english text:
Set objLabelText = ActiveDocument.HMIObjects("myButton").LDTexts.Add(1033, "English Text")
End Sub
```

Siehe auch

LanguageText-Objekt (Seite 1965)

ItemByLcid-Methode (Seite 1847)

Item-Methode (Seite 1845)

VBA-Referenz (Seite 1735)

Parent-Eigenschaft (Seite 2317)

LDTooltipTexts-Eigenschaft (Seite 2265)

LDTexts-Eigenschaft (Seite 2265)

LDStatusTexts-Eigenschaft (Seite 2263)

LDNames-Eigenschaft (Seite 2262)

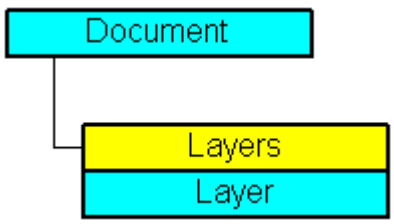
LDLabelTexts-Eigenschaft (Seite 2261)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

Layer-Objekt

Beschreibung



Stellt eine der 32 Ebenen dar, die im Bild verfügbar sind.

VBA-Objektbezeichnung

HMILayer

Verwendung

Verwenden Sie das Layer-Objekt, um für eine Ebene einen Namen und den Minimal- und Maximalzoom festzulegen. Die Sichtbarkeit von Ebenen legen Sie getrennt nach CS- und RT-Ebenen fest:

- Document-Objekt: Steuert die Sichtbarkeit der RT-Ebenen.
- View-Objekt: Steuert die Sichtbarkeit der CS-Ebenen.

Verwenden Sie die Layers-Auflistung, um ein Layer-Objekt zurückzugeben. Im folgenden Beispiel werden im aktiven Bild die Einstellungen der untersten Ebene konfiguriert:

```
Sub ConfigureSettingsOfLayer()
'VBA282
Dim objLayer As HMILayer
Set objLayer = ActiveDocument.Layers(1)
With objLayer
'configure "Layer 0"
.MinZoom = 10
.MaxZoom = 100
.Name = "Configured with VBA"
End With
End Sub
```

Siehe auch

Layers-Eigenschaft (Seite 2259)

VBA-Referenz (Seite 1735)

Ebenen mit VBA bearbeiten (Seite 1659)

Visible-Eigenschaft (Seite 2484)

Number-Eigenschaft (Seite 2305)

Name-Eigenschaft (Seite 2303)

MinZoom-Eigenschaft (Seite 2301)

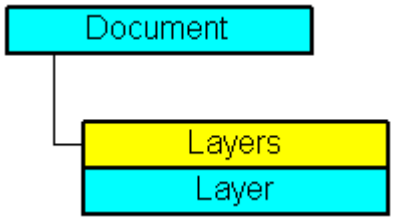
MaxZoom-Eigenschaft (Seite 2284)

LDNames-Eigenschaft (Seite 2262)

ActiveLayer-Eigenschaft (Seite 2068)

Layers-Objekt (Auflistung)

Beschreibung



Eine Auflistung von Layer-Objekten, welche die 32 Ebenen im Bild darstellen.

VBA-Objektbezeichnung

HMILayer

Verwendung

Verwenden Sie die LayersCS- oder LayersRT-Eigenschaft, um die Layers-Auflistung zurückzugeben. Im folgenden Beispiel werden die Ebenennamen in der Kopie des aktiven Bild ausgegeben:

```
Sub ShowLayer()
'VBA283
Dim colLayers As HMILayers
Dim objLayer As HMILayer
Dim strLayerList As String
Dim iCounter As Integer
iCounter = 1
Set colLayers = ActiveDocument.Layers
For Each objLayer In colLayers
If 1 = iCounter Mod 2 And 32 > iCounter Then
strLayerList = strLayerList & vbCrLf
ElseIf 11 > iCounter Then
strLayerList = strLayerList & "       "
Else
strLayerList = strLayerList & "     "
End If
strLayerList = strLayerList & objLayer.Name
iCounter = iCounter + 1
Next objLayer
MsgBox strLayerList
End Sub
```

Siehe auch

Layer-Objekt (Seite 1967)

Item-Methode (Seite 1845)

VBA-Referenz (Seite 1735)

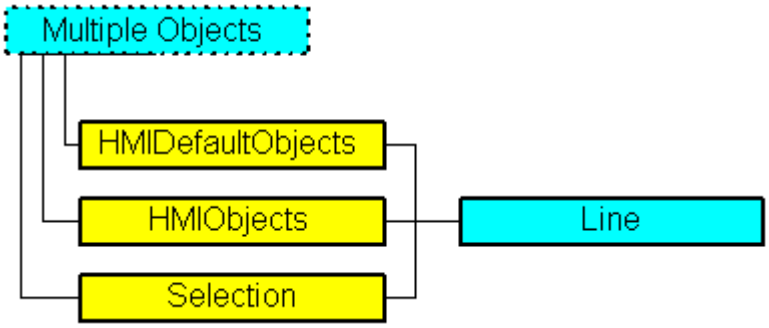
Parent-Eigenschaft (Seite 2317)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

Line-Objekt

Beschreibung



Stellt das Objekt "Linie" dar. Das Line-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMILine

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Linie" im Bild anzulegen:

```
Sub AddLine()
'VBA285
Dim objLine As HMILine
Set objLine = ActiveDocument.HMIObjects.AddHMIObject("Line1", "HMILine")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditLine()
'VBA286
Dim objLine As HMILine
Set objLine = ActiveDocument.HMIObjects("Line1")
objLine.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

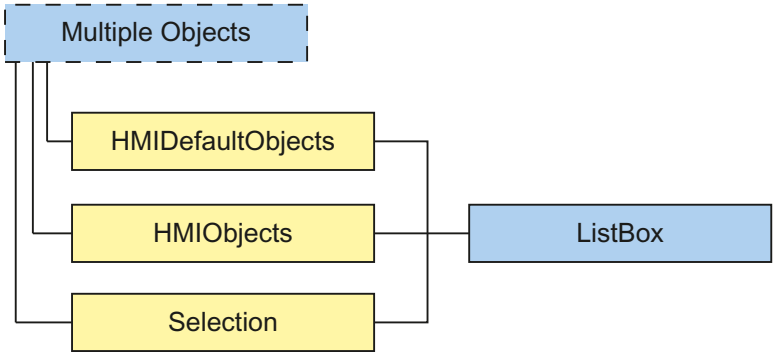
```
Sub ShowNameOfFirstSelectedObject()
'VBA287
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name of the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

AddHMIObject-Methode (Seite 1791)
BorderBackColor-Eigenschaft (Seite 2107)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
RotationAngle-Eigenschaft (Seite 2351)
ReferenceRotationTop-Eigenschaft (Seite 2348)
ReferenceRotationLeft-Eigenschaft (Seite 2347)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Index-Eigenschaft (Seite 2219)
Height-Eigenschaft (Seite 2213)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashBorderColor-Eigenschaft (Seite 2186)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
BorderEndStyle-Eigenschaft (Seite 2111)
BorderColor-Eigenschaft (Seite 2108)
ActualPointTop-Eigenschaft (Seite 2069)
ActualPointLeft-Eigenschaft (Seite 2068)

ListBox-Objekt

Beschreibung



Stellt das Objekt "Listenfeld" dar. Das ListBox-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Smart-, Windows- und Rohr-Objekten.

VBA-Objektbezeichnung

HMIListBox

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Listenfeld" im Bild anzulegen:

```
Sub AddListBox()
'VBA829
Dim objListBox As HMIListBox
Set objListBox = ActiveDocument.HMIObjects.AddHMIObject("Listenfeld", "HMIListBox")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditListBox()
'VBA830
Dim objListBox As HMIListBox
Set objListBox = ActiveDocument.HMIObjects("Listenfeld")
objListBox.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

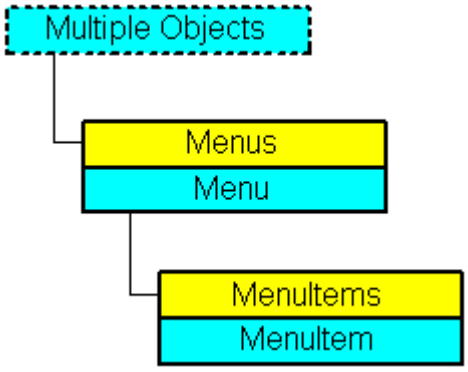
```
Sub ShowNameOfFirstSelectedObject()
'VBA831
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

ObjectName-Eigenschaft (Seite 2307)
Layer-Eigenschaft (Seite 2234)
Left-Eigenschaft (Seite 2267)
Top-Eigenschaft (Seite 2388)
Width-Eigenschaft (Seite 2486)
Height-Eigenschaft (Seite 2213)
NumberLines-Eigenschaft (Seite 2306)
ForeColor-Eigenschaft (Seite 2206)
BorderColor-Eigenschaft (Seite 2108)
BorderBackColor-Eigenschaft (Seite 2107)
BackColor-Eigenschaft (Seite 2088)
FillColor-Eigenschaft (Seite 2177)
BorderStyle-Eigenschaft (Seite 2115)
BorderWidth-Eigenschaft (Seite 2117)
FillStyle-Eigenschaft (Seite 2181)
GlobalShadow-Eigenschaft (Seite 2209)
FontName-Eigenschaft (Seite 2203)
FontSize-Eigenschaft (Seite 2204)
FontBold-Eigenschaft (Seite 2200)
FontItalic-Eigenschaft (Seite 2203)
FontUnderline-Eigenschaft (Seite 2205)
AlignmentLeft-Eigenschaft (Seite 2075)
Index-Eigenschaft (Seite 2219)
Text-Eigenschaft (Seite 2384)
Operation-Eigenschaft (Seite 2312)
PasswordLevel-Eigenschaft (Seite 2319)
Visible-Eigenschaft (Seite 2484)
TooltipText-Eigenschaft (Seite 2387)
OperationMessage-Eigenschaft (Seite 2313)
OperationReport-Eigenschaft (Seite 2314)
SelIndex-Eigenschaft (Seite 2363)
SelText-Eigenschaft (Seite 2363)

Menu-Objekt

Beschreibung



Stellt das Objekt "benutzerdefiniertes Menü" dar. Das Menu-Objekt ist Element der CustomMenus-Auflistung.

VBA-Objektbezeichnung

HMIMenu

Verwendung

Verwenden Sie CustomMenus(Index), um ein einzelnes Menu-Objekt zurückzugeben. Als "Index" können Sie entweder die Indexnummer oder den Namen des Objektes verwenden. Damit das folgende Beispiel funktioniert, legen Sie ein benutzerdefiniertes Menü an. Ein Beispiel dafür finden Sie in dieser Dokumentation unter "Neues anwendungsspezifisches Menü anlegen". Im folgenden Beispiel wird die Menübezeichnung des ersten benutzerdefinierten Menüs im aktiven Bild ausgegeben:

```
Sub ShowFirstMenuOfMenucollection()
'VBA288
Dim strName As String
strName = ActiveDocument.CustomMenus(1).Label
MsgBox strName
End Sub
```

Verwenden Sie die Delete-Methode, um aus der Auflistung "CustomMenus" ein Objekt "Menu" zu entfernen. Im folgenden Beispiel wird im aktiven Bild das erste benutzerdefinierte Menü entfernt:

```
Sub DeleteMenu()
'VBA289
Dim objMenu As HMIMenu
Set objMenu = ActiveDocument.CustomMenus(1)
objMenu.Delete
```

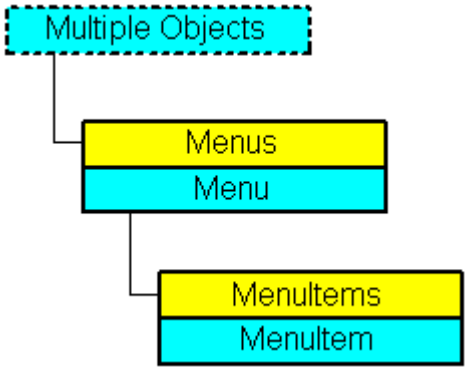
```
End Sub
```

Siehe auch

Menus-Objekt (Auflistung) (Seite 1977)
Delete-Methode (Seite 1818)
So legen Sie bildspezifische Menüs und Symbolleisten an (Seite 1657)
So legen Sie ein neues anwendungsspezifisches Menü an (Seite 1631)
VBA-Referenz (Seite 1735)
Eigene Menüs und Symbolleisten anlegen (Seite 1629)
Visible-Eigenschaft (Seite 2484)
StatusText-Eigenschaft (Seite 2374)
Position-Eigenschaft (Seite 2334)
Parent-Eigenschaft (Seite 2317)
MenuItems-Eigenschaft (Seite 2298)
LDStatusTexts-Eigenschaft (Seite 2263)
LDLabelTexts-Eigenschaft (Seite 2261)
Label-Eigenschaft (Seite 2231)
Key-Eigenschaft (Seite 2229)
Enabled-Eigenschaft (Seite 2170)
Application-Eigenschaft (Seite 2079)

Menus-Objekt (Auflistung)

Beschreibung



Eine Auflistung von Menu-Objekten, die alle benutzerdefinierten Menüs im Graphics Designer darstellen.

VBA-Objektbezeichnung

HMIMenus

Verwendung

Verwenden Sie die CustomMenus-Eigenschaft, um die Menus-Auflistung zurückzugeben. Im folgenden Beispiel werden alle benutzerdefinierten Menüs im aktiven Bild ausgegeben.

Hinweis

Die Menus-Auflistung unterscheidet bei der Ausgabe nicht zwischen anwendungs- und bildspezifischen Menüs.

```
Sub ShowCustomMenusOfDocument()
'VBA290
Dim colMenus As HMIMenus
Dim objMenu As HMIMenu
Dim strMenuList As String
Set colMenus = ActiveDocument.CustomMenus
For Each objMenu In colMenus
strMenuList = strMenuList & objMenu.Label & vbCrLf
Next objMenu
MsgBox strMenuList
End Sub
```

Verwenden Sie die Application-Eigenschaft und die InsertMenu-Methode, wenn Sie ein anwendungsspezifisches Menü anlegen wollen. Legen Sie den VBA-Code entweder im Dokument "Project Template Document" oder "Global Template Document" an. Im folgenden Beispiel wird das anwendungsspezifische Menü "myApplicationMenu" angelegt:

```
Sub InsertApplicationSpecificMenu()
'VBA291
Dim objMenu As HMIMenu
Set objMenu = Application.CustomMenus.InsertMenu(1, "a_Menu1", "myApplicationMenu")
End Sub
```

Verwenden Sie die ActiveDocument-Eigenschaft und die InsertMenu-Methode, wenn Sie ein bildspezifisches Menü anlegen wollen. Legen Sie den VBA-Code im Dokument "ThisDocument" an. Im folgenden Beispiel wird das bildspezifische Menü "myDocumentMenu" angelegt:

```
Sub InsertDocumentSpecificMenu()
'VBA292
Dim objMenu As HMIMenu
Set objMenu = ActiveDocument.CustomMenus.InsertMenu(1, "d_Menu1", "myDocumentMenu")
End Sub
```

Siehe auch

Menu-Objekt (Seite 1976)

Item-Methode (Seite 1845)

InsertMenu-Methode (Seite 1836)

So legen Sie bildspezifische Menüs und Symbolleisten an (Seite 1657)

So legen Sie ein neues anwendungsspezifisches Menü an (Seite 1631)

VBA-Referenz (Seite 1735)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

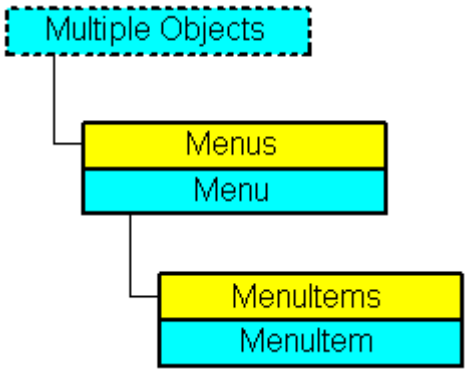
Parent-Eigenschaft (Seite 2317)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

MenuItem-Objekt

Beschreibung



Stellt den Menüeintrag eines benutzerdefinierten Menüs im GraphicsDesigner dar. Das MenuItem-Objekt ist Element der MenuItems-Auflistung.

VBA-Objektbezeichnung

HMIMenuItem

Verwendung

Hinweis

Damit die Beispiele funktionieren, legen Sie zunächst ein benutzerdefiniertes Menü an. Ein Beispiel dafür finden Sie z.B. unter "Neuen Menüeintrag zum Menü hinzufügen".

Verwenden Sie MenuItems(Index), um ein einzelnes MenuItem-Objekt zurückzugeben. Als "Index" können Sie entweder die Indexnummer oder den Namen des Objektes verwenden. Im folgenden Beispiel wird der erste Menüeintrag des ersten benutzerdefinierten Menüs im aktiven Bild ausgegeben:

```
Sub ShowFirstObjectOfCollection()
'VBA293
Dim strName As String
strName = ActiveDocument.CustomMenus(1).MenuItems(1).Label
MsgBox strName
End Sub
```

Verwenden Sie die Delete-Methode, um ein Objekt aus der Auflistung "MenuItems" zu entfernen. Im folgenden Beispiel wird der erste Menüeintrag des ersten benutzerdefinierten Menüs des aktiven Bildes gelöscht:

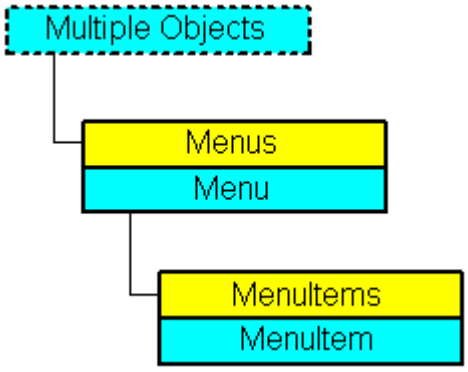
```
Sub DeleteMenuItem()
'VBA294
ActiveDocument.CustomMenus(1).MenuItems(1).Delete
End Sub
```

Siehe auch

Parent-Eigenschaft (Seite 2317)
MenuItems-Objekt (Auflistung) (Seite 1982)
Delete-Methode (Seite 1818)
Menüs und Symbolleisten konfigurieren (Seite 1628)
So weisen Sie Menüs und Symbolleisten VBA-Makros zu (Seite 1644)
So weisen Sie Menüs und Symbolleisten Hilfetexte zu (Seite 1641)
So fügen Sie einen neuen Menüeintrag zum Menü hinzu (Seite 1632)
VBA-Referenz (Seite 1735)
Eigene Menüs und Symbolleisten anlegen (Seite 1629)
Visible-Eigenschaft (Seite 2484)
Tag-Eigenschaft (Seite 2381)
SubMenu-Eigenschaft (Seite 2375)
StatusText-Eigenschaft (Seite 2374)
ShortCut-Eigenschaft (Seite 2366)
Position-Eigenschaft (Seite 2334)
MenuItemType-Eigenschaft (Seite 2299)
Macro-Eigenschaft (Seite 2281)
LDStatusTexts-Eigenschaft (Seite 2263)
LDLabelTexts-Eigenschaft (Seite 2261)
Label-Eigenschaft (Seite 2231)
Key-Eigenschaft (Seite 2229)
Icon-Eigenschaft (Seite 2218)
Enabled-Eigenschaft (Seite 2170)
Checked-Eigenschaft (Seite 2128)
Application-Eigenschaft (Seite 2079)

MenuItems-Objekt (Auflistung)

Beschreibung



Eine Auflistung von MenuItem-Objekten, die alle Menüeinträge eines benutzerdefinierten Menüs darstellen.

Verwendung

Hinweis

Damit die Beispiele funktionieren, legen Sie zunächst ein benutzerdefiniertes Menü an. Ein Beispiel dafür finden Sie z.B. unter "Neuen Menüeintrag zum Menü hinzufügen".

Verwenden Sie die MenuItems-Eigenschaft, um die MenuItems-Auflistung zurückzugeben. Im folgenden Beispiel werden alle Menüeinträge des ersten benutzerdefinierten Menüs im aktiven Bild ausgegeben.

Hinweis

Die MenuItems-Auflistung unterscheidet bei der Ausgabe nicht zwischen anwendungs- und bildspezifischen Menü.

```
Sub ShowMenuItems()
'VBA295
Dim colMenuItems As HMIMenuItems
Dim objMenuItem As HMIMenuItem
Dim strItemList As String
Set colMenuItems = ActiveDocument.CustomMenus(1).MenuItems
For Each objMenuItem In colMenuItems
strItemList = strItemList & objMenuItem.Label & vbCrLf
Next objMenuItem
MsgBox strItemList
End Sub
```

Verwenden Sie z.B. die InsertMenuItem-Methode, um einen Menüeintrag in ein vorhandenes benutzerdefiniertes Menü einzufügen. Im folgenden Beispiel wird das bildspezifische Menü "DocMenu2" im aktiven Bild angelegt und der Menüeintrag "MenuItem 1" eingefügt:

```
Sub InsertMenuItem()
'VBA296
Dim objMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Set objMenu = ActiveDocument.CustomMenus.InsertMenu(2, "d_Menu2", "DocMenu2")
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(1, "m_Item2_1", "MenuItem 1")
End Sub
```

Siehe auch

InsertSubmenu-Methode (Seite 1840)

MenuItem-Objekt (Seite 1979)

InsertSeparator-Methode (Seite 1839)

InsertMenuItem-Methode (Seite 1838)

So fügen Sie einen neuen Menüeintrag zum Menü hinzu (Seite 1632)

VBA-Referenz (Seite 1735)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

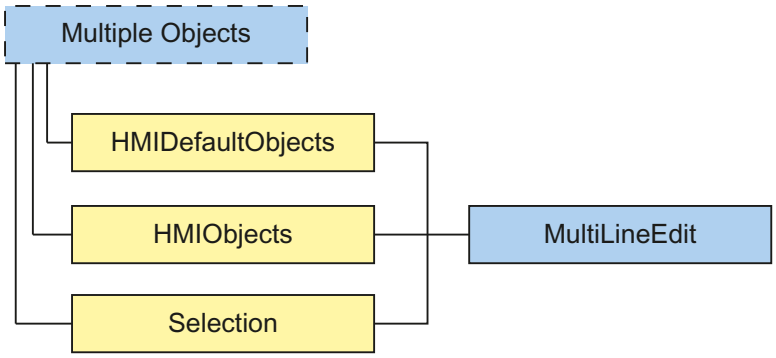
Parent-Eigenschaft (Seite 2317)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

MultiLineEdit-Objekt

Beschreibung



Stellt das Objekt "Mehrzeiliger Text" dar. Das MultiLineEdit-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Smart-, Windows- und Rohr-Objekten.

VBA-Objektbezeichnung

HMIMultiLineEdit

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Mehrzeiliger Text" im Bild anzulegen:

```
Sub AddMultiLineEdit()
'VBA832
Dim objMultiLineEdit As HMIMultiLineEdit
Set objMultiLineEdit = ActiveDocument.HMIObjects.AddHMIObject("Mehrzeiliger Text",
"HMIMultiLineEdit")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditMultiLineEdit()
'VBA833
Dim objMultiLineEdit As HMIMultiLineEdit
Set objMultiLineEdit = ActiveDocument.HMIObjects("Mehrzeiliger Text")
objMultiLineEdit.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

```
Sub ShowNameOfFirstSelectedObject()
'VBA834
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

Layer-Eigenschaft (Seite 2234)
Left-Eigenschaft (Seite 2267)
BorderColor-Eigenschaft (Seite 2108)
BorderBackColor-Eigenschaft (Seite 2107)
BorderStyle-Eigenschaft (Seite 2115)
BorderWidth-Eigenschaft (Seite 2117)
BackColor-Eigenschaft (Seite 2088)
FontName-Eigenschaft (Seite 2203)
FontSize-Eigenschaft (Seite 2204)
FontBold-Eigenschaft (Seite 2200)
FontItalic-Eigenschaft (Seite 2203)
FontUnderline-Eigenschaft (Seite 2205)
ForeColor-Eigenschaft (Seite 2206)
AlignmentLeft-Eigenschaft (Seite 2075)
Top-Eigenschaft (Seite 2388)
Width-Eigenschaft (Seite 2486)
Height-Eigenschaft (Seite 2213)
Text-Eigenschaft (Seite 2384)
Operation-Eigenschaft (Seite 2312)
PasswordLevel-Eigenschaft (Seite 2319)
Visible-Eigenschaft (Seite 2484)
TooltipText-Eigenschaft (Seite 2387)
ObjectName-Eigenschaft (Seite 2307)
GlobalShadow-Eigenschaft (Seite 2209)

objConnection-Objekt

Beschreibung

Stellt das Objekt "Verbinder" dar. Das objConnection-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.

Hinweis

Auf die Eigenschaften des objConnection-Objektes haben Sie nur Lesezugriff.

VBA-Objektbezeichnung

HMIobjConnection

Verwendung

Über die Eigenschaften des objConnection-Objektes erfahren Sie, welche Objekte verbunden sind.

Beispiel

Damit das folgende Beispiel funktioniert, müssen Sie im aktiven Bild des Graphics Designer zwei Objekte mit dem Verbinder verbunden haben. Das Objekt Verbinder finden Sie im Graphics Designer in der Objektpalette unter "Standard-Objekte". Geben Sie dem Verbinder den Namen "Connector1", damit dieses Beispiel funktioniert.

Im benutzerdefinierten Menü "Connector Info" können Sie mit dem Eintrag "Info Verbinder" anzeigen lassen, welche Objekte der Verbinder verbindet:

```
Sub ShowConnectorInfo_Menu()
'VBA297
Dim objMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Dim strDocName As String
strDocName = Application.ApplicationDataPath & ActiveDocument.Name
Set objMenu = Documents(strDocName).CustomMenus.InsertMenu(1, "ConnectorMenu",
"Connector_Info")
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(1, "ShowConnectInfo", "Info Connector")
End Sub

Sub ShowConnectorInfo()
Dim objConnector As HMIObjConnection
Dim iStart As Integer
Dim iEnd As Integer
Dim strStart As String
Dim strEnd As String
Dim strObjStart As String
Dim strObjEnd As String
Set objConnector = ActiveDocument.HMIObjects("Connector1")
iStart = objConnector.BottomConnectedConnectionPointIndex
iEnd = objConnector.TopConnectedConnectionPointIndex
strObjStart = objConnector.BottomConnectedObjectName
strObjEnd = objConnector.TopConnectedObjectName
Select Case iStart
Case 0
strStart = "top"
Case 1
strStart = "right"
Case 2
strStart = "bottom"
Case 3
strStart = "left"
End Select
```

```
Select Case iEnd
Case 0
strEnd = "top"
Case 1
strEnd = "right"
Case 2
strEnd = "bottom"
Case 3
strEnd = "left"
End Select
MsgBox "The selected connector links the objects " & vbCrLf & "'" & strObjStart & "' and '"
& strObjEnd & "'" & vbCrLf & "Connected points: " & vbCrLf & strObjStart & ": " & strStart
& vbCrLf & strObjEnd & ": " & strEnd
End Sub

Private Sub Document_MenuItemClicked(ByVal MenuItem As IHMIMenuItem)
Select Case MenuItem.Key
Case "ShowConnectInfo"
Call ShowConnectorInfo
End Select
End Sub
```

Siehe auch

TopConnectedConnectionPointIndex-Eigenschaft (Seite 2389)

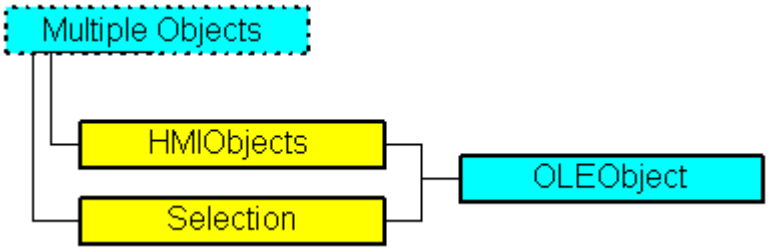
TopConnectedObjectName-Eigenschaft (Seite 2389)

BottomConnectedConnectionPointIndex-Eigenschaft (Seite 2119)

BottomConnectedObjectName-Eigenschaft (Seite 2118)

OLEObject-Objekt

Beschreibung



Stellt das Objekt "OLE-Objekt" dar. Das OLEObject-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.

VBA-Objektbezeichnung

HMIOLEObject

Verwendung

Verwenden Sie die AddOLEObject-Methode, um ein neues Objekt "OLE-Objekt" im Bild anzulegen. Im folgenden Beispiel wird ein OLE-Objekt in aktive Bild eingefügt, das ein WordPad-Dokument enthält:

```
Sub AddOLEObjectToActiveDocument()
'VBA298
Dim objOleObject As HMIOLEObject
Set objOleObject = ActiveDocument.HMIObjects.AddOLEObject("Wordpad Document",
"Wordpad.Document.1")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert. In diesem Beispiel wird die x-Koordinate des OLE-Objektes "Wordpad Document" auf 140 gesetzt:

```
Sub EditOLEObject()
'VBA299
Dim objOleObject As HMIOLEObject
Set objOleObject = ActiveDocument.HMIObjects("Wordpad Document")
objOleObject.Left = 140
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben. Als "Index" können Sie entweder die Indexnummer oder den Namen des Objektes verwenden. In diesem Beispiel wird der Name des ersten ausgewählten Objektes ausgegeben:

```
Sub ShowNameOfFirstSelectedObject()
'VBA300
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name of the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

So legen Sie bildspezifische Menüs und Symbolleisten an (Seite 1657)

Selection-Objekt (Auflistung) (Seite 2022)

HMIObjects-Objekt (Auflistung) (Seite 1957)

Delete-Methode (Seite 1818)

AddOLEObject-Methode (Seite 1794)

So legen Sie eine neue anwendungsspezifische Symbolleiste an (Seite 1637)

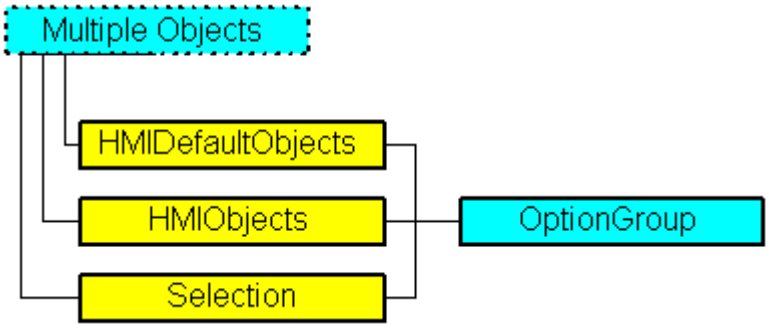
VBA-Referenz (Seite 1735)

OLE-Objekte (Seite 1672)

Application-Eigenschaft (Seite 2079)

OptionGroup-Objekt

Beschreibung



Stellt das Objekt "Radio-Box" dar. Das OptionGroup-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIOptionGroup

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Radio-Box" im Bild anzulegen:

```
Sub AddOptionGroup()
'VBA301
```

```
Dim objOptionGroup As HMIOptionGroup
Set objOptionGroup = ActiveDocument.HMIObjects.AddHMIObject("Radio-Box", "HMIOptionGroup")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditOptionGroup()
'VBA302
Dim objOptionGroup As HMIOptionGroup
Set objOptionGroup = ActiveDocument.HMIObjects("Radio-Box")
objOptionGroup.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

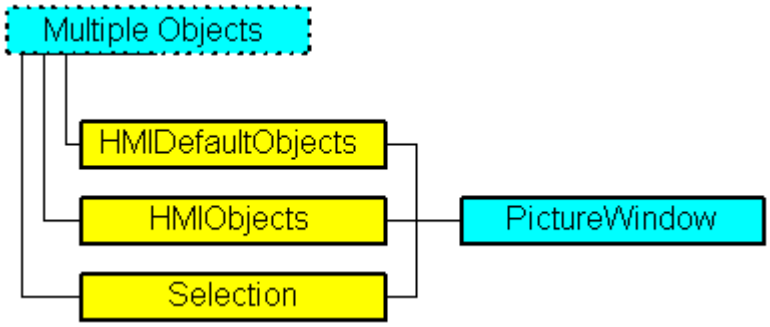
```
Sub ShowNameOfFirstSelectedObject()
'VBA303
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name of the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

Left-Eigenschaft (Seite 2267)
BorderStyle-Eigenschaft (Seite 2115)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
Text-Eigenschaft (Seite 2384)
Process-Eigenschaft (Seite 2337)
PasswordLevel-Eigenschaft (Seite 2319)
Orientation-Eigenschaft (Seite 2315)
OperationMessage-Eigenschaft (Seite 2313)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Layer-Eigenschaft (Seite 2234)
Index-Eigenschaft (Seite 2219)
Height-Eigenschaft (Seite 2213)
ForeFlashColorOn-Eigenschaft (Seite 2207)
ForeFlashColorOff-Eigenschaft (Seite 2206)
ForeColor-Eigenschaft (Seite 2206)
FontUnderline-Eigenschaft (Seite 2205)
FontSize-Eigenschaft (Seite 2204)
FontName-Eigenschaft (Seite 2203)
FontItalic-Eigenschaft (Seite 2203)
FontBold-Eigenschaft (Seite 2200)
FlashRateForeColor-Eigenschaft (Seite 2197)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashRateBackColor-Eigenschaft (Seite 2193)
FlashForeColor-Eigenschaft (Seite 2188)
FlashBorderColor-Eigenschaft (Seite 2186)
FlashBackColor-Eigenschaft (Seite 2185)
FillStyle-Eigenschaft (Seite 2181)
FillingIndex-Eigenschaft (Seite 2179)
Filling-Eigenschaft (Seite 2178)
FillColor-Eigenschaft (Seite 2177)

PictureWindow-Objekt

Beschreibung



Stellt das Objekt "Bildfenster" dar. Das PictureWindow-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIPictureWindow

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Bildfenster" im Bild anzulegen:

```
Sub AddPictureWindow()
'VBA304
Dim objPictureWindow As HMIPictureWindow
Set objPictureWindow = ActiveDocument.HMIObjects.AddHMIObject("PictureWindow1",
"HMIPictureWindow")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditPictureWindow()
'VBA305
Dim objPictureWindow As HMIPictureWindow
Set objPictureWindow = ActiveDocument.HMIObjects("PictureWindow1")
objPictureWindow.Sizeable = True
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

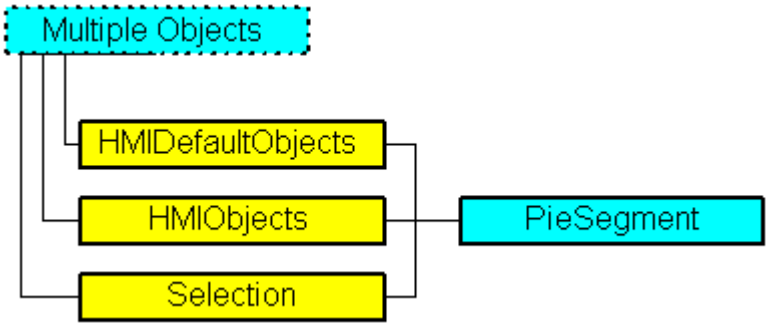
```
Sub ShowNameOfFirstSelectedObject()
'VBA306
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name of the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

MaximizeButton-Eigenschaft (Seite 2284)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Zoom-Eigenschaft (Seite 2492)
WindowBorder-Eigenschaft (Seite 2488)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
UpdateCycle-Eigenschaft (Seite 2404)
Top-Eigenschaft (Seite 2388)
TagPrefix-Eigenschaft (Seite 2382)
Sizeable-Eigenschaft (Seite 2369)
ServerPrefix-Eigenschaft (Seite 2365)
ScrollPositionY-Eigenschaft (Seite 2359)
ScrollPositionX-Eigenschaft (Seite 2359)
ScrollBars-Eigenschaft (Seite 2358)
PictureName-Eigenschaft (Seite 2329)
OnTop-Eigenschaft (Seite 2311)
OffsetTop-Eigenschaft (Seite 2311)
OffsetLeft-Eigenschaft (Seite 2310)
Name-Eigenschaft (Seite 2303)
Moveable-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
CloseButton-Eigenschaft (Seite 2137)
CaptionText-Eigenschaft (Seite 2126)
Caption-Eigenschaft (Seite 2125)
AdaptSize-Eigenschaft (Seite 2072)
AdaptPicture-Eigenschaft (Seite 2071)

PieSegment-Objekt

Beschreibung



Stellt das Objekt "Kreissegment" dar. Das PieSegment-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIPieSegment

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Kreissegment" im Bild anzulegen:

```
Sub AddPieSegment()
'VBA307
Dim objPieSegment As HMIPieSegment
Set objPieSegment = ActiveDocument.HMIObjects.AddHMIObject("PieSegment1", "HMIPieSegment")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditPieSegment()
'VBA308
Dim objPieSegment As HMIPieSegment
Set objPieSegment = ActiveDocument.HMIObjects("PieSegment1")
objPieSegment.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

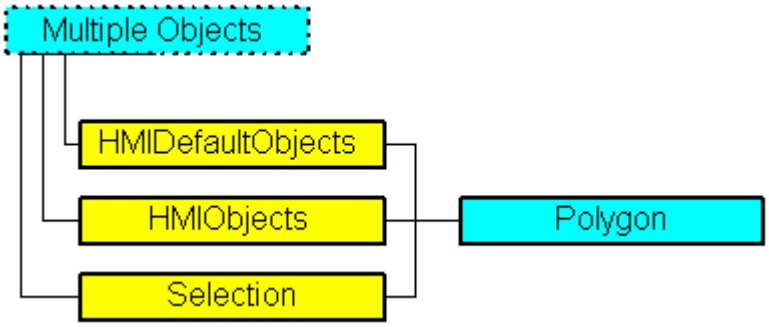
```
Sub ShowNameOfFirstSelectedObject()
'VBA309
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

Filling-Eigenschaft (Seite 2178)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
StartAngle-Eigenschaft (Seite 2373)
Radius-Eigenschaft (Seite 2345)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashRateBackColor-Eigenschaft (Seite 2193)
FlashBorderColor-Eigenschaft (Seite 2186)
FlashBackColor-Eigenschaft (Seite 2185)
FillStyle-Eigenschaft (Seite 2181)
FillingIndex-Eigenschaft (Seite 2179)
FillColor-Eigenschaft (Seite 2177)
EndAngle-Eigenschaft (Seite 2171)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
BorderColor-Eigenschaft (Seite 2108)
BorderBackColor-Eigenschaft (Seite 2107)
BackFlashColorOn-Eigenschaft (Seite 2094)
BackFlashColorOff-Eigenschaft (Seite 2093)
BackColor-Eigenschaft (Seite 2088)

Polygon-Objekt

Beschreibung



Stellt das Objekt "Polygon" dar. Das Polygon-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIPolygon

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Polygon" im Bild anzulegen:

```
Sub AddPolygon()
'VBA310
Dim objPolygon As HMIPolygon
Set objPolygon = ActiveDocument.HMIObjects.AddHMIObject("Polygon", "HMIPolygon")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditPolygon()
'VBA311
Dim objPolygon As HMIPolygon
Set objPolygon = ActiveDocument.HMIObjects("Polygon")
objPolygon.BorderColor = RGB (255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

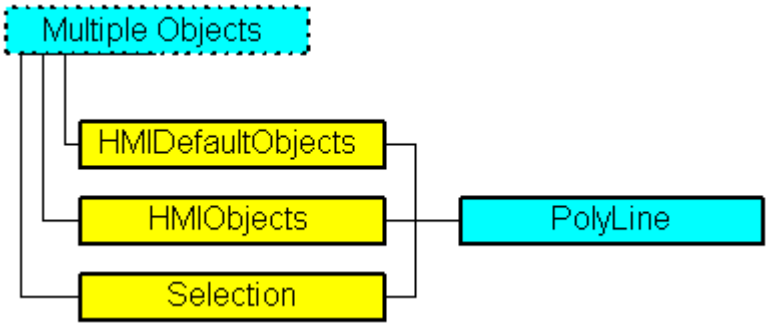
```
Sub ShowNameOfFirstSelectedObject()
'VBA312
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

TooltipText-Eigenschaft (Seite 2387)
BorderBackColor-Eigenschaft (Seite 2107)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
RotationAngle-Eigenschaft (Seite 2351)
ReferenceRotationTop-Eigenschaft (Seite 2348)
ReferenceRotationLeft-Eigenschaft (Seite 2347)
PointCount-Eigenschaft (Seite 2333)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Index-Eigenschaft (Seite 2219)
Height-Eigenschaft (Seite 2213)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashRateBackColor-Eigenschaft (Seite 2193)
FlashBorderColor-Eigenschaft (Seite 2186)
FlashBackColor-Eigenschaft (Seite 2185)
FillStyle-Eigenschaft (Seite 2181)
FillingIndex-Eigenschaft (Seite 2179)
Filling-Eigenschaft (Seite 2178)
FillColor-Eigenschaft (Seite 2177)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
BorderColor-Eigenschaft (Seite 2108)
BackFlashColorOn-Eigenschaft (Seite 2094)
BackFlashColorOff-Eigenschaft (Seite 2093)
BackColor-Eigenschaft (Seite 2088)
ActualPointTop-Eigenschaft (Seite 2069)
ActualPointLeft-Eigenschaft (Seite 2068)

PolyLine-Objekt

Beschreibung



Stellt das Objekt "Polygonzug" dar. Das PolyLine-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIPolyLine

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Polygonzug" im Bild anzulegen:

```
Sub AddPolyLine()
'VBA313
Dim objPolyLine As HMIPolyLine
Set objPolyLine = ActiveDocument.HMIObjects.AddHMIObject("PolyLine1", "HMIPolyLine")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditPolyLine()
'VBA314
Dim objPolyLine As HMIPolyLine
Set objPolyLine = ActiveDocument.HMIObjects("PolyLine1")
objPolyLine.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

```
Sub ShowNameOfFirstSelectedObject()
'VBA315
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name of the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Verwenden Sie "HMIDefaultObjects(Index)", um ein Objekt aus der HMIDefaultObjects-Auflistung zurückzugeben:

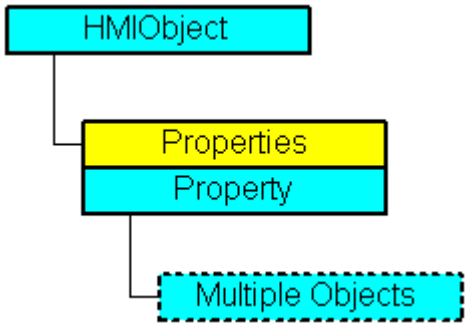
```
Sub EditDefaultPropertiesOfPolyLine()
'VBA316
Dim objPolyLine As HMIPolyLine
Set objPolyLine = Application.DefaultHMIObjects("HMIPolyLine")
objPolyLine.BorderColor = RGB(255, 255, 0)
End Sub
```

Siehe auch

HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
BorderEndStyle-Eigenschaft (Seite 2111)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
RotationAngle-Eigenschaft (Seite 2351)
ReferenceRotationTop-Eigenschaft (Seite 2348)
ReferenceRotationLeft-Eigenschaft (Seite 2347)
PointCount-Eigenschaft (Seite 2333)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Index-Eigenschaft (Seite 2219)
Height-Eigenschaft (Seite 2213)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashBorderColor-Eigenschaft (Seite 2186)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
BorderColor-Eigenschaft (Seite 2108)
BorderBackColor-Eigenschaft (Seite 2107)
ActualPointTop-Eigenschaft (Seite 2069)
ActualPointLeft-Eigenschaft (Seite 2068)

Properties-Objekt (Auflistung)

Beschreibung



Eine Auflistung von Property-Objekten, die alle Eigenschaften eines Objektes darstellen.

VBA-Objektbezeichnung

HMIProperties

Verwendung

Verwenden Sie die Properties(Index)-Eigenschaft, um ein Property-Objekt zurückzugeben, wenn Sie nicht direkt auf eine Objekteigenschaft zugreifen können. Als "Index" können Sie entweder die Indexnummer oder die VBA-Eigenschaftsbezeichnung des Objektes verwenden. Im folgenden Beispiel muss die Properties-Eigenschaft verwendet werden, um auf die individuellen Eigenschaften eines Kreises zugreifen zu können. Der Kreis wird als HMIObject-Objekt in das Bild eingefügt:

```
Sub AddObject()
'VBA319
Dim objObject As HMIObject
Set objObject = ActiveDocument.HMIObjects.AddHMIObject("CircleAsHMIObject", "HMICircle")
'
'Standard properties (e.g. "Position") are available every time:
objObject.Top = 40
objObject.Left = 40
'
'Individual properties have to be called using
'property "Properties":
objObject.Properties("FlashBackColor") = True
End Sub
```

Siehe auch

Item-Methode (Seite 1845)

VBA-Referenz (Seite 1735)

Objekte mit VBA bearbeiten (Seite 1662)

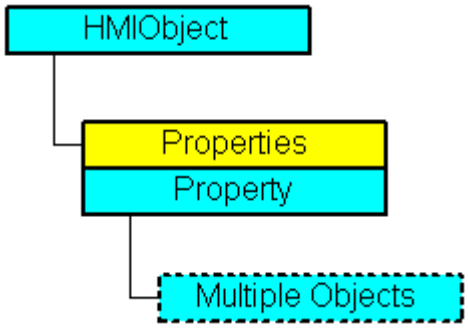
Parent-Eigenschaft (Seite 2317)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

Property-Objekt

Beschreibung



Repräsentiert die Eigenschaft eines Objektes. Beim Property-Objekt ist die Verwendung der Value-Eigenschaft voreingestellt. Aus diesem Grund können Sie folgende Schreibweise verwenden, um einer Objekteigenschaft z.B. einen neuen Wert zuzuweisen:

```
<Object>.<Property> = <Value>
```

Mit der Eigenschaft "Dynamic" können Sie eine Objekteigenschaft mit VBA dynamisieren. Für die Aktionsprojektierung mit VBA verwenden Sie die Auflistung "Events".

Das Property-Objekt ist Element der Properties-Auflistung.

VBA-Objektbezeichnung

HMIProperty

Verwendung

Verwenden Sie Properties(Index), um ein einzelnes Property-Objekt zurückzugeben. Als "Index" können Sie entweder die Indexnummer oder den Namen der Objekteigenschaft verwenden. Im folgenden Beispiel wird der Name der ersten Objekteigenschaft des Kreis-Objektes ausgegeben:

```
Sub ShowFirstObjectOfCollection()
```

```
'VBA317
Dim objCircle As HMICircle
Dim strName As String
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle", "HMICircle")
strName = objCircle.Properties(1).Name
MsgBox strName
End Sub
```

Verwenden Sie z.B. die CreateDynamic-Methode, um eine Objekteigenschaft zu dynamisieren. Im folgenden Beispiel wird die Objekteigenschaft "Radius" eines Kreises mit der Variable "Otto" dynamisiert, die alle zwei Sekunden aktualisiert wird:

```
Sub DynamicToRadiusOfNewCircle()
'VBA318
Dim objVariableTrigger As HMIVariableTrigger
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects("Circle")
Set objVariableTrigger =
objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeVariableDirect, "NewDynamic1")
objVariableTrigger.CycleType = hmiCycleType_2s
End Sub
```

Siehe auch

DisplayName-Eigenschaft (Seite 2164)

Properties-Objekt (Auflistung) (Seite 2004)

DeleteDynamic-Methode (Seite 1820)

CreateDynamic-Methode (Seite 1815)

VBA-Referenz (Seite 1735)

Dynamisierungen anlegen mit VBA (Seite 1691)

Objekte mit VBA bearbeiten (Seite 1662)

Value-Eigenschaft (Seite 2408)

Type-Eigenschaft (Seite 2392)

Parent-Eigenschaft (Seite 2317)

Name-Eigenschaft (Seite 2303)

IsDynamicable-Eigenschaft (Seite 2223)

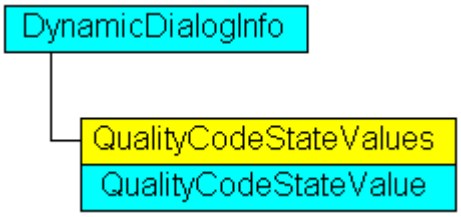
Events-Eigenschaft (Seite 2171)

Dynamic-Eigenschaft (Seite 2167)

Application-Eigenschaft (Seite 2079)

QualityCodeStateValue-Objekt

Beschreibung



Stellt den Quality Code einer Variablen dar, der im Dynamik-Dialog ein Wert zugewiesen wird, der zur Dynamisierung verwendet wird.

VBA-Objektbezeichnung

HMIQualityCodeStateValue

Objekteigenschaften

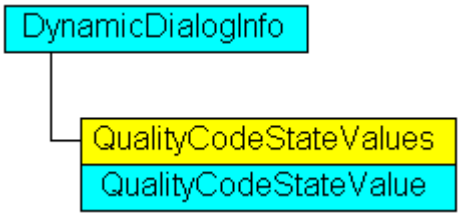
Das Objekt QualityCodeStateValue verfügt über folgende Eigenschaften:

Siehe auch

VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VBA-Referenz (Seite 1735)
VarName-Eigenschaft (Seite 2481)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
Parent-Eigenschaft (Seite 2317)
Application-Eigenschaft (Seite 2079)

QualityCodeStateValues-Objekt (Auflistung)

Beschreibung



Eine Auflistung von QualityCodeStateValue-Objekten, welche alle Quality Codes im Dynamik-Dialog enthalten, die zur Dynamsierung verwendet werden.

VBA-Objektbezeichnung

HMIQualityCodeStateValues

Verwendung

Verwenden Sie z.B. die Item-Eigenschaft, um im Dynamik-Dialog Werte festzulegen, die zur Dynamisierung verwendet werden, wenn die angegebene Variable den projektierten Quality Code zurückliefert. Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA813
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'Activate qualitycode-statecheck
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
```

```
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Objekteigenschaften

Das Objekt QualityCodeStateValues verfügt über folgende Eigenschaften:

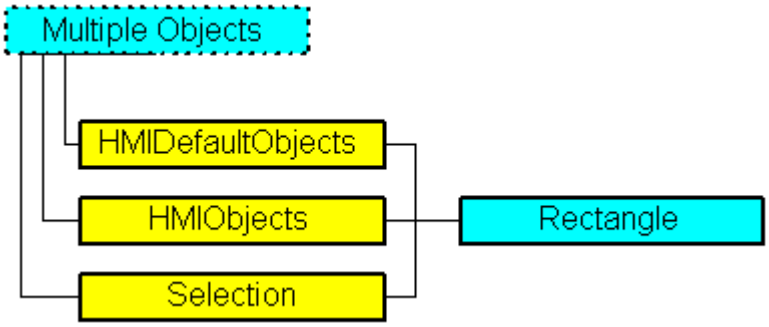
Siehe auch

VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VBA-Referenz (Seite 1735)
VarName-Eigenschaft (Seite 2481)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
Parent-Eigenschaft (Seite 2317)
Item-Eigenschaft (Seite 2225)
Count-Eigenschaft (Seite 2153)
Application-Eigenschaft (Seite 2079)

R-Z

Rectangle-Objekt

Beschreibung



Stellt das Objekt "Rechteck" dar. Das Rectangle-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIRectangle

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Rechteck" im Bild anzulegen:

```
Sub AddRectangle()
'VBA320
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditRectangle()
'VBA321
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects("Rectangle1")
objRectangle.BorderColor = RGB(255, 0, 0)
```

```
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

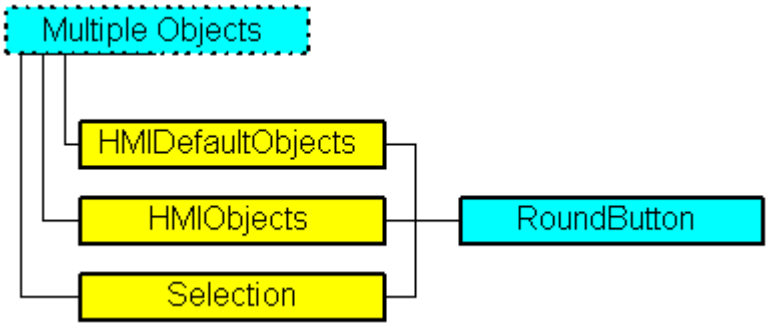
```
Sub ShowNameOfFirstSelectedObject()
'VBA322
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name of the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

Name-Eigenschaft (Seite 2303)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashRateBackColor-Eigenschaft (Seite 2193)
FlashBorderColor-Eigenschaft (Seite 2186)
FlashBackColor-Eigenschaft (Seite 2185)
FillStyle-Eigenschaft (Seite 2181)
FillingIndex-Eigenschaft (Seite 2179)
Filling-Eigenschaft (Seite 2178)
FillColor-Eigenschaft (Seite 2177)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
BorderColor-Eigenschaft (Seite 2108)
BorderBackColor-Eigenschaft (Seite 2107)
BackFlashColorOn-Eigenschaft (Seite 2094)
BackFlashColorOff-Eigenschaft (Seite 2093)
BackColor-Eigenschaft (Seite 2088)

RoundButton-Objekt

Beschreibung



Stellt das Objekt "Rundbutton" dar. Das RoundButton-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIRoundButton

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Rundbutton" im Bild anzulegen:

```
Sub AddRoundButton()
'VBA323
Dim objRoundButton As HMIRoundButton
Set objRoundButton = ActiveDocument.HMIObjects.AddHMIObject("Roundbutton1",
"HMIRoundButton")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditRoundButton()
'VBA324
Dim objRoundButton As HMIRoundButton
Set objRoundButton = ActiveDocument.HMIObjects("Roundbutton1")
objRoundButton.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

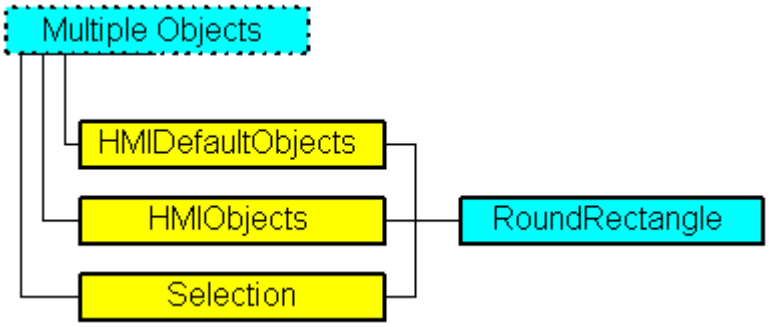
```
Sub ShowNameOfFirstSelectedObject()
'VBA325
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

TooltipText-Eigenschaft (Seite 2387)
FlashBackColor-Eigenschaft (Seite 2185)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
Toggle-Eigenschaft (Seite 2384)
Radius-Eigenschaft (Seite 2345)
Pressed-Eigenschaft (Seite 2337)
PicUpUseTransColor-Eigenschaft (Seite 2332)
PicUpTransparent-Eigenschaft (Seite 2331)
PicUpReferenced-Eigenschaft (Seite 2330)
PictureUp-Eigenschaft (Seite 2330)
PictureDown-Eigenschaft (Seite 2328)
PictureDeactivated-Eigenschaft (Seite 2327)
PicDownUseTransColor-Eigenschaft (Seite 2325)
PicDownTransparent-Eigenschaft (Seite 2324)
PicDownReferenced-Eigenschaft (Seite 2324)
PicDeactUseTransColor-Eigenschaft (Seite 2323)
PicDeactTransparent-Eigenschaft (Seite 2322)
PicDeactReferenced-Eigenschaft (Seite 2322)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashRateBackColor-Eigenschaft (Seite 2193)
FlashBorderColor-Eigenschaft (Seite 2186)
FillStyle-Eigenschaft (Seite 2181)
FillingIndex-Eigenschaft (Seite 2179)
Filling-Eigenschaft (Seite 2178)
FillColor-Eigenschaft (Seite 2177)
BorderWidth-Eigenschaft (Seite 2117)

RoundRectangle-Objekt

Beschreibung



Stellt das Objekt "Rundrechteck" dar. Das RoundRectangle-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIRoundRectangle

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Rundrechteck" im Bild anzulegen:

```
Sub AddRoundRectangle()
'VBA326
Dim objRoundRectangle As HMIRoundRectangle
Set objRoundRectangle = ActiveDocument.HMIObjects.AddHMIObject("Roundrectangle1",
"HMIRoundRectangle")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditRoundRectangle()
'VBA327
Dim objRoundRectangle As HMIRoundRectangle
Set objRoundRectangle = ActiveDocument.HMIObjects("Roundrectangle1")
objRoundRectangle.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

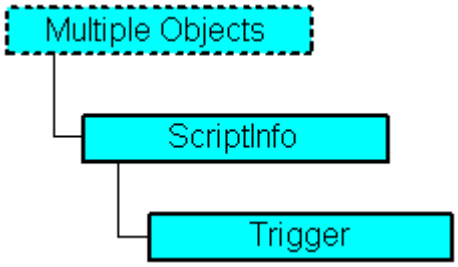
```
Sub ShowNameOfFirstSelectedObject()
'VBA328
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name of the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

Width-Eigenschaft (Seite 2486)
BorderBackColor-Eigenschaft (Seite 2107)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
RoundCornerWidth-Eigenschaft (Seite 2352)
RoundCornerHeight-Eigenschaft (Seite 2352)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashRateBackColor-Eigenschaft (Seite 2193)
FlashBorderColor-Eigenschaft (Seite 2186)
FlashBackColor-Eigenschaft (Seite 2185)
FillStyle-Eigenschaft (Seite 2181)
FillingIndex-Eigenschaft (Seite 2179)
Filling-Eigenschaft (Seite 2178)
FillColor-Eigenschaft (Seite 2177)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
BorderColor-Eigenschaft (Seite 2108)
BackFlashColorOn-Eigenschaft (Seite 2094)
BackFlashColorOff-Eigenschaft (Seite 2093)
BackColor-Eigenschaft (Seite 2088)

ScriptInfo-Objekt

Beschreibung



Stellt ein Skript (C, VB) dar, das zur Dynamisierung einer Eigenschaft oder als Aktion an ein Ereignis projektiert wird.

VBA-Objektbezeichnung

HMIScriptInfo

Verwendung

Verwenden Sie z.B. die CreateDynamic-Methode, um eine Eigenschaft mit einem Skript zu dynamisieren. Im folgenden Beispiel...

```
Sub AddDynamicAsCSkriptToProperty()
'VBA329
Dim objCScript As HMIScriptInfo
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle1", "HMICircle")
Set objCScript = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeCScript)
'
'Define triggertype and cycletime:
With objCScript
.SourceCode = ""
.Trigger.Type = hmiTriggerTypeStandardCycle
.Trigger.CycleType = hmiCycleType_2s
.Trigger.Name = "Trigger1"
End With
End Sub
```

Verwenden Sie z.B. die AddAction-Methode, um eine Aktion an ein Ereignis zu projektieren. Im folgenden Beispiel...

```
Sub AddActionToPropertyTypeCScript()
'VBA330
Dim objEvent As HMIEvent
Dim objCScript As HMIScriptInfo
Dim objCircle As HMICircle
```

```
'Add circle to picture. By changing of property "Radius"
'a C-Aktion is initiated:
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_AB", "HMICircle")
Set objEvent = objCircle.Radius.Events(1)
Set objCScript = objEvent.Actions.AddAction(hmiActionCreationTypeCScript)
End Sub
```

Siehe auch

Prototype-Eigenschaft (Seite 2341)

Delete-Methode (Seite 1818)

VBA-Referenz (Seite 1735)

Dynamisierungen anlegen mit VBA (Seite 1691)

Trigger-Eigenschaft (Seite 2391)

SourceCode-Eigenschaft (Seite 2372)

ScriptType-Eigenschaft (Seite 2357)

Parent-Eigenschaft (Seite 2317)

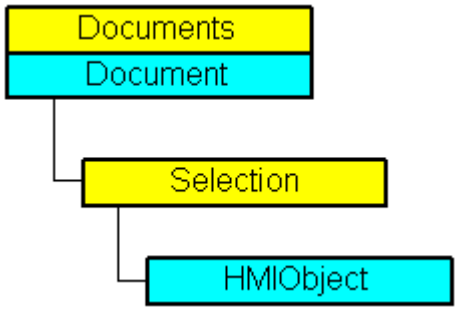
Compiled-Eigenschaft (Seite 2152)

Application-Eigenschaft (Seite 2079)

UsedLanguage-Eigenschaft (Seite 2401)

Selection-Objekt (Auflistung)

Beschreibung



Eine Auflistung von HMIObject-Objekten, die alle ausgewählten Objekte eines Bildes darstellen.

VBA-Objektbezeichnung

HMISelectedObjects

Verwendung

Verwenden Sie die Selection-Eigenschaft, um die Selection-Auflistung zurückzugeben. Im folgenden Beispiel werden die Namen von allen ausgewählten Objekten des aktiven Bildes ausgegeben:

```
Sub ShowSelectionOfDocument()
'VBA331
Dim colSelection As HMISelectedObjects
Dim objObject As HMIObject
Dim strObjectList As String
Set colSelection = ActiveDocument.Selection
If colSelection.Count <> 0 Then
strObjectList = "List of selected objects:"
For Each objObject In colSelection
strObjectList = strObjectList & vbCrLf & objObject.ObjectName
Next objObject
Else
strObjectList = "No objects selected"
End If
MsgBox strObjectList
End Sub
```

Verwenden Sie z.B. die SelectAll-Methode, um alle Objekte im Bild auszuwählen. Im folgenden Beispiel werden alle Objekte im aktiven Bild ausgewählt:

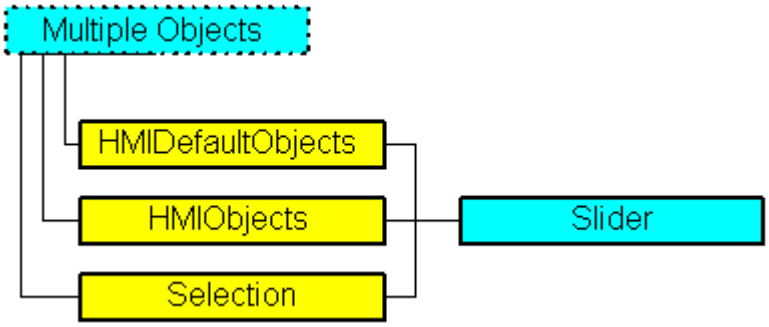
```
Sub SelectAllObjects()
'VBA332
ActiveDocument.Selection.SelectAll
End Sub
```

Siehe auch

HMIObjects-Objekt (Auflistung) (Seite 1957)
AlignTop-Methode (Seite 1800)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
BringToFront-Methode (Seite 1803)
SendToBack-Methode (Seite 1868)
SelectAll-Methode (Seite 1867)
SameWidthAndHeight-Methode (Seite 1863)
SameWidth-Methode (Seite 1861)
SameHeight-Methode (Seite 1860)
Rotate-Methode (Seite 1859)
Remove-Methode (Seite 1857)
ForwardOneLevel-Methode (Seite 1832)
BackwardOneLevel-Methode (Seite 1802)
MoveSelection-Methode (Seite 1852)
Item-Methode (Seite 1845)
FlipVertically-Methode (Seite 1831)
FlipHorizontally-Methode (Seite 1830)
EvenlySpaceVertically-Methode (Seite 1825)
EvenlySpaceHorizontally-Methode (Seite 1824)
DuplicateSelection-Methode (Seite 1823)
DeselectAll-Methode (Seite 1821)
DeleteAll-Methode (Seite 1819)
CreateGroup-Methode (Seite 1816)
CreateCustomizedObject-Methode (Seite 1813)
CopySelection-Methode (Seite 1811)
CenterVertically-Methode (Seite 1806)
CenterHorizontally-Methode (Seite 1805)
AlignRight-Methode (Seite 1799)
AlignLeft-Methode (Seite 1798)
AlignBottom-Methode (Seite 1797)
So bearbeiten Sie ein Anwender-Objekt mit VBA (Seite 1688)
So bearbeiten Sie Gruppen-Objekte mit VBA (Seite 1681)
So bearbeiten Sie Standard-, Smart- Windows- und Rohr-Objekte (Seite 1667)
VBA-Referenz (Seite 1735)
Anwender-Objekte (Seite 1687)
Gruppen-Objekte (Seite 1679)
Standard-, Smart-, Windows- und Rohr-Objekte (Seite 1665)
Objekte mit VBA bearbeiten (Seite 1662)
Parent-Eigenschaft (Seite 2317)
Count-Eigenschaft (Seite 2153)

Slider-Objekt

Beschreibung



Stellt das Objekt "Slider-Objekt" dar. Das Slider-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMISlider

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Slider-Objekt" im Bild anzulegen:

```
Sub AddSlider()
'VBA333
Dim objSlider As HMISlider
Set objSlider = ActiveDocument.HMIObjects.AddHMIObject("Slider1", "HMISlider")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditSlider()
'VBA334
Dim objSlider As HMISlider
Set objSlider = ActiveDocument.HMIObjects("Slider1")
objSlider.ButtonColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

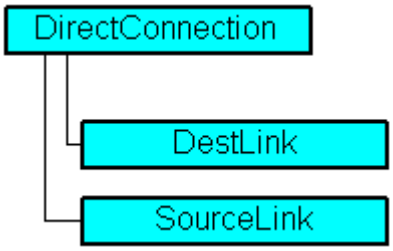
```
Sub ShowNameOfFirstSelectedObject()
'VBA335
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name of the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

OperationReport-Eigenschaft (Seite 2314)
BorderFlashColorOff-Eigenschaft (Seite 2112)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
SmallChange-Eigenschaft (Seite 2369)
Process-Eigenschaft (Seite 2337)
PasswordLevel-Eigenschaft (Seite 2319)
OperationMessage-Eigenschaft (Seite 2313)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Min-Eigenschaft (Seite 2300)
Max-Eigenschaft (Seite 2283)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashRateBackColor-Eigenschaft (Seite 2193)
FlashBorderColor-Eigenschaft (Seite 2186)
FlashBackColor-Eigenschaft (Seite 2185)
FillStyle-Eigenschaft (Seite 2181)
FillingIndex-Eigenschaft (Seite 2179)
Filling-Eigenschaft (Seite 2178)
FillColor-Eigenschaft (Seite 2177)
ExtendedOperation-Eigenschaft (Seite 2175)
Direction-Eigenschaft (Seite 2163)
ColorTop-Eigenschaft (Seite 2147)
ColorBottom-Eigenschaft (Seite 2140)
ButtonColor-Eigenschaft (Seite 2124)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderColor-Eigenschaft (Seite 2108)

SourceLink-Objekt

Beschreibung



Stellt die Quelle bei der Direktverbindung dar.

VBA-Objektbezeichnung

HMISourceLink

Verwendung

Verwenden Sie die SourceLink-Eigenschaft, um das SourceLink-Objekt zurückzugeben. Im folgenden Beispiel wird in Runtime bei einem Mausklick auf den Button die x-Position von "Rectangle_A" in die y-Position von "Rectangle_B" kopiert:

```
Sub DirectConnection()
'VBA336
Dim objButton As HMIButton
Dim objRectangleA As HMIRectangle
Dim objRectangleB As HMIRectangle
Dim objEvent As HMIEvent
Dim objDirConnection As HMIDirectConnection
'
'Add objects to active document:
Set objRectangleA = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_A", "HMIRectangle")
Set objRectangleB = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_B", "HMIRectangle")
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objRectangleA
.Top = 100
.Left = 100
End With
With objRectangleB
.Top = 250
.Left = 400
.BackColor = RGB(255, 0, 0)
End With
With objButton
.Top = 10
.Left = 10
.Text = "SetPosition"
End With
```

```
'
'Initiation of directconnection by mouseclick:
Set objDirConnection =
objButton.Events(1).Actions.AddAction(hmiActionCreationTypeDirectConnection)
With objDirConnection
'Sourceobject: Top-property of Rectangle_A
.SourceLink.Type = hmiSourceTypeProperty
.SourceLink.ObjectName = "Rectangle_A"
.SourceLink.AutomationName = "Top"
'
'Targetobject: Left-property of Rectangle_B
.DestinationLink.Type = hmiDestTypeProperty
.DestinationLink.ObjectName = "Rectangle_B"
.DestinationLink.AutomationName = "Left"
End With
End Sub
```

Siehe auch

DirectConnection-Objekt (Seite 1918)

VBA-Referenz (Seite 1735)

Type-Eigenschaft (Seite 2392)

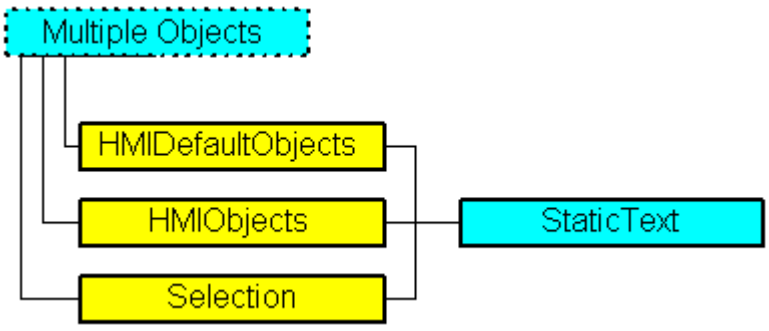
SourceLink-Eigenschaft (Seite 2370)

ObjectName-Eigenschaft (Seite 2307)

AutomationName-Eigenschaft (Seite 2082)

StaticText-Objekt

Beschreibung



Stellt das Objekt "Statischer Text" dar. Das StaticText-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIStaticText

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Statischer Text" im Bild anzulegen:

```
Sub AddStaticText()
'VBA337
Dim objStaticText As HMIStaticText
Set objStaticText = ActiveDocument.HMIObjects.AddHMIObject("Static_Text1", "HMIStaticText")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditStaticText()
'VBA338
Dim objStaticText As HMIStaticText
Set objStaticText = ActiveDocument.HMIObjects("Static_Text1")
objStaticText.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

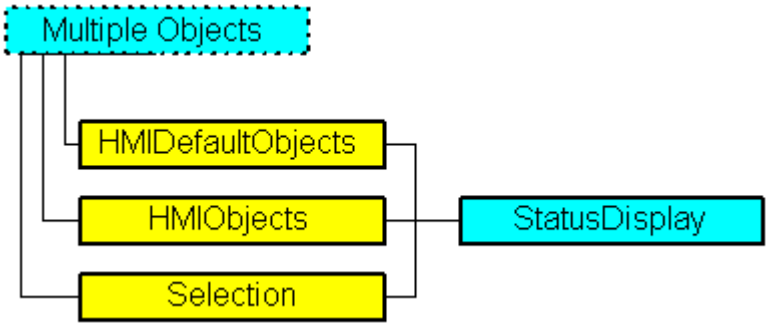
```
Sub ShowNameOfFirstSelectedObject()
'VBA339
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name of the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)
FontBold-Eigenschaft (Seite 2200)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
Text-Eigenschaft (Seite 2384)
PasswordLevel-Eigenschaft (Seite 2319)
Orientation-Eigenschaft (Seite 2315)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Height-Eigenschaft (Seite 2213)
ForeFlashColorOn-Eigenschaft (Seite 2207)
ForeFlashColorOff-Eigenschaft (Seite 2206)
ForeColor-Eigenschaft (Seite 2206)
FontUnderline-Eigenschaft (Seite 2205)
FontSize-Eigenschaft (Seite 2204)
FontName-Eigenschaft (Seite 2203)
FontItalic-Eigenschaft (Seite 2203)
FlashRateForeColor-Eigenschaft (Seite 2197)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashRateBackColor-Eigenschaft (Seite 2193)
FlashForeColor-Eigenschaft (Seite 2188)
FlashBorderColor-Eigenschaft (Seite 2186)
FlashBackColor-Eigenschaft (Seite 2185)
FillStyle-Eigenschaft (Seite 2181)
FillingIndex-Eigenschaft (Seite 2179)
Filling-Eigenschaft (Seite 2178)
FillColor-Eigenschaft (Seite 2177)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)

StatusDisplay-Objekt

Beschreibung



Stellt das Objekt "Zustandsanzeige" dar. Das "StatusDisplay"-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMIStatusDisplay

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Zustandsanzeige" im Bild anzulegen:

```
Sub AddStatusDisplay()
'VBA340
Dim objStatusDisplay As HMIStatusDisplay
Set objStatusDisplay = ActiveDocument.HMIObjects.AddHMIObject("Statusdisplay1",
"HMIStatusDisplay")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditStatusDisplay()
'VBA341
Dim objStatusDisplay As HMIStatusDisplay
Set objStatusDisplay = ActiveDocument.HMIObjects("Statusdisplay1")
objStatusDisplay.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

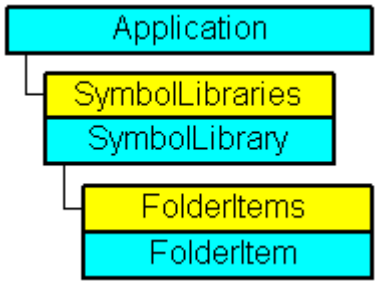
```
Sub ShowNameOfFirstSelectedObject()
'VBA342
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name of the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

TooltipText-Eigenschaft (Seite 2387)
BasePicReferenced-Eigenschaft (Seite 2099)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Width-Eigenschaft (Seite 2486)
Visible-Eigenschaft (Seite 2484)
Top-Eigenschaft (Seite 2388)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
Name-Eigenschaft (Seite 2303)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Index-Eigenschaft (Seite 2219)
Height-Eigenschaft (Seite 2213)
FlashRateFlashPic-Eigenschaft (Seite 2196)
FlashRateBorderColor-Eigenschaft (Seite 2194)
FlashPicUseTransColor-Eigenschaft (Seite 2191)
FlashPicture-Eigenschaft (Seite 2190)
FlashPicTransColor-Eigenschaft (Seite 2189)
FlashPicReferenced-Eigenschaft (Seite 2189)
FlashFlashPicture-Eigenschaft (Seite 2187)
FlashBorderColor-Eigenschaft (Seite 2186)
BorderWidth-Eigenschaft (Seite 2117)
BorderStyle-Eigenschaft (Seite 2115)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
BorderColor-Eigenschaft (Seite 2108)
BorderBackColor-Eigenschaft (Seite 2107)
BasePicUseTransColor-Eigenschaft (Seite 2101)
BasePicture-Eigenschaft (Seite 2100)
BasePicTransColor-Eigenschaft (Seite 2099)

SymbolLibrary-Objekt

Beschreibung



Stellt die "Globale Bibliothek" oder die "Projekt Bibliothek" dar. Das SymbolLibrary-Objekt ist Element der SymbolLibraries-Auflistung.

VBA-Objektbezeichnung

HMISymbolLibrary

Verwendung

Verwenden Sie SymbolLibraries(Index), um ein einzelnes SymbolLibrary-Objekt zurückzugeben. Als "Index" können Sie entweder die Indexnummer oder den Namen des Objektes verwenden. Im folgenden Beispiel wird der Name der "Globalen Bibliothek" ausgegeben:

```
Sub ShowFirstObjectOfCollection()
'VBA343
Dim strName As String
strName = Application.SymbolLibraries(1).Name
MsgBox strName
End Sub
```

Siehe auch

SymbolLibraries-Objekt (Auflistung) (Seite 2036)

GetItemByPath-Methode (Seite 1833)

FindByDisplayName-Methode (Seite 1829)

VBA-Referenz (Seite 1735)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

Parent-Eigenschaft (Seite 2317)

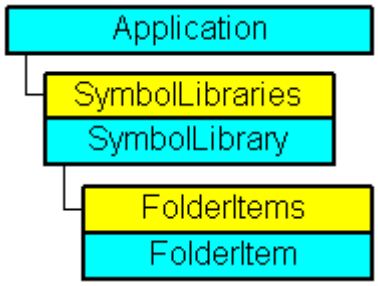
Name-Eigenschaft (Seite 2303)

FolderItems-Eigenschaft (Seite 2199)

Application-Eigenschaft (Seite 2079)

SymbolLibraries-Objekt (Auflistung)

Beschreibung



Eine Auflistung von SymbolLibrary-Objekten, welche die Bausteinbibliothek darstellen. Die Auflistung enthält zwei Objekte: Das erste Objekt ist die "Globale Bibliothek", das zweite Objekt die "Projekt Bibliothek".

VBA-Objektbezeichnung

HMISymbolLibraries

Verwendung

Verwenden Sie die SymbolLibraries-Eigenschaft, um die SymbolLibraries-Auflistung zurückzugeben. Im folgenden Beispiel werden die Namen der Bibliotheken ausgeben:

```
Sub ShowSymbolLibraries()
'VBA344
Dim colSymbolLibraries As HMISymbolLibraries
Dim objSymbolLibrary As HMISymbolLibrary
Dim strLibraryList As String
Set colSymbolLibraries = Application.SymbolLibraries
```

```
For Each objSymbolLibrary In colSymbolLibraries
strLibraryList = strLibraryList & objSymbolLibrary.Name & vbCrLf
Next objSymbolLibrary
MsgBox strLibraryList
End Sub
```

Siehe auch

SymbolLibrary-Objekt (Seite 2035)

Item-Methode (Seite 1845)

VBA-Referenz (Seite 1735)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

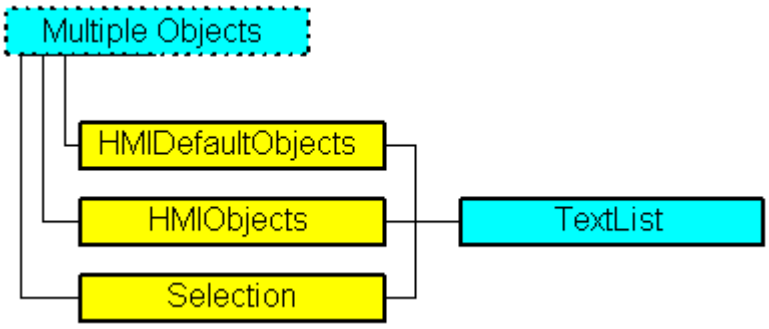
Parent-Eigenschaft (Seite 2317)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

TextList-Objekt

Beschreibung



Stellt das Objekt "Textliste" dar. Das TextList-Objekt ist Element folgender Auflistungen:

- Objects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Windows- und Smart-Objekten.

VBA-Objektbezeichnung

HMITextList

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Textliste" im Bild anzulegen:

```
Sub AddTextList()
'VBA345
Dim objTextList As HMITextList
Set objTextList = ActiveDocument.HMIObjects.AddHMIObject("Textlist1", "HMITextList")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditTextList()
'VBA346
Dim objTextList As HMITextList
Set objTextList = ActiveDocument.HMIObjects("Textlist1")
objTextList.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

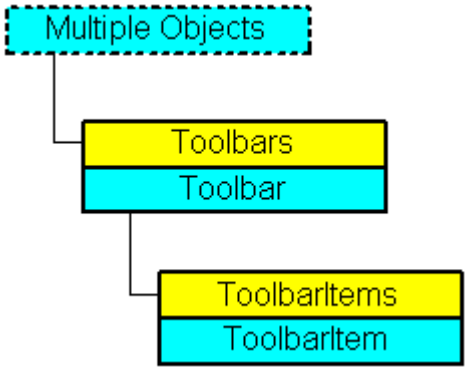
```
Sub ShowNameOfFirstSelectedObject()
'VBA347
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name of the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

Width-Eigenschaft (Seite 2486)
ForeFlashColorOn-Eigenschaft (Seite 2207)
BitNumber-Eigenschaft (Seite 2104)
Selection-Objekt (Auflistung) (Seite 2022)
HMIObjects-Objekt (Auflistung) (Seite 1957)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
AddHMIObject-Methode (Seite 1791)
VBA-Referenz (Seite 1735)
Objekte mit VBA bearbeiten (Seite 1662)
Visible-Eigenschaft (Seite 2484)
UnselTextColor-Eigenschaft (Seite 2404)
UnselBGColor-Eigenschaft (Seite 2403)
Top-Eigenschaft (Seite 2388)
TooltipText-Eigenschaft (Seite 2387)
SelTextColor-Eigenschaft (Seite 2364)
SelBGColor-Eigenschaft (Seite 2361)
PasswordLevel-Eigenschaft (Seite 2319)
OutputValue-Eigenschaft (Seite 2316)
Orientation-Eigenschaft (Seite 2315)
OperationReport-Eigenschaft (Seite 2314)
OperationMessage-Eigenschaft (Seite 2313)
Operation-Eigenschaft (Seite 2312)
NumberLines-Eigenschaft (Seite 2306)
Name-Eigenschaft (Seite 2303)
ListType-Eigenschaft (Seite 2274)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
LanguageSwitch-Eigenschaft (Seite 2233)
ItemBorderWidth-Eigenschaft (Seite 2228)
ItemBorderStyle-Eigenschaft (Seite 2228)
ItemBorderColor-Eigenschaft (Seite 2227)
ItemBorderBackColor-Eigenschaft (Seite 2226)
Height-Eigenschaft (Seite 2213)
ForeFlashColorOff-Eigenschaft (Seite 2206)
ForeColor-Eigenschaft (Seite 2206)
FontUnderline-Eigenschaft (Seite 2205)
FontSize-Eigenschaft (Seite 2204)
FontName-Eigenschaft (Seite 2203)
FontItalic-Eigenschaft (Seite 2203)
FontBold-Eigenschaft (Seite 2200)

Toolbar-Objekt

Beschreibung



Stellt das Objekt "benutzerdefinierte Symbolleiste" dar. Das Toolbar-Objekt ist Element der CustomToolbars-Auflistung.

VBA-Objektbezeichnung

HMIToolbar

Verwendung

Verwenden Sie CustomToolbars(Index), um ein einzelnes Toolbar-Objekt zurückzugeben. Als "Index" können Sie entweder die Indexnummer oder den Namen des Objektes verwenden. Im folgenden Beispiel wird der Parameter "Key" der ersten benutzerdefinierten Sybmolleiste im aktiven Bild ausgegeben:

```
Sub ShowFirstObjectOfCollection()
'VBA348
Dim strName As String
strName = ActiveDocument.CustomToolbars(1).Key
MsgBox strName
End Sub
```

Verwenden Sie die Delete-Methode, um aus der Auflistung "CustomToolbars" ein Objekt "Toolbar" zu entfernen. Im folgenden Beispiel wird im aktiven Bild die erste benutzerdefinierte Symbolleiste entfernt:

```
Sub DeleteToolbar()
'VBA349
Dim objToolbar As HMIToolbar
Set objToolbar = ActiveDocument.CustomToolbars(1)
objToolbar.Delete
End Sub
```

Siehe auch

Key-Eigenschaft (Seite 2229)

Toolbars-Objekt (Auflistung) (Seite 2041)

Delete-Methode (Seite 1818)

So legen Sie bildspezifische Menüs und Symbolleisten an (Seite 1657)

So legen Sie eine neue anwendungsspezifische Symbolleiste an (Seite 1637)

VBA-Referenz (Seite 1735)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

Visible-Eigenschaft (Seite 2484)

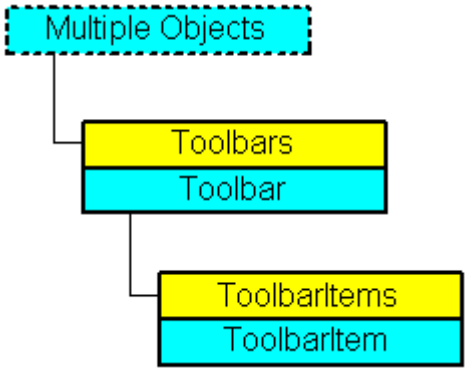
ToolbarItems-Eigenschaft (Seite 2386)

Parent-Eigenschaft (Seite 2317)

Application-Eigenschaft (Seite 2079)

Toolbars-Objekt (Auflistung)

Beschreibung



Eine Auflistung von Toolbar-Objekten, die alle benutzerdefinierten Symbolleisten im Graphics Designer darstellen.

VBA-Objektbezeichnung

HMICustomToolbars

Verwendung

Hinweis

Damit die Beispiele funktionieren, legen Sie zunächst eine benutzerdefinierte Symbolleiste an. Ein Beispiel dafür finden Sie z.B. unter "Neue anwendungsspezifische Symbolleiste anlegen".

Verwenden Sie die CustomToolbars-Eigenschaft, um die Toolbars-Auflistung zurückzugeben. Im folgenden Beispiel wird der Wert der Eigenschaft "Key" aller benutzerdefinierten Symbolleisten im aktiven Bild ausgegeben.

Hinweis

Die Toolbars-Auflistung unterscheidet bei der Ausgabe nicht zwischen anwendungs- und bildspezifischen Symbolleisten.

```
Sub ShowCustomToolbarsOfDocument()
'VBA350
Dim colToolbars As HMIToolbars
Dim objToolbar As HMIToolbar
Dim strToolbarList As String
Set colToolbars = ActiveDocument.CustomToolbars
If 0 <> colToolbars.Count Then
For Each objToolbar In colToolbars
strToolbarList = strToolbarList & objToolbar.Key & vbCrLf
Next objToolbar
Else
strToolbarList = "No toolbars existing"
End If
MsgBox strToolbarList
End Sub
```

Verwenden Sie die Application-Eigenschaft und die Add-Methode, wenn Sie eine anwendungsspezifische Symbolleiste anlegen wollen. Legen Sie den VBA-Code entweder im Dokument "Project Template Document" oder "Global Template Document" an:

```
Sub InsertApplicationSpecificToolbar()
'VBA351
Dim objToolbar As HMIToolbar
Set objToolbar = Application.CustomToolbars.Add("a_Toolbar1")
End Sub
```

Verwenden Sie die ActiveDocument-Eigenschaft und die Add-Methode, wenn Sie eine bildspezifische Symbolleiste anlegen wollen. Legen Sie den VBA-Code im Dokument "ThisDocument" an:

```
Sub InsertDocumentSpecificToolbar()
```

```
'VBA352
Dim objToolbar As HMIToolbar
Set objToolbar = ActiveDocument.CustomToolbars.Add("d_Toolbar1")
End Sub
```

Siehe auch

Toolbar-Objekt (Seite 2040)

Item-Methode (Seite 1845)

Add-Methode (Seite 1776)

So legen Sie bildspezifische Menüs und Symbolleisten an (Seite 1657)

So legen Sie eine neue anwendungsspezifische Symbolleiste an (Seite 1637)

VBA-Referenz (Seite 1735)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

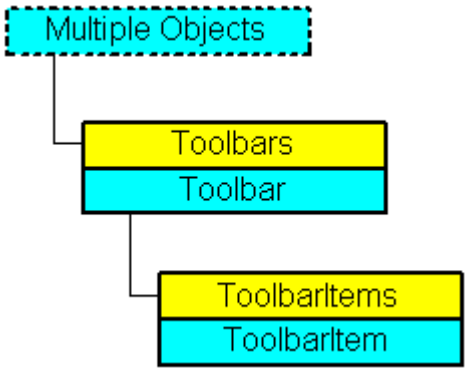
Parent-Eigenschaft (Seite 2317)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

ToolbarItem-Objekt

Beschreibung



Stellt ein Objekt (Symbol oder Trennlinie) einer benutzerdefinierten Symbolleiste im GraphicsDesigner dar. Das ToolbarItem-Objekt ist Element der ToolbarItems-Auflistung.

VBA-Objektbezeichnung

HMIToolbarItem

Verwendung

Hinweis

Damit die Beispiele funktionieren, legen Sie zunächst eine benutzerdefinierte Symbolleiste an. Ein Beispiel dafür finden Sie z.B. unter "Neue anwendungsspezifische Symbolleiste anlegen".

Verwenden Sie ToolbarItems(Index), um ein einzelnes ToolbarItem-Objekt zurückzugeben. Als "Index" können Sie entweder die Indexnummer oder den Namen des Objektes verwenden. Im folgenden Beispiel wird der Typ des ersten Objektes der ersten benutzerdefinierten Symbolleiste im aktiven Bild ausgegeben:

```
Sub ShowFirstObjectOfCollection()
'VBA353
Dim strType As String
strType = ActiveDocument.CustomToolbars(1).ToolbarItems(1).ToolbarItemType
MsgBox strType
End Sub
```

Verwenden Sie die Delete-Methode, um ein Objekt aus der Auflistung "ToolbarItems" zu entfernen. Im folgenden Beispiel wird das erste Objekt aus der ersten benutzerdefinierten Symbolleiste des aktiven Bildes gelöscht:

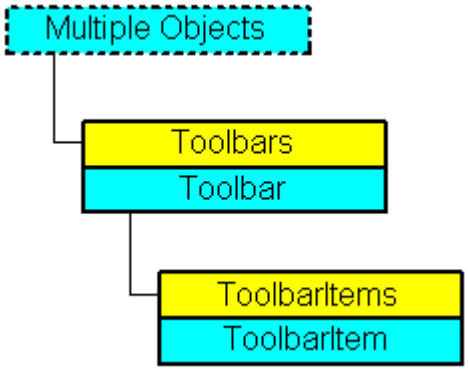
```
Sub DeleteToolbarItem()
'VBA354
ActiveDocument.CustomToolbars(1).ToolbarItems(1).Delete
End Sub
```

Siehe auch

Macro-Eigenschaft (Seite 2281)
ToolbarItems-Objekt (Auflistung) (Seite 2046)
Delete-Methode (Seite 1818)
Menüs und Symbolleisten konfigurieren (Seite 1628)
So weisen Sie Menüs und Symbolleisten VBA-Makros zu (Seite 1644)
So weisen Sie Menüs und Symbolleisten Hilfetexte zu (Seite 1641)
So fügen Sie ein neues Symbol zur Symbolleiste hinzu (Seite 1639)
VBA-Referenz (Seite 1735)
Eigene Menüs und Symbolleisten anlegen (Seite 1629)
Visible-Eigenschaft (Seite 2484)
Type-Eigenschaft (Seite 2392)
TooltipText-Eigenschaft (Seite 2387)
Tag-Eigenschaft (Seite 2381)
StatusText-Eigenschaft (Seite 2374)
ShortCut-Eigenschaft (Seite 2366)
Position-Eigenschaft (Seite 2334)
Parent-Eigenschaft (Seite 2317)
LDTooltipTexts-Eigenschaft (Seite 2265)
LDStatusTexts-Eigenschaft (Seite 2263)
Key-Eigenschaft (Seite 2229)
Icon-Eigenschaft (Seite 2218)
Enabled-Eigenschaft (Seite 2170)
Application-Eigenschaft (Seite 2079)

ToolbarItems-Objekt (Auflistung)

Beschreibung



Eine Auflistung von ToolbarItem-Objekten, die alle Objekte einer benutzerdefinierten Symbolleiste darstellen.

VBA-Objektbezeichnung

HMIToolbarItems

Verwendung

Verwenden Sie die ToolbarItems-Eigenschaft, um die ToolbarItems-Auflistung zurückzugeben. Im folgenden Beispiel werden alle Objekttypen der ersten benutzerdefinierten Symbolleiste im aktiven Bild ausgegeben.

Hinweis

Die ToolbarItems-Auflistung unterscheidet bei der Ausgabe nicht zwischen anwendungs- und bildspezifischen Symbolleisten.

```
Sub ShowToolbarItems()
'VBA355
Dim colToolbarItems As HMIToolbarItems
Dim objToolbarItem As HMIToolbarItem
Dim strTypeList As String
Set colToolbarItems = ActiveDocument.CustomToolbars(1).ToolbarItems
If 0 <> colToolbarItems.Count Then
For Each objToolbarItem In colToolbarItems
strTypeList = strTypeList & objToolbarItem.ToolbarItemType & vbCrLf
Next objToolbarItem
Else
strTypeList = "No Toolbaritems existing"
End If
MsgBox strTypeList
```

```
End Sub
```

Verwenden Sie z.B. die InsertToolbarItem-Methode, um ein Symbol in eine vorhandene benutzerdefinierte Symbolleiste einzufügen. Im folgenden Beispiel wird eine bildspezifische Symbolleiste im aktiven Bild angelegt und ein Symbol hinzugefügt:

```
Sub InsertToolbarItem()
'VBA356
Dim objToolbar As HMIToolbar
Dim objToolbarItem As HMIToolbarItem
Set objToolbar = ActiveDocument.CustomToolbars.Add("d_Toolbar2")
Set objToolbarItem = objToolbar.ToolbarItems.InsertToolbarItem(1, "t_Item2_1",
"ToolbarItem 1")
End Sub
```

Siehe auch

ToolbarItem-Objekt (Seite 2043)

InsertToolbarItem-Methode (Seite 1842)

InsertSeparator-Methode (Seite 1839)

InsertFromMenuItem-Methode (Seite 1834)

So fügen Sie ein neues Symbol zur Symbolleiste hinzu (Seite 1639)

VBA-Referenz (Seite 1735)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

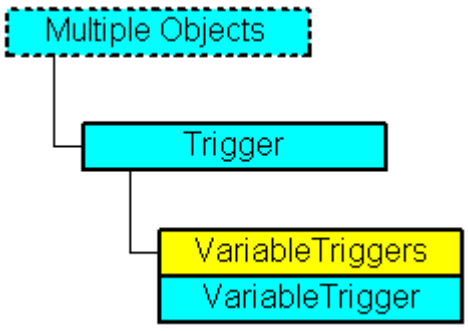
Parent-Eigenschaft (Seite 2317)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

Trigger-Objekt

Beschreibung



Stellt den Trigger (z.B. Bildzyklus) dar, der zur Dynamisierung von Eigenschaften mit Skripten benötigt wird. Ein Trigger kann mehrere Variablen-Trigger besitzen.

VBA-Objektbezeichnung

HMITrigger

Verwendung

Verwenden Sie die Trigger-Eigenschaft, um das Trigger-Objekt zurückzugeben. In diesem Beispiel wird die Eigenschaft "Radius" eines Kreises mit einem VB-Skript dynamisiert (der Rückgabewert setzt den Radius):

```
Sub AddDynamicAsVBSkriptToProperty()
'VBA357
Dim objVBScript As HMIScriptInfo
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle1", "HMICircle")
Set objVBScript = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeVBScript)
'
'Define cycletime and sourcecode
With objVBScript
.SourceCode = ""
.Trigger.Type = hmiTriggerTypeStandardCycle
.Trigger.CycleType = hmiCycleType_2s
.Trigger.Name = "Trigger1"
End With
End Sub
```

Siehe auch

Delete-Methode (Seite 1818)

VBA-Referenz (Seite 1735)

VariableTriggers-Eigenschaft (Seite 2481)

Type-Eigenschaft (Seite 2392)

Trigger-Eigenschaft (Seite 2391)

Parent-Eigenschaft (Seite 2317)

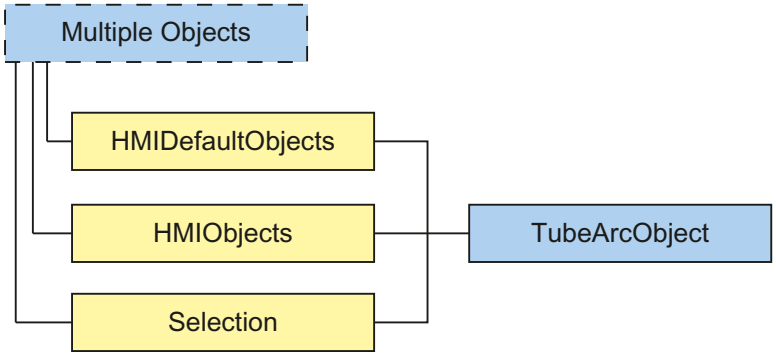
Name-Eigenschaft (Seite 2303)

CycleType-Eigenschaft (Seite 2160)

Application-Eigenschaft (Seite 2079)

TubeArcObject-Objekt

Beschreibung



Stellt das Objekt "Rohrbogen" dar. Das TubeArcObject-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Smart-, Windows- und Rohr-Objekten.

VBA-Objektbezeichnung

HMITubeArcObject

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Rohrbogen" im Bild anzulegen:

```
Sub AddTubeArcObject()
'VBA835
Dim objTubeArcObject As HMITubeArcObject
Set objTubeArcObject = ActiveDocument.HMIObjects.AddHMIObject("Rohrbogen",
"HMITubeArcObject")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditTubeArcObject()
'VBA836
Dim objTubeArcObject As HMITubeArcObject
Set objTubeArcObject = ActiveDocument.HMIObjects("Rohrbogen")
objTubeArcObject.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

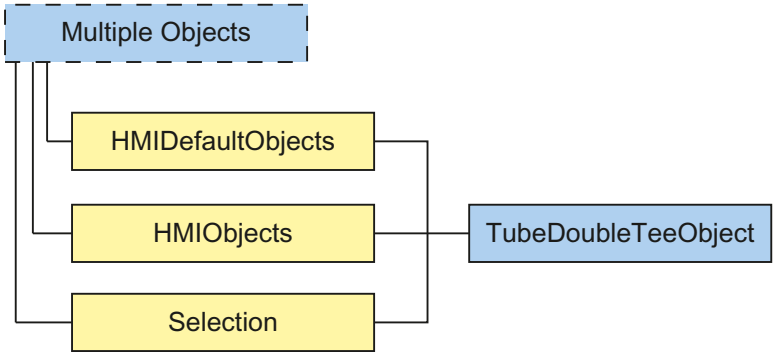
```
Sub ShowNameOfFirstSelectedObject()
'VBA837
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

ObjectName-Eigenschaft (Seite 2307)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Top-Eigenschaft (Seite 2388)
Width-Eigenschaft (Seite 2486)
Height-Eigenschaft (Seite 2213)
BorderColor-Eigenschaft (Seite 2108)
BorderWidth-Eigenschaft (Seite 2117)
TooltipText-Eigenschaft (Seite 2387)
Visible-Eigenschaft (Seite 2484)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
FlashRateBorderColor-Eigenschaft (Seite 2194)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
FlashBorderColor-Eigenschaft (Seite 2186)
Transparency-Eigenschaft (Seite 2390)
GlobalShadow-Eigenschaft (Seite 2209)
GlobalColorScheme-Eigenschaft (Seite 2208)
StartAngle-Eigenschaft (Seite 2373)
EndAngle-Eigenschaft (Seite 2171)
RadiusHeight-Eigenschaft (Seite 2345)
RadiusWidth-Eigenschaft (Seite 2346)

TubeDoubleTeeObject-Objekt

Beschreibung



Stellt das Objekt "Doppel-T-Stück" dar. Das TubeDoubleTeeObject-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Smart-, Windows- und Rohr-Objekten.

VBA-Objektbezeichnung

HMITubeDoubleTeeObject

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Doppel-T-Stück" im Bild anzulegen:

```
Sub AddTubeDoubleTeeObject()
'VBA838
Dim objTubeDoubleTeeObject As HMITubeDoubleTeeObject
Set objTubeDoubleTeeObject = ActiveDocument.HMIObjects.AddHMIObject("Doppel-T-Stück",
"HMITubeDoubleTeeObject")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditTubeDoubleTeeObject()
'VBA839
Dim objTubeDoubleTeeObject As HMITubeDoubleTeeObject
Set objTubeDoubleTeeObject = ActiveDocument.HMIObjects("Doppel-T-Stück")
objTubeDoubleTeeObject.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

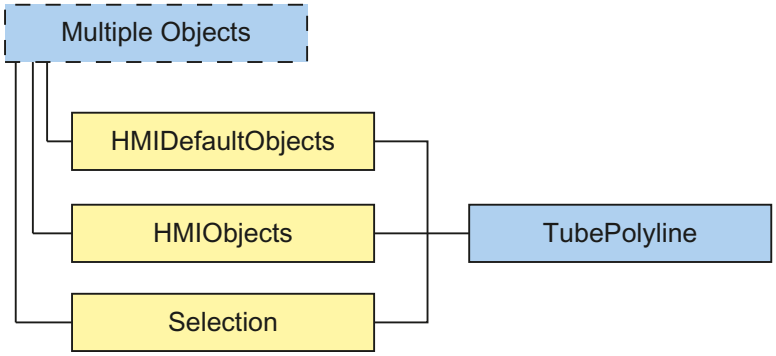
```
Sub ShowNameOfFirstSelectedObject()
'VBA840
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

ObjectName-Eigenschaft (Seite 2307)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Top-Eigenschaft (Seite 2388)
Width-Eigenschaft (Seite 2486)
Height-Eigenschaft (Seite 2213)
BorderColor-Eigenschaft (Seite 2108)
BorderWidth-Eigenschaft (Seite 2117)
TooltipText-Eigenschaft (Seite 2387)
Visible-Eigenschaft (Seite 2484)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
FlashRateBorderColor-Eigenschaft (Seite 2194)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
FlashBorderColor-Eigenschaft (Seite 2186)
Transparency-Eigenschaft (Seite 2390)
GlobalShadow-Eigenschaft (Seite 2209)
GlobalColorScheme-Eigenschaft (Seite 2208)

TubePolyline-Objekt

Beschreibung



Stellt das Objekt "Polygonrohr" dar. Das TubePolyline-Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Smart-, Windows- und Rohr-Objekten.

VBA-Objektbezeichnung

HMITubePolyline

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Polygonrohr" im Bild anzulegen:

```
Sub AddTubePolyline()
'VBA841
Dim objTubePolyline As HMITubePolyline
Set objTubePolyline = ActiveDocument.HMIObjects.AddHMIObject("Polygonrohr",
"HMITubePolyline")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditTubePolyline()
'VBA842
Dim objTubePolyline As HMITubePolyline
Set objTubePolyline = ActiveDocument.HMIObjects("Polygonrohr")
objTubePolyline.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

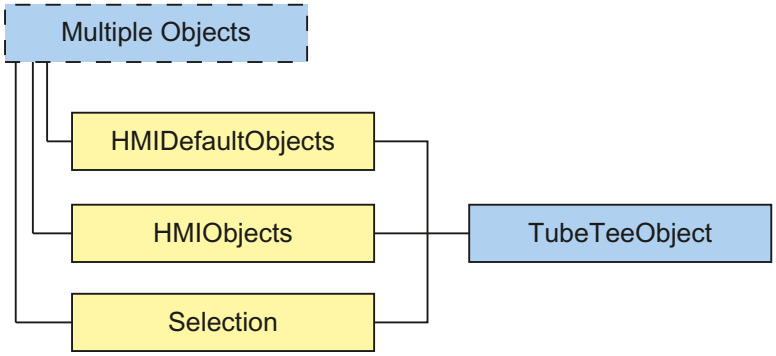
```
Sub ShowNameOfFirstSelectedObject()
'VBA843
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

ObjectName-Eigenschaft (Seite 2307)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Top-Eigenschaft (Seite 2388)
Width-Eigenschaft (Seite 2486)
Height-Eigenschaft (Seite 2213)
BorderColor-Eigenschaft (Seite 2108)
BorderWidth-Eigenschaft (Seite 2117)
TooltipText-Eigenschaft (Seite 2387)
Visible-Eigenschaft (Seite 2484)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
FlashRateBorderColor-Eigenschaft (Seite 2194)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
FlashBorderColor-Eigenschaft (Seite 2186)
Transparency-Eigenschaft (Seite 2390)
GlobalShadow-Eigenschaft (Seite 2209)
GlobalColorScheme-Eigenschaft (Seite 2208)
PointCount-Eigenschaft (Seite 2333)
ActualPointLeft-Eigenschaft (Seite 2068)
ActualPointTop-Eigenschaft (Seite 2069)
Index-Eigenschaft (Seite 2219)
LineJoinStyle-Eigenschaft (Seite 2273)

TubeTeeObject-Objekt

Beschreibung



Stellt das Objekt "T-Stück" dar. Das TubeTeeObject -Objekt ist Element folgender Auflistungen:

- HMIObjects: Enthält alle Objekte eines Bildes.
- Selection: Enthält alle Objekte eines Bildes, die ausgewählt sind.
- HMIDefaultObjects: Enthält die Vorbelegungen der Eigenschaftswerte von allen Standard-, Smart-, Windows- und Rohr-Objekten.

VBA-Objektbezeichnung

HMITubeTeeObject

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "T-Stück" im Bild anzulegen:

```
Sub AddTubeTeeObject()
'VBA844
Dim objTubeTeeObject As HMITubeTeeObject
Set objTubeTeeObject = ActiveDocument.HMIObjects.AddHMIObject("T-Stück",
"HMITubeTeeObject")
End Sub
```

Verwenden Sie "HMIObjects(Index)", um ein Objekt aus der HMIObjects-Auflistung zurückzugeben, wobei "Index" in diesem Fall das Objekt mit seinem Namen identifiziert:

```
Sub EditTubeTeeObject()
'VBA845
Dim objTubeTeeObject As HMITubeTeeObject
Set objTubeTeeObject = ActiveDocument.HMIObjects("T-Stück")
objTubeTeeObject.BorderColor = RGB(255, 0, 0)
End Sub
```

Verwenden Sie "Selection(Index)", um ein Objekt aus der Selection-Auflistung zurückzugeben:

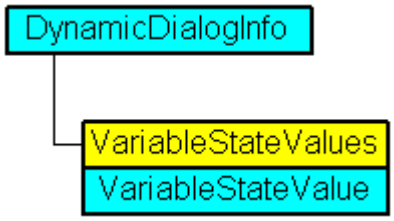
```
Sub ShowNameOfFirstSelectedObject()
'VBA846
'Select all objects in the picture:
ActiveDocument.Selection.SelectAll
'Get the name from the first object of the selection:
MsgBox ActiveDocument.Selection(1).ObjectName
End Sub
```

Siehe auch

ObjectName-Eigenschaft (Seite 2307)
Left-Eigenschaft (Seite 2267)
Layer-Eigenschaft (Seite 2234)
Top-Eigenschaft (Seite 2388)
Width-Eigenschaft (Seite 2486)
Height-Eigenschaft (Seite 2213)
BorderColor-Eigenschaft (Seite 2108)
BorderWidth-Eigenschaft (Seite 2117)
TooltipText-Eigenschaft (Seite 2387)
Visible-Eigenschaft (Seite 2484)
PasswordLevel-Eigenschaft (Seite 2319)
Operation-Eigenschaft (Seite 2312)
FlashRateBorderColor-Eigenschaft (Seite 2194)
BorderFlashColorOn-Eigenschaft (Seite 2114)
BorderFlashColorOff-Eigenschaft (Seite 2112)
FlashBorderColor-Eigenschaft (Seite 2186)
Transparency-Eigenschaft (Seite 2390)
GlobalShadow-Eigenschaft (Seite 2209)
GlobalColorScheme-Eigenschaft (Seite 2208)
RotationAngle-Eigenschaft (Seite 2351)

VariableStateValue-Objekt

Beschreibung



Stellt den Status einer Variablen dar, der im Dynamik-Dialog ein Wert zugewiesen wird, der zur Dynamisierung verwendet wird.

VBA-Objektbezeichnung

HMIVariableStateValue

Siehe auch

VALUE_SERVERDOWN-Eigenschaft (Seite 2451)

VBA-Referenz (Seite 1735)

VarName-Eigenschaft (Seite 2481)

VALUE_TIMEOUT-Eigenschaft (Seite 2454)

VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)

VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)

VALUE_MIN_RANGE-Eigenschaft (Seite 2448)

VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)

VALUE_MAX_RANGE-Eigenschaft (Seite 2445)

VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)

VALUE_INVALID_KEY-Eigenschaft (Seite 2439)

VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)

VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)

VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)

VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)

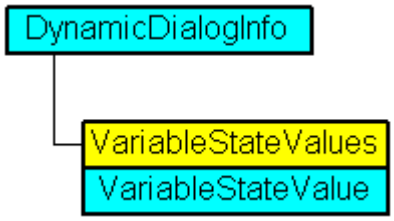
VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)

Parent-Eigenschaft (Seite 2317)

Application-Eigenschaft (Seite 2079)

VariableStateValues-Objekt (Auflistung)

Beschreibung



Eine Auflistung von VariableStateValue-Objekten, welche alle Variablenstati im Dynamik-Dialog enthalten, die zur Dynamsierung verwendet werden.

VBA-Objektbezeichnung

HMIVariableStateValues

Verwendung

Verwenden Sie z.B. die Item-Eigenschaft, um im Dynamik-Dialog Werte festzulegen, die zur Dynamisierung verwendet werden, wenn die angegebene Variable den projektierten Status zurückliefert. Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA358
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'Activate variable-statecheck
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
```

```
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VALUE_MAX_RANGE-Eigenschaft (Seite 2445)

VBA-Referenz (Seite 1735)

VarName-Eigenschaft (Seite 2481)

VALUE_TIMEOUT-Eigenschaft (Seite 2454)

VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)

VALUE_SERVERDOWN-Eigenschaft (Seite 2451)

VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)

VALUE_MIN_RANGE-Eigenschaft (Seite 2448)

VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)

VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)

VALUE_INVALID_KEY-Eigenschaft (Seite 2439)

VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)

VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)

VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)

VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)

VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)

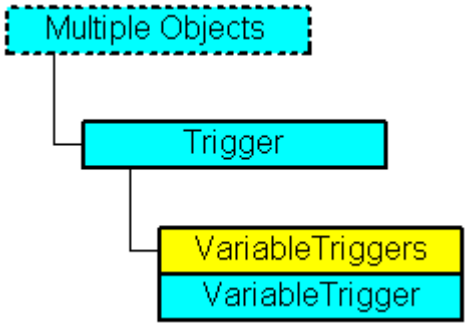
Parent-Eigenschaft (Seite 2317)

Item-Eigenschaft (Seite 2225)

Application-Eigenschaft (Seite 2079)

VariableTrigger-Objekt

Beschreibung



Stellt einen Variablen-Trigger dar.

VBA-Objektbezeichnung

HMIVariableTrigger

Verwendung

Verwenden Sie das VariableTrigger-Objekt, um einen vorhandenen Variablen-Trigger zu bearbeiten oder zu löschen. In diesem Beispiel wird die Eigenschaft "Top" einen Kreises mit der Variable "Otto" dynamisiert:

```
Sub AddDynamicAsVariableDirectToProperty()
'VBA359
Dim objVariableTrigger As HMIVariableTrigger
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle1", "HMICircle")
Set objVariableTrigger = objCircle.Top.CreateDynamic(hmiDynamicCreationTypeVariableDirect,
"'NewDynamic1'")
'
'Define cycletime
With objVariableTrigger
.CycleType = hmiCycleType_2s
End With
End Sub
```

Siehe auch

Delete-Methode (Seite 1818)

VariableTriggers-Objekt (Auflistung) (Seite 2061)

VBA-Referenz (Seite 1735)

VariableTriggers-Eigenschaft (Seite 2481)

Type-Eigenschaft (Seite 2392)

Parent-Eigenschaft (Seite 2317)

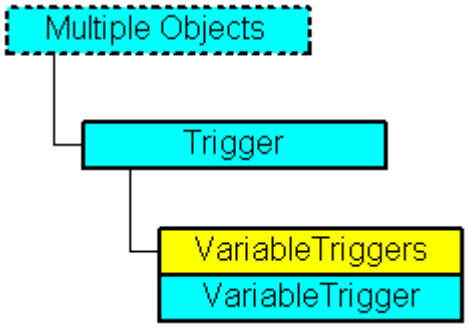
Name-Eigenschaft (Seite 2303)

CycleType-Eigenschaft (Seite 2160)

Application-Eigenschaft (Seite 2079)

VariableTriggers-Objekt (Auflistung)

Beschreibung



Eine Auflistung von VariableTrigger-Objekten, die alle verwendeten Variblen-Trigger darstellen.

VBA-Objektbezeichnung

HMIVariableTriggers

Verwendung

Verwenden Sie z.B. die Add-Methode, um einen neuen Variablen-Trigger anzulegen. Im folgenden Beispiel wird der Kreisradius mit einem VB-Skript dynamisiert. Als Trigger wird ein Variablen-Trigger verwendet:

```
Sub DynamicWithVariableTriggerCycle()
'VBA360
Dim objVBScript As HMIScriptInfo
Dim objVarTrigger As HMIVariableTrigger
```

```
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_VariableTrigger",
"HMICircle")
Set objVBScript = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeVBScript)
With objVBScript
'Definition of triggername and cycletime is to do with the Add-methode
Set objVarTrigger = .Trigger.VariableTriggers.Add("VarTrigger", hmiVariableCycleType_10s)
.SourceCode = ""
End With
End Sub
```

Siehe auch

Add-Methode (VariableTriggers-Auflistung) (Seite 1783)

VBA-Referenz (Seite 1735)

Parent-Eigenschaft (Seite 2317)

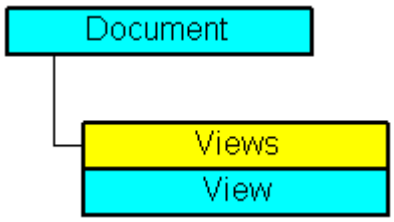
Item-Eigenschaft (Seite 2225)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

View-Objekt

Beschreibung



Stellt die Kopie eines Bildes dar. Das View-Objekt ist Element der Views-Auflistung.

Mit den Eigenschaften des View-Objektes können Sie unter anderem die Sichtbarkeit der CS Ebenen steuern und den Zoom festlegen.

VBA-Objektbezeichnung

HMIView

Verwendung

Verwenden Sie Views(Index), um ein einzelnes View-Objekt zurückzugeben. Im folgenden Beispiel wird die Anzahl der Kopien des aktiven Bildes ausgegeben:

```
Sub ShowNumberOfExistingViews()
'VBA361
Dim iMaxViews As Integer
iMaxViews = ActiveDocument.Views.Count
MsgBox "Number of copies from active document: " & iMaxViews
End Sub
```

Verwenden Sie die Add-Methode, um der Auflistung "Views" ein neues View-Objekt hinzuzufügen. Im folgenden Beispiel wird vom aktiven Bild eine Kopie erzeugt und aktiviert:

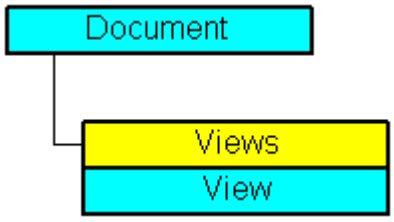
```
Sub AddView()
'VBA362
Dim objView As HMIView
Set objView = ActiveDocument.Views.Add
objView.Activate
End Sub
```

Siehe auch

Height-Eigenschaft (Seite 2213)
Views-Objekt (Auflistung) (Seite 2064)
SetCSLayerVisible-Methode (Seite 1870)
PrintDocument-Methode (Seite 1856)
IsCSLayerVisible-Methode (Seite 1843)
Delete-Methode (Seite 1818)
Add-Methode (Views-Auflistung) (Seite 1784)
Activate Methode (Seite 1776)
VBA-Referenz (Seite 1735)
Kopie eines Bildes mit VBA bearbeiten (Seite 1660)
Ebenen mit VBA bearbeiten (Seite 1659)
ExtendedZoomingEnable-Eigenschaft (Seite 2175)
Zoom-Eigenschaft (Seite 2492)
WindowState-Eigenschaft (Seite 2490)
Width-Eigenschaft (Seite 2486)
Top-Eigenschaft (Seite 2388)
ScrollPosY-Eigenschaft (Seite 2361)
ScrollPosX-Eigenschaft (Seite 2360)
Parent-Eigenschaft (Seite 2317)
Left-Eigenschaft (Seite 2267)
IsActive-Eigenschaft (Seite 2222)
Application-Eigenschaft (Seite 2079)
ActiveLayer-Eigenschaft (Seite 2068)

Views-Objekt (Auflistung)

Beschreibung



Eine Auflistung von View-Objekten, die eine Kopie eines Bildes darstellen.

VBA-Objektbezeichnung

HMIViews

Verwendung

Verwenden Sie die Views-Auflistung, um ein View-Objekt zurückzugeben. Im folgenden Beispiel wird die Anzahl der existierenden Kopien des aktiven Bildes ausgegeben:

```
Sub ShowNumberOfExistingViews()
'VBA363
Dim iMaxViews As Integer
iMaxViews = ActiveDocument.Views.Count
MsgBox "Number of copies from active document: " & iMaxViews
End Sub
```

Verwenden Sie die Add-Methode, um eine Kopie eines Bildes zu erzeugen. Im folgenden Beispiel wird vom aktiven Bild eine Kopie erzeugt und aktiviert:

```
Sub AddViewToActiveDocument()
'VBA364
Dim objView As HMIView
Set objView = ActiveDocument.Views.Add
objView.Activate
End Sub
```

Siehe auch

Item-Methode (Seite 1845)

View-Objekt (Seite 2062)

Add-Methode (Seite 1776)

VBA-Referenz (Seite 1735)

Parent-Eigenschaft (Seite 2317)

Count-Eigenschaft (Seite 2153)

Application-Eigenschaft (Seite 2079)

3.5.1.8 Eigenschaften

A

Actions-Eigenschaft

Beschreibung

Gibt die Actions-Auflistung zurück. Verwenden Sie die Actions-Eigenschaft, um eine ereignisgesteuerte Aktion zu projektieren.

Beispiel

In diesem Beispiel werden ein Button und ein Kreis in das aktive Bild eingefügt. In Runtime vergrößert sich der Kreisradius mit jedem Klick auf den Button:

```
Sub CreateVBActionToClickedEvent()
'VBA365
Dim objButton As HMIButton
Dim objCircle As HMICircle
Dim objEvent As HMIEvent
Dim objVBScript As HMIScriptInfo
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_VB", "HMICircle")
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objCircle
.Top = 100
.Left = 100
.BackColor = RGB(255, 0, 0)
End With
With objButton
.Top = 10
.Left = 10
.Width = 120
.Text = "Increase Radius"
End With
'Define event and assign sourcecode:
Set objVBScript = objButton.Events(1).Actions.AddAction(hmiActionCreationTypeVBScript)
With objVBScript
.SourceCode = "Dim myCircle" & vbCrLf & _
              "Set myCircle = HMIRuntime.ActiveScreen.ScreenItems(""Circle_VB"")" & _
              vbCrLf & "myCircle.Radius = myCircle.Radius + 5"
End With
End Sub
```

Siehe auch

Actions-Objekt (Auflistung) (Seite 1884)

AddAction-Methode (Seite 1784)

Projektierung von ereignisgesteuerten Aktionen mit VBA (Seite 1704)

ActiveDocument-Eigenschaft

Beschreibung

Gibt ein Objekt vom Typ "Document" zurück, welches das aktive Bild im Graphics Designer darstellt. Ist im Graphics Designer kein Bild geöffnet oder aktiviert, erhalten Sie eine Fehlermeldung.

Hinweis

Die Eigenschaft "ActiveDocument" bezieht sich auf das Fenster, das den Eingabefokus besitzt. Wenn andere Editoren (z.B. CrossReference) auf ein Bild zugreifen, kann sich der Eingabefokus ändern. Um daraus resultierende Fehler zu vermeiden, referenzieren das Bild über die Documents-Auflistung eindeutig.

Beispiel

Die Prozedur "CreateMenuItem()" legt das Menü "Delete Objects" an und fügt zwei Menüeinträge ("Delete Rectangles" und "Delete Circles") hinzu:

```
Sub CreateMenuItem()
'VBA366
Dim objMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
'
'Create new menu "Delete Objects":
Set objMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DeleteObjects", "Delete Objects")
'
'Add two menuitems to the menu "Delete Objects
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(1, "DeleteAllRectangles", "Delete
Rectangles")
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(2, "DeleteAllCircles", "Delete Circles")
End Sub
```

Siehe auch

Documents-Objekt (Auflistung) (Seite 1923)

ActiveLayer-Eigenschaft

Beschreibung

Legt für das Objekt View die aktive Ebene fest oder gibt sie zurück. Wertebereich von 0 bis 31, wobei "0" die oberste Ebene und "31" die unterste Ebene darstellt.

Beispiel

Die Prozedur "ActiveDocumentConfiguration()" greift auf die Eigenschaften des aktuellen Bildes im Graphics Designer zu. In diesem Beispiel wird ein neues Objekt View erzeugt und die Ebene 1 auf "aktiv" gesetzt:

```
Sub ActiveDocumentConfiguration()
'VBA367
Application.ActiveDocument.Views.Add
Application.ActiveDocument.Views(1).ActiveLayer = 2
End Sub
```

Siehe auch

View-Objekt (Seite 2062)

ActualPointLeft-Eigenschaft

Beschreibung

Legt bei den Objekten "Polygon" und "Polyline" die x-Koordinate des aktuellen Eckpunktes in Bezug auf den Bildursprung (links oben) fest oder gibt sie zurück. Jeder Eckpunkt wird über einen Index identifiziert, der sich aus der Anzahl ("PointCount") der vorhandenen Eckpunkte ableitet.

Eine Änderung des Wertes kann sich auf die Eigenschaften "Width" (Objektbreite) und "Left" (x-Koordinate der Objektposition) auswirken.

Beispiel

Die Prozedur "PolygonCoordinatesOutput()" gibt die Koordinaten aller Eckpunkte des ersten Polygonzuges im aktuellen Bild aus:

```
Sub PolygonCoordinatesOutput()
'VBA368
Dim objPolyline As HMIPolyLine
Dim iPosX As Integer
Dim iPosY As Integer
Dim iCounter As Integer
Dim strResult As String
```

```
iCounter = 1
Set objPolyline = ActiveDocument.HMIObjects.AddHMIObject("Polyline1", "HMIPolyLine")
For iCounter = 1 To objPolyline.PointCount
With objPolyline
.index = iCounter
iPosX = .ActualPointLeft
iPosY = .ActualPointTop
End With
strResult = strResult & vbCrLf & "Corner " & iCounter & ": x=" & iPosX & " y=" & iPosY
Next iCounter
MsgBox strResult
End Sub
```

Siehe auch

PointCount-Eigenschaft (Seite 2333)

Index-Eigenschaft (Seite 2219)

ActualPointTop-Eigenschaft (Seite 2069)

PolyLine-Objekt (Seite 2001)

Polygon-Objekt (Seite 1998)

Line-Objekt (Seite 1970)

ActualPointTop-Eigenschaft

Beschreibung

Legt bei den Objekten "Polygon" und "Polyline" die y-Koordinate des aktuellen Eckpunktes in Bezug auf den Bildursprung (links oben) fest oder gibt sie zurück. Jeder Eckpunkt wird über einen Index identifiziert, der sich aus der Anzahl ("PointCount") der vorhandenen Eckpunkte ableitet.

Eine Änderung des Wertes kann sich auf die Eigenschaften "Height" (Objekthöhe) und "Top" (y-Koordinate der Position) auswirken.

Beispiel

Die Prozedur "Polygon()" gibt die Koordinaten aller Eckpunkte des ersten Polygonzuges im aktuellen Bild aus:

```
Sub PolygonCoordinatesOutput()
'VBA369
Dim objPolyline As HMIPolyLine
Dim iPosX As Integer
Dim iPosY As Integer
Dim iCounter As Integer
Dim strResult As String
```

```
iCounter = 1
Set objPolyline = ActiveDocument.HMIObjects.AddHMIObject("Polyline1", "HMIPolyLine")
For iCounter = 1 To objPolyline.PointCount
With objPolyline
.index = iCounter
iPosX = .ActualPointLeft
iPosY = .ActualPointTop
End With
strResult = strResult & vbCrLf & "Corner " & iCounter & ": x=" & iPosX & " y=" & iPosY
Next iCounter
MsgBox strResult
End Sub
```

Siehe auch

PointCount-Eigenschaft (Seite 2333)

Index-Eigenschaft (Seite 2219)

ActualPointLeft-Eigenschaft (Seite 2068)

PolyLine-Objekt (Seite 2001)

Polygon-Objekt (Seite 1998)

Line-Objekt (Seite 1970)

AdaptBorder-Eigenschaft

Beschreibung

TRUE, wenn der Feldrahmen dynamisch an die Textgröße angepasst werden soll. BOOLEAN Schreib-Lese-Zugriff.

Hinweis

Bei dynamischer Änderung des Feldinhalts kann es zum Pumpen des Feldes kommen.

Mit "AdaptBorder = False" erreichen Sie eine bessere Leistung in Runtime.

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel wird die Textgröße dynamisch an die Feldgröße angepasst.

```
Sub IOFieldConfiguration()
'VBA372
Dim objIOField As HMIIOField
'
'Add new IO-Feld to active document:
```

```
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.AdaptBorder = True
End With
End Sub
```

Siehe auch

OptionGroup-Objekt (Seite 1989)

TextList-Objekt (Seite 2037)

StaticText-Objekt (Seite 2029)

IOField-Objekt (Seite 1959)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

AdaptPicture-Eigenschaft

Beschreibung

TRUE, wenn die Bildgröße der Bildfenstergröße angepasst wird. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "PictureWindowConfig" greift auf die Eigenschaften des Bildfensters zu. In diesem Beispiel wird das Bildfenster konfiguriert:

```
Sub PictureWindowConfig()
'VBA373
Dim objPicWindow As HMIPictureWindow
'
'Add new picturewindow into active document:
Set objPicWindow = ActiveDocument.HMIObjects.AddHMIObject("PicWindow1", "HMIPictureWindow")
With objPicWindow
.AdaptPicture = False
.AdaptSize = False
.Caption = True
.CaptionText = "Picturewindow in runtime"
.OffsetLeft = 5
.OffsetTop = 10
'
'Replace the picturename "Test.PDL" with the name of
'an existing document from your "GraCS"-Folder of your active project
.PictureName = "Test.PDL"
.ScrollBars = True
.ServerPrefix = ""
```

```
.TagPrefix = "Struct."
.UpdateCycle = 5
.Zoom = 100
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

AdaptSize-Eigenschaft

Beschreibung

TRUE, wenn die Bildfenstergröße der Bildgröße angepasst wird. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "PictureWindowConfig" greift auf die Eigenschaften des Bildfensters zu. In diesem Beispiel wird das Bildfenster konfiguriert:

```
Sub PictureWindowConfig()
'VBA374
Dim objPicWindow As HMIPictureWindow
'
'Add new picturewindow into active document:
Set objPicWindow = ActiveDocument.HMIObjects.AddHMIObject("PicWindow1", "HMIPictureWindow")
With objPicWindow
.AdaptPicture = False
.AdaptSize = False
.Caption = True
.CaptionText = "Picturewindow in runtime"
.OffsetLeft = 5
.OffsetTop = 10
'
'Replace the picturename "Test.PDL" with the name of
'an existing document from your "GraCS"-Folder of your active project
.PictureName = "Test.PDL"
.ScrollBars = True
.ServerPrefix = ""
.TagPrefix = "Struct."
.UpdateCycle = 5
.Zoom = 100
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

AlarmHigh-Eigenschaft

Beschreibung

Legt den oberen Grenzwert fest, bei dem Alarm ausgelöst wird oder gibt ihn zurück.

Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft "TypeAlarmHigh" fest.

Die Eigenschaft "CheckAlarmHigh" legt fest, ob die Überwachung dieses Grenzwertes aktiviert ist.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "50" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA375
Dim objBarGraph As HMIBarGraph
'
'Add new BarGraph to active document:
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolut
.TypeAlarmHigh = False
'Activate monitoring
.CheckAlarmHigh = True
'Set barcolor to "yellow"
.ColorAlarmHigh = RGB(255, 255, 0)
'set upper limit to "50"
.AlarmHigh = 50
End With
End Sub
```

Siehe auch

TypeAlarmHigh-Eigenschaft (Seite 2393)

ColorAlarmHigh-Eigenschaft (Seite 2138)

CheckAlarmHigh-Eigenschaft (Seite 2127)

BarGraph-Objekt (Seite 1893)

AlarmLow-Eigenschaft

Beschreibung

Legt den unteren Grenzwert fest, bei dem Alarm ausgelöst wird oder gibt ihn zurück.

Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft "TypeAlarmLow" fest.

Die Eigenschaft "CheckAlarmLow" legt fest, ob die Überwachung dieses Grenzwertes aktiviert ist.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "10" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA376
Dim objBarGraph As HMIBarGraph
'
'Add new BarGraph to active document:
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolut
.TypeAlarmLow = False
'Activate monitoring
.CheckAlarmLow = True
'Set Barcolor to "yellow"
.ColorAlarmLow = RGB(255, 255, 0)
'set lower limit to "10"
.AlarmLow = 10
End With
End Sub
```

Siehe auch

TypeAlarmLow-Eigenschaft (Seite 2394)

ColorAlarmLow-Eigenschaft (Seite 2139)

CheckAlarmLow-Eigenschaft (Seite 2127)

BarGraph-Objekt (Seite 1893)

Alignment-Eigenschaft

Beschreibung

Legt Darstellung der Skala (links/rechts oder oben/unten) abhängig von der Lage des BarGraph-Objektes fest oder gibt sie zurück. Die Scaling-Eigenschaft muss auf TRUE gesetzt sein, damit die Skala angezeigt wird.

Darstellung	zugeordneter Wert
Rechts oder unten	True
Links oder oben	False

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel soll sich die Skala rechts vom Balken befinden:

```
Sub BarGraphConfiguration()
'VBA377
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.Alignment = True
.Scaling = True
End With
End Sub
```

Siehe auch

Scaling-Eigenschaft (Seite 2355)

Direction-Eigenschaft (Seite 2163)

BarGraph-Objekt (Seite 1893)

AlignmentLeft-Eigenschaft

Beschreibung

Legt die horizontale Ausrichtung des Textes fest oder gibt sie zurück. Wertebereich von 0 bis 2.

Horizontale Ausrichtung	zugeordneter Wert
Links	0
Zentriert	1
Rechts	2

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel wird der Text des EA-Feldes horizontal zentriert:

```
Sub IOFieldConfiguration()
'VBA378
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.AlignmentLeft = 1
End With
End Sub
```

End SubVerwandte Themen

Siehe auch

AlignmentTop-Eigenschaft (Seite 2076)

TextList-Objekt (Seite 2037)

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

GroupDisplay-Objekt (Seite 1947)

IOField-Objekt (Seite 1959)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

AlignmentTop-Eigenschaft

Beschreibung

Legt die vertikale Ausrichtung des Textes fest oder gibt sie zurück. Wertebereich von 0 bis 2.

Horizontale Ausrichtung	zugeordneter Wert
Oben	0
Zentriert	1
Unten	2

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel wird der Text des EA-Feldes mittig zentriert:

```
Sub IOFieldConfiguration()
'VBA379
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.AlignmentLeft = 1
.AlignmentTop = 1
End With
End Sub
```

Siehe auch

AlignmentLeft-Eigenschaft (Seite 2075)

TextList-Objekt (Seite 2037)

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

GroupDisplay-Objekt (Seite 1947)

IOField-Objekt (Seite 1959)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

AnalogResultInfos-Eigenschaft

Beschreibung

Gibt die AnalogResultInfos-Auflistung zurück. Verwenden Sie die AnalogResultInfos-Eigenschaft, um im Dynamik-Dialog Wertebereiche und Eigenschaftswerte festzulegen.

Beispiel

Ein Beispiel zur Anwendung der AnalogResultInfos-Eigenschaft finden Sie in dieser Dokumentation unter "AnalogResultInfos-Objekt (Auflistung)".

Siehe auch

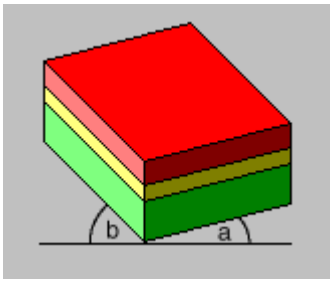
DynamicDialog-Objekt (Seite 1924)

AnalogResultInfos-Objekt (Auflistung) (Seite 1887)

AngleAlpha-Eigenschaft

Beschreibung

Legt den Tiefenwinkel a für den 3D-Effekt des "3DBarGraph"-Objektes fest oder gibt ihn zurück. Wertebereich in Grad von 0 bis 90.



Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel werden den Tiefenwinkeln a und b die Werte "15" und "45" zugewiesen:

```
Sub HMI3DBarGraphConfiguration()
'VBA380
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
'Depth-angle a = 15 degrees
.AngleAlpha = 15
'Depth-angle b = 45 degrees
.AngleBeta = 45
End With
End Sub
```

Siehe auch

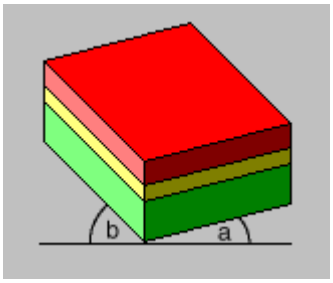
AngleBeta-Eigenschaft (Seite 2079)

3DBarGraph-Objekt (Seite 1879)

AngleBeta-Eigenschaft

Beschreibung

Legt den Tiefenwinkel b für den 3D-Effekt des "3DBarGraph"-Objektes fest oder gibt ihn zurück. Wertebereich in Grad von 0 bis 90.



Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel werden den Tiefenwinkeln a und b die Werte "15" und "45" zugewiesen:

```
Sub HMI3DBarGraphConfiguration()
'VBA381
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
'Depth-angle a = 15 degrees
.AngleAlpha = 15
'Depth-angle b = 45 degrees
.AngleBeta = 45
End With
End Sub
```

Siehe auch

AngleAlpha-Eigenschaft (Seite 2078)

3DBarGraph-Objekt (Seite 1879)

Application-Eigenschaft

Beschreibung

Gibt die Graphics-Designer-Anwendung zurück, wenn die Application-Eigenschaft ohne Objektkennzeichner verwendet wird. Wird die Application-Eigenschaft mit Objektkennzeichner verwendet, gibt sie ein Application-Objekt zurück, das die Anwendung darstellt, mit der das festgelegte Objekt erstellt wurde. Lese-Zugriff.

Beispiel

In diesem Beispiel wird ein Excel-Objekt erzeugt und der Anwendungsname ausgegeben:

```
Sub CreateExcelApplication()
'VBA382
'
'Open Excel invisible
Dim objExcelApp As New Excel.Application
MsgBox objExcelApp
'Delete the reference to Excel and close it
Set objExcelApp = Nothing
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

ApplicationDataPath-Eigenschaft

Beschreibung

Gibt den vollständigen Pfad des im Graphics Designers aktivierten Bildes zurück. Nur-Lese-Zugriff.

Beispiel

Die Prozedur "ShowApplicationDataPath()" gibt den Pfad des aktuellen Bildes aus:

```
Sub ShowApplicationDataPath()
'VBA383
MsgBox Application.ApplicationDataPath
End Sub
```

Siehe auch

Application-Eigenschaft (Seite 2079)

Application-Objekt (Seite 1888)

Assignments-Eigenschaft

Beschreibung

Eine Auflistung, welche die Zuordnungen zwischen dem Ausgabewert und dem tatsächlich auszugebenden Ausgabetext enthält.

Die Zuordnungen sind abhängig von der eingestellten Listenart. Die Listenart legen Sie mit der ListType-Eigenschaft fest.

Die Anzahl der Einträge ist abhängig von der Gesamtlänge des Strings, der an die "Assignments"-Eigenschaft übergeben wird. Dieser String darf nicht länger als 500.000 Bytes sein. Zur Überprüfung kann vor Absetzen des Zugriffs auf die "Assignments"-Eigenschaft die Funktion LenB() verwendet werden.

Beispiel

--

Siehe auch

ListType-Eigenschaft (Seite 2274)

TextList-Objekt (Seite 2037)

AssumeOnExit-Eigenschaft

Beschreibung

TRUE, wenn der eingegebene Text nach dem Verlassen des Eingabefeldes (z.B. mit der Taste <Tab> oder Mausklick) übernommen wird. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel wird eingegebener Text bei Verlassen des EA-Feldes als Eingabe übernommen.

```
Sub IOFieldConfiguration()
'VBA385
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.AssumeOnExit = True
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

IOField-Objekt (Seite 1959)

AssumeOnFull-Eigenschaft

Beschreibung

TRUE, wenn das Eingabefeld nach vollständiger Eingabe (die vorgegebene Anzahl an Zeichen wurde eingegeben) automatisch verlassen und die Eingabe dabei übernommen wird. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel wird eingegebener Text bei Verlassen des EA-Feldes als Eingabe übernommen.

```
Sub IOFieldConfiguration()
'VBA386
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.AssumeOnFull = True
End With
End Sub
```

Siehe auch

OutputFormat-Eigenschaft (Seite 2315)

DataFormat-Eigenschaft (Seite 2161)

IOField-Objekt (Seite 1959)

AutomationName-Eigenschaft

Beschreibung

Legt abhängig vom gewählten Quell- und Zielobjekttyp der Direktverbindung entweder den Namen einer Eigenschaft fest oder gibt ihn zurück.

Die beiden Tabellen zeigen Ihnen, wann Sie die AutomationName-Eigenschaft verwenden müssen. Ein "--" bedeutet, dass die Eigenschaft beim Erzeugen des DirectConnection-Objektes mit einem Leerstring (" ") vorbelegt wird.

Quellobjekttyp (SourceLink-Eigenschaft)

Type-Eigenschaft	AutomationName-Eigenschaft	ObjectName-Eigenschaft
hmiSourceTypeConstant	--	Name der Konstanten (z.B. der Bildname)
hmiSourceTypeProperty	Eigenschaft des Quellobjektes (z.B. "Top")	Name des Quellobjektes (z.B. "Rectangle_A")
hmiSourceTypePropertyOfThisObject	--	--
hmiSourceTypeVariableDirect	--	Variablenname
hmiSourceTypeVariableIndirect	--	Variablenname

Zielobjekttyp (DestinationLink-Eigenschaft)

Type-Eigenschaft	AutomationName-Eigenschaft	ObjectName-Eigenschaft
hmiDestTypeProperty	Eigenschaft des Zielobjektes (z.B. "Left")	Name des Zielobjektes (z.B. "Rectangle_A")
hmiDestTypePropertyOfThisObject	--	--
hmiDestTypePropertyOfActualWindow	Eigenschaft des Zielobjektes (z.B. "Left")	--
hmiDestTypeVariableDirect	--	Variablenname
hmiDestTypeVariableIndirect	--	Variablenname
hmiDestTypeDirectMessage	--	Variablenname
hmiDestTypeIndirectMessage	--	Variablenname

Beispiel

Im folgenden Beispiel wird in Runtime bei einem Mausklick auf den Button die x-Position von "Rectangle_A" in die y-Position von "Rectangle_B" kopiert:

```
Sub DirectConnection()
'VBA387
Dim objButton As HMIButton
Dim objRectangleA As HMIRectangle
Dim objRectangleB As HMIRectangle
Dim objEvent As HMIEvent
Dim objDynConnection As HMIDirectConnection
'
'Add objects to active document:
Set objRectangleA = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_A", "HMIRectangle")
Set objRectangleB = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_B", "HMIRectangle")
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
'
'to position and configure objects:
With objRectangleA
.Top = 100
.Left = 100
End With
```

```
With objRectangleB
.Top = 250
.Left = 400
.BackColor = RGB(255, 0, 0)
End With
With objButton
.Top = 10
.Left = 10
.Text = "SetPosition"
End With
'
'Directconnection is initiate by mouseclick:
Set objDynConnection =
objButton.Events(1).Actions.AddAction(hmiActionCreationTypeDirectConnection)
With objDynConnection
'Sourceobject: Top-Property of Rectangle_A
.SourceLink.Type = hmiSourceTypeProperty
.SourceLink.ObjectName = "Rectangle_A"
.SourceLink.AutomationName = "Top"
'
'Targetobject: Left-Property of Rectangle_B
.DestinationLink.Type = hmiDestTypeProperty
.DestinationLink.ObjectName = "Rectangle_B"
.DestinationLink.AutomationName = "Left"
End With
End Sub
```

Siehe auch

DestinationLink-Eigenschaft (Seite 2162)

Type-Eigenschaft (Seite 2392)

SourceLink-Eigenschaft (Seite 2370)

ObjectName-Eigenschaft (Seite 2307)

SourceLink-Objekt (Seite 2028)

DestLink-Objekt (Seite 1917)

AvailableDataLanguages-Eigenschaft

Beschreibung

Gibt eine Auflistung der vorhandenen Projektiersprachen zurück.

Beispiel

Die Prozedur "AusgabeDataLanguages()" gibt alle vorhandenen Projektiersprachen mit ihren Sprachkennungen (als Dezimalwert) aus:

```
Sub OutputDataLanguages()
'VBA388
Dim colDataLang As HMIDataLanguages
Dim objDataLang As HMIDataLanguage
Dim strLangList As String
Dim iCounter As Integer
'
'Save collection of datalanguages
'into variable "colDataLang"
Set colDataLang = Application.AvailableDataLanguages
iCounter = 1
'
'Get every languagename and the assigned ID
For Each objDataLang In colDataLang
With objDataLang
If 0 = iCounter Mod 3 Or 1 = iCounter Then
 strLangList = strLangList & vbCrLf & .LanguageID & " " & .LanguageName
Else
 strLangList = strLangList & " / " & .LanguageID & " " & .LanguageName
End If
End With
iCounter = iCounter + 1
Next objDataLang
MsgBox strLangList
End Sub
```

Siehe auch

LanguageName-Eigenschaft (Seite 2232)

LanguageID-Eigenschaft (Seite 2232)

So weisen Sie Menüs und Symbolleisten Hilfetexte zu (Seite 1641)

So legen Sie Menüs mehrsprachig an (Seite 1635)

Sprachabhängige Projektierung mit VBA (Seite 1626)

Average-Eigenschaft

Beschreibung

TRUE, wenn ein Mittelwert aus den letzten 10 Werten gebildet wird. Damit sich ein neuer Mittelwert bildet, muss sich ein Wert ändern. Der Mittelwert wird bei einem Bildwechsel zurückgesetzt. Wenn z. B. nach dem Bildwechsel nur ein Wert existiert, bildet sich folgender Mittelwert: (5+0+0+0+0+0+0+0+0+0)/10=0,5.

BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel soll die Mittelwertbildung aktiviert werden:

```
Sub BarGraphConfiguration()
'VBA389
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.Average = True
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

Axe-Eigenschaft

Beschreibung

Legt die Achse für die Darstellung des Meßwertes fest oder gibt sie zurück. Wertebereich von 0 bis 2.

Achse	zugeordneter Wert
X	0
Y	1
Z	2

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird die Y-Achse zur Meßwertdarstellung festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA390
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Axe = 1
End With
End Sub
```

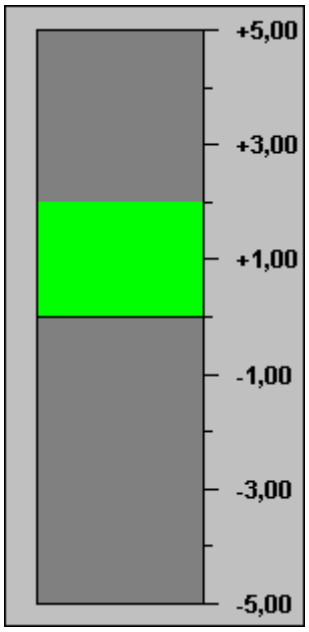
Siehe auch

3DBarGraph-Objekt (Seite 1879)

AxisSection-Eigenschaft

Beschreibung

Legt den Abstand zweier langer Achsabschnitte fest oder gibt ihn zurück. Die Angabe des Abstandes erfolgt dabei in Skaleneinheiten und ist abhängig von den projektierten Minimal- und Maximalwerten.



BarGraph-Objekt (Minimal-/Maximalwert: -5/5; AxisSection = 2)

Beispiel

Die Prozedur "BarGraphConfiguration()" greift auf die Eigenschaften des BarGraph-Objektes zu. In diesem Beispiel wird der Achsabschnitt auf "2" gesetzt:

```
Sub BarGraphConfiguration()

'VBA391

Dim objBar As HMIBarGraph

Set objBar = ActiveDocument.HMIObjects.AddHMIObject("Bar1",
"HMIBarGraph")

With objBar

.AxisSection = 2

End With

End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

B

BackBorderWidth-Eigenschaft

Beschreibung

Legt die Breite des 3D-Rahmens in Pixel fest oder gibt sie zurück. Der Wert für die Breite ist abhängig von der Größe des Objektes.

Slider

Legt die Breite der Umrandung in Pixel fest oder gibt sie zurück. BackBorderWidth = 0 unterbindet die Anzeige der Umrandung beim Slider-Objekt.

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel wird die Breite des 3D-Rahmens auf "2" gesetzt:

```
Sub ButtonConfiguration()
'VBA392
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.BackBorderWidth = 2
End With
End Sub
```

Siehe auch

Slider-Objekt (Seite 2025)

RoundButton-Objekt (Seite 2015)

GroupDisplay-Objekt (Seite 1947)

Button-Objekt (Seite 1898)

BackColor-Eigenschaft

Beschreibung

Legt die Hintergrundfarbe des Objektes fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff

Die Hintergrundfarbe wird nicht angezeigt, wenn als Füllmuster "Transparent" eingestellt ist.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel wird die Hintergrundfarbe auf "Gelb" gesetzt:

```
Sub RectangleConfiguration()
'VBA393
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.BackColor = RGB(255, 255, 0)
End With
End Sub
```

Siehe auch

EllipseSegment-Objekt (Seite 1932)
StaticText-Objekt (Seite 2029)
Slider-Objekt (Seite 2025)
TextList-Objekt (Seite 2037)
RoundRectangle-Objekt (Seite 2018)
RoundButton-Objekt (Seite 2015)
Rectangle-Objekt (Seite 2012)
Polygon-Objekt (Seite 1998)
PieSegment-Objekt (Seite 1995)
OptionGroup-Objekt (Seite 1989)
GroupDisplay-Objekt (Seite 1947)
GraphicObject-Objekt (Seite 1943)
IOField-Objekt (Seite 1959)
Ellipse-Objekt (Seite 1926)
Document-Objekt (Seite 1920)
Circle-Objekt (Seite 1902)
CheckBox-Objekt (Seite 1901)
Button-Objekt (Seite 1898)
BarGraph-Objekt (Seite 1893)
3DBarGraph-Objekt (Seite 1879)

BackColor2-Eigenschaft

Beschreibung

Legt die Balkenfarbe für die Anzeige des aktuellen Werts fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird die Balkenfarbe zur Anzeige des aktuelle Wertes auf "Gelb" gesetzt:

```
Sub BarGraphConfiguration()
'VBA394
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.BackColor2 = RGB(255, 255, 0)
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

BackColor3-Eigenschaft

Beschreibung

Legt die Farbe des Balkenhintergrunds fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird die Farbe des Balkenhintergrundes auf "Blau" gesetzt:

```
Sub BarGraphConfiguration()
'VBA395
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.BackColor3 = RGB(0, 0, 255)
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

BackColorBottom-Eigenschaft

Beschreibung

Legt die Farbe für den unteren/rechten Teil des Sliders fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "SliderConfiguration()" greift auf die Eigenschaften des Sliders zu. In diesem Beispiel wird die Farbe des unteren Teils des Sliders auf "Blau" gesetzt:

```
Sub SliderConfiguration()
'VBA396
Dim objSlider As HMISlider
Set objSlider = ActiveDocument.HMIObjects.AddHMIObject("SliderObject1", "HMISlider")
With objSlider
.BackColorBottom = RGB(0, 0, 255)
End With
End Sub
```

Siehe auch

Slider-Objekt (Seite 2025)

BackColorTop-Eigenschaft

Beschreibung

Legt die Farbe für den oberen/linken Teil des Sliders fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "SliderConfiguration()" greift auf die Eigenschaften des Sliders zu. In diesem Beispiel wird die Farbe des oberen Teils des Sliders auf "Gelb" gesetzt:

```
Sub SliderConfiguration()
'VBA397
Dim objSlider As HMISlider
Set objSlider = ActiveDocument.HMIObjects.AddHMIObject("SliderObject1", "HMISlider")
With objSlider
.BackColorTop = RGB(255, 255, 0)
End With
End Sub
```

Siehe auch

Slider-Objekt (Seite 2025)

BackFlashColorOff-Eigenschaft

Beschreibung

Legt die Farbe des Objekthintergrundes für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel wird die Farbe für den Blinkzustand "Aus" auf "Gelb" gesetzt:

```
Sub RectangleConfiguration()
'VBA398
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.BackFlashColorOff = RGB(255, 255, 0)
End With
```

```
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)
StaticText-Objekt (Seite 2029)
Slider-Objekt (Seite 2025)
TextList-Objekt (Seite 2037)
RoundRectangle-Objekt (Seite 2018)
RoundButton-Objekt (Seite 2015)
Rectangle-Objekt (Seite 2012)
Polygon-Objekt (Seite 1998)
PieSegment-Objekt (Seite 1995)
OptionGroup-Objekt (Seite 1989)
GraphicObject-Objekt (Seite 1943)
IOField-Objekt (Seite 1959)
EllipseSegment-Objekt (Seite 1932)
Ellipse-Objekt (Seite 1926)
Circle-Objekt (Seite 1902)
CheckBox-Objekt (Seite 1901)
Button-Objekt (Seite 1898)

BackFlashColorOn-Eigenschaft

Beschreibung

Legt die Farbe des Objekthintergrundes für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. I In diesem Beispiel wird die Farbe für den Blinkzustand "Ein" auf "Blau" gesetzt

```
Sub RectangleConfiguration()
'VBA399
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.BackFlashColorOn = RGB(0, 0, 255)
End With
End Suba
```

Siehe auch

RoundButton-Objekt (Seite 2015)

StaticText-Objekt (Seite 2029)

Slider-Objekt (Seite 2025)

TextList-Objekt (Seite 2037)

RoundRectangle-Objekt (Seite 2018)

Rectangle-Objekt (Seite 2012)

Polygon-Objekt (Seite 1998)

PieSegment-Objekt (Seite 1995)

OptionGroup-Objekt (Seite 1989)

GraphicObject-Objekt (Seite 1943)

IOField-Objekt (Seite 1959)

EllipseSegment-Objekt (Seite 1932)

Ellipse-Objekt (Seite 1926)

Circle-Objekt (Seite 1902)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

BarGraph-Objekt (Seite 1893)

Background-Eigenschaft

Beschreibung

TRUE, wenn der Hintergrund beim 3DBarGraph-Objekt sichtbar sein soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird der Hintergrund auf "transparent" gesetzt:

```
Sub HMI3DBarGraphConfiguration()
'VBA400
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Background = False
End With
End Sub
```

Siehe auch

3DBarGraph-Objekt (Seite 1879)

BackPictureAlignment-Eigenschaft

Beschreibung

Legt als Attribut "Darstellungsart" die Position und die Skalierung für das Hintergrundbild des Prozessbildes fest.

normal	Das Hintergrundbild wird mittig in Originalgröße wiedergegeben. Beim Aufziehen des Bildes in Runtime bleibt es am Ort.
gestreckt (Fenster)	Das Hintergrundbild wird auf das jeweils größere der beiden Fenster Runtime-Fenster und Prozessbild skaliert. In Runtime ist es auf die Größe des Runtime-Fensters skaliert und wird beim Aufziehen weiter skaliert.
gekachelt	Mit dem Bild in Originalgröße werden Graphics Designer und Prozessbild ausgelegt.
gestreckt (Bild)	Das Hintergrundbild wird auf die projektierte Größe des Prozessbildes skaliert. Beim Aufziehen des Bildes in Runtime behält es seine Größe.

BackPictureName-Eigenschaft

Beschreibung

Legt Pfad und Namen der Datei fest, die als Hintergrundbild im Prozessbild verwendet wird, oder gibt Pfad und Dateinamen zurück.

Geeignet sind Dateien der Formate EMF, WMF, DB, BMP, GIF, JPG, JPEG und ICO.

Wenn kein Pfad angegeben ist, wird die Datei im Unterverzeichnis \GraCS gesucht. Wenn Sie einen anderen Pfad angeben, wird eine Kopie im Verzeichnis \GraCS angelegt.

Pfadangaben

Folgende Pfadangabeformate sind möglich:

- absolut: z.B. "C:\Siemens\WinCC\Icons\myIcon.ICO.
- relativ: Das Ausgangsverzeichnis für die relative Pfadangabe ist das "GraCS"-Verzeichnis des aktuellen Projektes.
- <global>: Bezeichnet den Installationspfad von WinCC. Die Pfadangabe "<global>\Icons \myIcon" entspricht der Pfadangabe unter "absolut".
- <project>: Bezeichnet das aktuelle Projektverzeichnis.

BarDepth-Eigenschaft

Beschreibung

Legt die Balkentiefe in Pixel fest oder gibt sie zurück.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird die Balkentiefe auf "40" gesetzt:

```
Sub HMI3DBarGraphConfiguration()
'VBA401
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.BarDepth = 40
End With
End Sub
```

Siehe auch

3DBarGraph-Objekt (Seite 1879)

BarHeight-Eigenschaft

Beschreibung

Legt die Balkenhöhe in Pixel fest oder gibt sie zurück.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird die Balkenhöhe auf "60" gesetzt:

```
Sub HMI3DBarGraphConfiguration()
'VBA402
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.BarHeight = 60
End With
End Sub
```

Siehe auch

3DBarGraph-Objekt (Seite 1879)

BarWidth-Eigenschaft

Beschreibung

Legt die Balkenbreite in Pixel fest oder gibt sie zurück.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird die Balkenbreite auf "80" gesetzt:

```
Sub HMI3DBarGraphConfiguration()
'VBA403
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.BarWidth = 80
End With
End Sub
```

Siehe auch

3DBarGraph-Objekt (Seite 1879)

BasePicReferenced-Eigenschaft

Beschreibung

TRUE, wenn das zugeordnete Bild im Objekt Zustandsanzeige gespeichert werden soll. Sonst wird nur der zugehörige Objektverweis gespeichert. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "StatusDisplayConfiguration()" greift auf die Eigenschaften der Zustandsanzeige zu. In diesem Beispiel soll das zugeordnete Bild im Objekt Zustandsanzeige gespeichert werden:

```
Sub StatusDisplayConfiguration()
'VBA404
Dim objStatDisp As HMIStatusDisplay
Set objStatDisp = ActiveDocument.HMIObjects.AddHMIObject("Statusdisplay1",
"HMIStatusDisplay")
With objStatDisp
.BasePicReferenced = True
End With
End Sub
```

Siehe auch

StatusDisplay-Objekt (Seite 2032)

BasePicTransColor-Eigenschaft

Beschreibung

Legt fest, welche Farbe des zugeordneten Bitmap-Objekts (.bmp, .dib) auf "transparent" gesetzt werden soll oder gibt die Farbe zurück. LONG Schreib-Lese-Zugriff.

Die Farbe wird nur dann auf "transparent" gesetzt, wenn die Eigenschaft "BasePicUseTransColor" den Wert "True" hat.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "StatusDisplayConfiguration()" greift auf die Eigenschaften der Zustandsanzeige zu. In diesem Beispiel wird die Farbe "Gelb" auf "transparent" gesetzt:

```
Sub StatusDisplayConfiguration()
'VBA405
Dim objStatDisp As HMIStatusDisplay
Set objStatDisp = ActiveDocument.HMIObjects.AddHMIObject("Statusdisplay1",
"HMIStatusDisplay")
With objStatDisp
.BasePicTransColor = RGB(255, 255, 0)
.BasePicUseTransColor = True
End With
End Sub
```

Siehe auch

BasePicUseTransColor-Eigenschaft (Seite 2101)

StatusDisplay-Objekt (Seite 2032)

BasePicture-Eigenschaft

Beschreibung

Legt das Grundbild für das Objekt Zustandsanzeige fest oder gibt es zurück.

Das Bild (*.BMP oder *.DIB) muss sich im Verzeichnis "GraCS" des aktuellen Projekts befinden, damit es eingebunden werden kann.

Die Eigenschaft "BasePicReferenced" legt in diesem Zusammenhang fest, ob das Grundbild zusammen mit dem Objekt Zustandsanzeige gespeichert oder referenziert wird.

Beispiel

Die Prozedur "StatusDisplayConfiguration()" greift auf die Eigenschaften der Zustandsanzeige zu. In diesem Beispiel wird das Bild "Testbild.BMP" als Grundbild verwendet :

```
Sub StatusDisplayConfiguration()
'VBA406
Dim objStatDisp As HMIStatusDisplay
Set objStatDisp = ActiveDocument.HMIObjects.AddHMIObject("Statusdisplay1",
"HMIStatusDisplay")
With objStatDisp
'
'To use this example copy a Bitmap-Graphic
'to the "GraCS"-Folder of the actual project.
'Replace the picturename "Testpicture.BMP" with the name of
'the picture you copied
```

```
.BasePicture = "Testpicture.BMP"
End With
End Sub
```

Siehe auch

BasePicReferenced-Eigenschaft (Seite 2099)

StatusDisplay-Objekt (Seite 2032)

BasePicUseTransColor-Eigenschaft

Beschreibung

TRUE, wenn die projektierte Farbe ("BasePicTransColor"-Eigenschaft) des Bitmap-Objektes auf "transparent" gesetzt werden soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "StatusDisplayConfiguration()" greift auf die Eigenschaften der Zustandsanzeige zu. In diesem Beispiel wird die Farbe "Gelb" auf "transparent" gesetzt:

```
Sub StatusDisplayConfiguration()
'VBA407
Dim objStatDisp As HMIStatusDisplay
Set objStatDisp = ActiveDocument.HMIObjects.AddHMIObject("Statusdisplay1",
"HMIStatusDisplay")
With objStatDisp
.BasePicTransColor = RGB(255, 255, 0)
.BasePicUseTransColor = True
End With
End Sub
```

Siehe auch

BasePicTransColor-Eigenschaft (Seite 2099)

StatusDisplay-Objekt (Seite 2032)

BaseX-Eigenschaft

Beschreibung

Legt beim Objekt 3DBarGraph den horizontalen Abstand in Pixel des rechten Balkenrands zum linken Rand des Objektfeldes fest oder gibt ihn zurück.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird der horizontale Abstand auf "80" gesetzt:

```
Sub HMI3DBarGraphConfiguration()
'VBA408
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.BaseX = 80
End With
End Sub
```

Siehe auch

3DBarGraph-Objekt (Seite 1879)

BaseY-Eigenschaft

Beschreibung

Legt beim Objekt 3DBarGraph den vertikalen Abstand des unteren Balkenrands zum oberen Rand des Objektfeldes fest oder gibt ihn zurück.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird der vertikale Abstand auf "100" gesetzt:

```
Sub HMI3DBarGraphConfiguration()
'VBA409
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.BaseY = 100
End With
End Sub
```

Siehe auch

3DBarGraph-Objekt (Seite 1879)

BinaryResultInfo-Eigenschaft

Beschreibung

Gibt das BinaryResultInfo-Objekt zurück.

Beispiel

Ein Beispiel zur Anwendung der BinaryResultInfo-Eigenschaft finden Sie in dieser Dokumentation unter "BinaryResultInfo-Objekt".

Siehe auch

BinaryResultInfo-Objekt (Seite 1896)

BitNotSetValue-Eigenschaft

Beschreibung

Legt den Wert für die dynamisierte Eigenschaft fest, wenn das angegebene Bit einer projektierten Variable nicht gesetzt ist oder gibt ihn zurück.

Welches Bit gesetzt werden muss, um die Wertänderung auszulösen, legen Sie mit der Eigenschaft BitNumber fest.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert, ein Variablenname vergeben sowie das zu setzende Bit definiert und den Zuständen "gesetzt"/"nicht gesetzt" die dazugehörigen Eigenschaftswerte zugeordnet:

```
Sub AddDynamicDialogToCircleRadiusTypeBit()
'VBA410
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_B", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeBit
.Trigger.VariableTriggers(1).CycleType = hmiVariableCycleType_5s
.BitResultInfo.BitNumber = 1
.BitResultInfo.BitSetValue = 40
.BitResultInfo.BitNotSetValue = 80
End With
End Sub
```

Siehe auch

BitNumber-Eigenschaft (Seite 2104)

BitResultInfo-Objekt (Seite 1897)

BitNumber-Eigenschaft

Beschreibung

Legt das Bit fest, dessen Zustand sich ändern muss, um eine Wertänderung auszulösen oder gibt es zurück. Die verwendete Variable muss vom Typ BYTE, WORD oder DWORD sein.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert, ein Variablenname vergeben sowie das zu setzende Bit definiert und den Zuständen "gesetzt"/"nicht gesetzt" die dazugehörigen Eigenschaftswerte zugeordnet:

```
Sub AddDynamicDialogToCircleRadiusTypeBit()
'VBA411
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_B", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeBit
.BitResultInfo.BitNumber = 1
.BitResultInfo.BitSetValue = 40
.BitResultInfo.BitNotSetValue = 80
End With
End Sub
```

Siehe auch

BitResultInfo-Objekt (Seite 1897)

VBA-Referenz (Seite 1735)

BitResultInfo-Eigenschaft

Beschreibung

Gibt das BitResultInfo-Objekt zurück.

Beispiel

Ein Beispiel zur Anwendung der BitResultInfo-Eigenschaft finden Sie in dieser Dokumentation unter "BitResultInfo-Objekt".

Siehe auch

BitResultInfo-Objekt (Seite 1897)

BitSetValue-Eigenschaft

Beschreibung

Legt den Wert für die dynamisierte Eigenschaft fest, wenn das angegebene Bit einer projektierten Variable gesetzt ist oder gibt ihn zurück.

Welches Bit gesetzt werden muss, um die Wertänderung auszulösen, legen Sie mit der Eigenschaft BitNumber fest.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert, ein Variablenname vergeben sowie das zu setzende Bit definiert und den Zuständen "gesetzt"/"nicht gesetzt" die dazugehörigen Eigenschaftswerte zugeordnet:

```
Sub AddDynamicDialogToCircleRadiusTypeBit()
'VBA412
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_B", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeBit
.BitResultInfo.BitNumber = 1
.BitResultInfo.BitSetValue = 40
.BitResultInfo.BitNotSetValue = 80
End With
End Sub
```

Siehe auch

BitNumber-Eigenschaft (Seite 2104)

BitResultInfo-Objekt (Seite 1897)

Bold-Eigenschaft

Beschreibung

TRUE, wenn das Schriftattribut "Fett" für den sprachabhängigen Text im Objekt gesetzt ist. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Hinweis

Sie müssen in den Sprachen projektiert haben, damit das Beispiel funktioniert.

Im folgenden Beispiel werden für einen Button die Schriftattribute für Französisch und Englisch gesetzt:

```
Sub ExampleForLanguageFonts()
'VBA413
Dim colLangFonts As HMILanguageFonts
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
objButton.Text = "Displaytext"
Set colLangFonts = objButton.LDFonts
'Set french fontproperties:
With colLangFonts.ItemByLCID(1036)
.Family = "Courier New"
.Bold = True
.Italic = False
.Underlined = True
.Size = 12
End With
'Set english fontproperties:
With colLangFonts.ItemByLCID(1033)
.Family = "Times New Roman"
.Bold = False
.Italic = True
.Underlined = False
.Size = 14
End With
End Sub
```

Siehe auch

Underlined-Eigenschaft (Seite 2402)

Size-Eigenschaft (Seite 2368)

Parent-Eigenschaft (Seite 2317)

LanguageID-Eigenschaft (Seite 2232)

Italic-Eigenschaft (Seite 2224)

FontFamily-Eigenschaft (Seite 2201)

Application-Eigenschaft (Seite 2079)

LanguageFont-Objekt (Seite 1962)

BorderBackColor-Eigenschaft

Beschreibung

Legt die Hintergrundfarbe der Linie für das Objekt fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Die Hintergrundfarbe ist nur sichtbar, wenn die Eigenschaft BorderStyle > 0 gesetzt ist.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel wird die Hintergrundfarbe der Linie auf "Gelb" gesetzt:

```
Sub RectangleConfiguration()
'VBA415
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.BorderBackColor = RGB(255, 255, 0)
End With
End Sub
```

Siehe auch

PieSegment-Objekt (Seite 1995)
BorderStyle-Eigenschaft (Seite 2115)
TextList-Objekt (Seite 2037)
StatusDisplay-Objekt (Seite 2032)
StaticText-Objekt (Seite 2029)
Slider-Objekt (Seite 2025)
RoundRectangle-Objekt (Seite 2018)
RoundButton-Objekt (Seite 2015)
Rectangle-Objekt (Seite 2012)
PolyLine-Objekt (Seite 2001)
OptionGroup-Objekt (Seite 1989)
Line-Objekt (Seite 1970)
IOField-Objekt (Seite 1959)
GraphicObject-Objekt (Seite 1943)
EllipseArc-Objekt (Seite 1929)
EllipseSegment-Objekt (Seite 1932)
Ellipse-Objekt (Seite 1926)
CircularArc-Objekt (Seite 1905)
Circle-Objekt (Seite 1902)
CheckBox-Objekt (Seite 1901)
Button-Objekt (Seite 1898)
BarGraph-Objekt (Seite 1893)

BorderColor-Eigenschaft

Beschreibung

Legt die Linienfarbe für das Objekt fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel wird die Linienfarbe auf "Blau" gesetzt:

```
Sub RectangleConfiguration()
'VBA416
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.BorderColor = RGB(0, 0, 255)
End With
End Sub
```

Siehe auch

GraphicObject-Objekt (Seite 1943)

TextList-Objekt (Seite 2037)

StatusDisplay-Objekt (Seite 2032)

StaticText-Objekt (Seite 2029)

Slider-Objekt (Seite 2025)

RoundRectangle-Objekt (Seite 2018)

RoundButton-Objekt (Seite 2015)

Rectangle-Objekt (Seite 2012)

PolyLine-Objekt (Seite 2001)

PieSegment-Objekt (Seite 1995)

OptionGroup-Objekt (Seite 1989)

Line-Objekt (Seite 1970)

IOField-Objekt (Seite 1959)

EllipseArc-Objekt (Seite 1929)

EllipseSegment-Objekt (Seite 1932)

Ellipse-Objekt (Seite 1926)

CircularArc-Objekt (Seite 1905)

Circle-Objekt (Seite 1902)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

BarGraph-Objekt (Seite 1893)

BorderColorBottom-Eigenschaft

Beschreibung

Legt die Farbe für den rechten und den unteren Teil des 3D-Rahmens fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel wird die 3D-Rahmenfarbe festgelegt:

```
Sub ButtonConfiguration()
'VBA417
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.BorderColorBottom = RGB(255, 0, 0)
.BorderColorTop = RGB(0, 0, 255)
End With
End Sub
```

Siehe auch

RoundButton-Objekt (Seite 2015)

Button-Objekt (Seite 1898)

BorderColorTop-Eigenschaft

Beschreibung

Legt die Farbe für den linken und den oberen Teil des 3D-Rahmens fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel wird die 3D-Rahmenfarbe festgelegt:

```
Sub ButtonConfiguration()
'VBA418
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.BorderColorBottom = RGB(255, 0, 0)
.BorderColorTop = RGB(0, 0, 255)
End With
End Sub
```

Siehe auch

RoundButton-Objekt (Seite 2015)

Button-Objekt (Seite 1898)

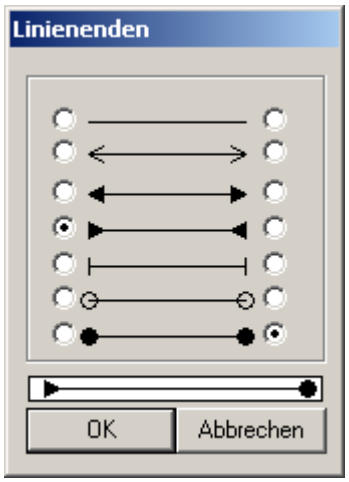
BorderEndStyle-Eigenschaft

Beschreibung

Legt die Art der Linienenden des Objektes fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff

Ermittlung der Linienenden

Die Art der Linienenden ermitteln Sie anhand eines fünfstelligen Hexadezimalwertes, den Sie dann in seinen Dezimalwert umrechnen.



Ausgehend vom Fenster "Linienenden" ermitteln Sie die Linienenden für das Objekt wie folgt:

- Linke Spalte: Konfiguriert den Linienanfang. Wertebereich (von oben nach unten) von 0 bis 6. Der Linienanfang entspricht der ersten Stelle im Hexadezimalwert. In der abgebildeten Konfiguration hat die erste Stelle den Wert "3".
- Rechte Spalte: Konfiguriert das Linienende. Wertebereich (von oben nach unten) von 0 bis 6. Das Linienende entspricht der fünften Stelle im Hexadezimalwert. In der abgebildeten Konfiguration hat die fünfte Stelle den Wert "6"

Daraus ergibt sich ein Hexadezimalwert von "60003". Dies entspricht einem Dezimalwert von "393219", den Sie der Eigenschaft BorderEndStyle zuweisen.

Beispiel

Die Prozedur "LineConfiguration()" greift auf die Eigenschaften der Linie zu. In diesem Beispiel wird die Art der Linienende auf die oben abgebildete Konfiguration gesetzt:

```
Sub LineConfiguration()
'VBA419
Dim objLine As HMILine
Set objLine = ActiveDocument.HMIObjects.AddHMIObject("Line1", "HMILine")
With objLine
.BorderEndStyle = 393219
End With
End Sub
```

Siehe auch

PolyLine-Objekt (Seite 2001)

Line-Objekt (Seite 1970)

BorderFlashColorOff-Eigenschaft

Beschreibung

Legt die Linienfarbe des Objektes für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel wird die Farbe für den Blinkzustand "Aus" auf "Schwarz" gesetzt:

```
Sub RectangleConfiguration()
'VBA420
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.BorderFlashColorOff = RGB(0, 0, 0)
End With
End Sub
```

Siehe auch

RoundButton-Objekt (Seite 2015)

StatusDisplay-Objekt (Seite 2032)

StaticText-Objekt (Seite 2029)

Slider-Objekt (Seite 2025)

TextList-Objekt (Seite 2037)

RoundRectangle-Objekt (Seite 2018)

Rectangle-Objekt (Seite 2012)

PolyLine-Objekt (Seite 2001)

Polygon-Objekt (Seite 1998)

PieSegment-Objekt (Seite 1995)

OptionGroup-Objekt (Seite 1989)

Line-Objekt (Seite 1970)

GraphicObject-Objekt (Seite 1943)

IOField-Objekt (Seite 1959)

EllipseSegment-Objekt (Seite 1932)

EllipseArc-Objekt (Seite 1929)

Ellipse-Objekt (Seite 1926)

CircularArc-Objekt (Seite 1905)

Circle-Objekt (Seite 1902)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

BorderFlashColorOn-Eigenschaft

Beschreibung

Legt die Linienfarbe des Objektes für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG-Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel wird die Farbe für den Blinkzustand "Ein" auf "Rot" gesetzt:

```
Sub RectangleConfiguration()
'VBA421
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.BorderFlashColorOn = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

StaticText-Objekt (Seite 2029)
StatusDisplay-Objekt (Seite 2032)
Slider-Objekt (Seite 2025)
TextList-Objekt (Seite 2037)
RoundRectangle-Objekt (Seite 2018)
RoundButton-Objekt (Seite 2015)
Rectangle-Objekt (Seite 2012)
PolyLine-Objekt (Seite 2001)
Polygon-Objekt (Seite 1998)
PieSegment-Objekt (Seite 1995)
OptionGroup-Objekt (Seite 1989)
Line-Objekt (Seite 1970)
GraphicObject-Objekt (Seite 1943)
IOField-Objekt (Seite 1959)
EllipseSegment-Objekt (Seite 1932)
EllipseArc-Objekt (Seite 1929)
Ellipse-Objekt (Seite 1926)
CircularArc-Objekt (Seite 1905)
Circle-Objekt (Seite 1902)
CheckBox-Objekt (Seite 1901)
Button-Objekt (Seite 1898)

BorderStyle-Eigenschaft

Beschreibung

Legt die Linienart für das Objekt fest oder gibt sie zurück. Wertebereich von 0 bis 4:

Linienart	zugeordneter Wert
———	0
— —	1
- - - - - -	2
— · —	3
— · · —	4

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel wird die Linienart auf "1" gesetzt:

```
Sub RectangleConfiguration()
'VBA422
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.BorderStyle = 1
End With
End Sub
```

Siehe auch

IOField-Objekt (Seite 1959)

StatusDisplay-Objekt (Seite 2032)

StaticText-Objekt (Seite 2029)

Slider-Objekt (Seite 2025)

RoundRectangle-Objekt (Seite 2018)

RoundButton-Objekt (Seite 2015)

Rectangle-Objekt (Seite 2012)

Polygon-Objekt (Seite 1998)

PolyLine-Objekt (Seite 2001)

TextList-Objekt (Seite 2037)

PieSegment-Objekt (Seite 1995)

OptionGroup-Objekt (Seite 1989)

Line-Objekt (Seite 1970)

GraphicObject-Objekt (Seite 1943)

EllipseSegment-Objekt (Seite 1932)

EllipseArc-Objekt (Seite 1929)

Ellipse-Objekt (Seite 1926)

CircularArc-Objekt (Seite 1905)

Circle-Objekt (Seite 1902)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

BarGraph-Objekt (Seite 1893)

3DBarGraph-Objekt (Seite 1879)

BorderWidth-Eigenschaft

Beschreibung

Legt die Linienstärke (in Pixel) für das Objekt fest oder gibt sie zurück.

Beispiel

Im fogenden Beispiel wird die Linienstärke eines neu hinzugefügten Kreises auf "2" gesetzt.

```
Sub CircleConfiguration()
'VBA423
Dim objCircle As IHMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle1", "HMICircle")
With objCircle
.BorderWidth = 2
End With
End Sub
```

Siehe auch

IOField-Objekt (Seite 1959)
StatusDisplay-Objekt (Seite 2032)
StaticText-Objekt (Seite 2029)
Slider-Objekt (Seite 2025)
TextList-Objekt (Seite 2037)
RoundRectangle-Objekt (Seite 2018)
RoundButton-Objekt (Seite 2015)
Rectangle-Objekt (Seite 2012)
PolyLine-Objekt (Seite 2001)
Polygon-Objekt (Seite 1998)
PieSegment-Objekt (Seite 1995)
OptionGroup-Objekt (Seite 1989)
Line-Objekt (Seite 1970)
GraphicObject-Objekt (Seite 1943)
EllipseSegment-Objekt (Seite 1932)
EllipseArc-Objekt (Seite 1929)
Ellipse-Objekt (Seite 1926)
CircularArc-Objekt (Seite 1905)
Circle-Objekt (Seite 1902)
CheckBox-Objekt (Seite 1901)
Button-Objekt (Seite 1898)

BottomConnectedObjectName-Eigenschaft

Beschreibung

Gibt den Namen des Start-Objektes zurück, das der Verbinder verbindet. Lese-Zugriff.

Beispiel:

Ein Beispiel für die Anwendung der BottomConnectedObjectName-Eigenschaft finden Sie in dieser Dokumentation unter "objConnection-Objekt".

Siehe auch

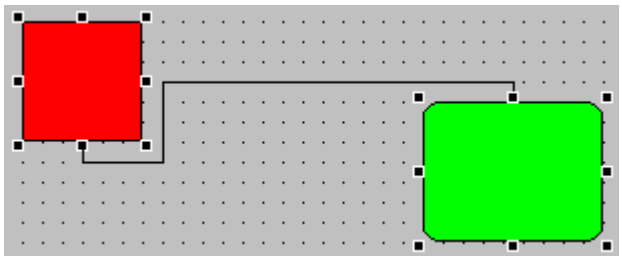
objConnection-Objekt (Seite 1985)

BottomConnectedConnectionPointIndex-Eigenschaft

Beschreibung

Gibt die Anschlussstelle am Objekt zurück, mit dem der Verbinder verbunden ist.

Anschlussstelle	zugeordneter Wert
Oben	0
Rechts	1
Unten	2
Links	3



Beispiel

Ein Beispiel für die Anwendung der BottomConnectedObjectName-Eigenschaft finden Sie in dieser Dokumentation unter "objConnection-Objekt".

Siehe auch

objConnection-Objekt (Seite 1985)

BoxAlignment-Eigenschaft

Beschreibung

TRUE, wenn die Felder rechtsbündig angeordnet sind. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "CreateOptionGroup()" erzeugt das Objekt OptionGroup mit vier Optionsfeldern. Jedes Optionsfeld wird mit dem Standardnamen "myCustomText<Nummer>"belegt:

```
Sub CreateOptionGroup()
'VBA424
Dim objRadioBox As HMIOptionGroup
Dim iCounter As Integer
Set objRadioBox = ActiveDocument.HMIObjects.AddHMIObject("RadioBox_1", "HMIOptionGroup")
```

```
iCounter = 1
With objRadioBox
.Height = 100
.Width = 180
.BoxCount = 4
.BoxAlignment = False
For iCounter = 1 To .BoxCount
.index = iCounter
.Text = "CustomText" & .index
Next iCounter
End With
End Sub
```

Siehe auch

BoxCount-Eigenschaft (Seite 2120)

OptionGroup-Objekt (Seite 1989)

CheckBox-Objekt (Seite 1901)

BoxCount-Eigenschaft

Beschreibung

Legt die Anzahl der Felder fest oder gibt sie zurück. Wertebereich von 1 bis 32.

Beispiel

Die Prozedur "CreateOptionGroup()" erzeugt das Objekt OptionGroup mit vier Optionsfeldern. Jedes Optionsfeld wird mit dem Standardnamen "myCustomText<Nummer>"belegt:

```
Sub CreateOptionGroup()
'VBA425
Dim objRadioBox As HMIOptionGroup
Dim iCounter As Integer
Set objRadioBox = ActiveDocument.HMIObjects.AddHMIObject("RadioBox_1", "HMIOptionGroup")
iCounter = 1
With objRadioBox
.Height = 100
.Width = 180
.BoxCount = 4
.BoxAlignment = True
For iCounter = 1 To .BoxCount
.index = iCounter
.Text = "CustomText" & .index
Next iCounter
End With
End Sub
```

Siehe auch

BoxAlignment-Eigenschaft (Seite 2119)

OptionGroup-Objekt (Seite 1989)

CheckBox-Objekt (Seite 1901)

BoxType-Eigenschaft

Beschreibung

Legt den Feldtyp fest oder gibt ihn zurück. Wertebereich von 0 bis 2.

Feldtyp	zugeordneter Wert
Ausgabe	0
Eingabe	1
E/A-Feld	2

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel wird als Feldtyp "Eingabe" projektiert:

```
Sub IOFieldConfiguration()
'VBA426
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.BoxType = 1
End With
End Sub
```

Siehe auch

IOField-Objekt (Seite 1959)

Button1Width-Eigenschaft

Beschreibung

Legt für das Objekt Sammelanzeige die Breite des Buttons 1 in Pixel fest oder gibt sie zurück.

Wenn die Eigenschaft SameSize auf TRUE gesetzt ist, erhalten alle Buttons dieselbe Breite.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel wird die Breite von Button 1 auf "50" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA427
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.Button1Width = 50
End With
End Sub
```

Siehe auch

SameSize-Eigenschaft (Seite 2353)

Button4Width-Eigenschaft (Seite 2123)

Button3Width-Eigenschaft (Seite 2123)

Button2Width-Eigenschaft (Seite 2122)

GroupDisplay-Objekt (Seite 1947)

Button2Width-Eigenschaft

Beschreibung

Legt für das Objekt Sammelanzeige die Breite des Buttons 2 in Pixel fest oder gibt sie zurück.

Wenn die Eigenschaft SameSize auf TRUE gesetzt ist, erhalten alle Buttons dieselbe Breite.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel wird die Breite von Button 2 auf "50" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA428
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.Button2Width = 50
End With
End Sub
```

Siehe auch

SameSize-Eigenschaft (Seite 2353)

Button4Width-Eigenschaft (Seite 2123)

Button3Width-Eigenschaft (Seite 2123)

Button1Width-Eigenschaft (Seite 2121)

GroupDisplay-Objekt (Seite 1947)

Button3Width-Eigenschaft

Beschreibung

Legt für das Objekt Sammelanzeige die Breite des Buttons 3 in Pixel fest oder gibt sie zurück.

Wenn die Eigenschaft SameSize auf TRUE gesetzt ist, erhalten alle Buttons dieselbe Breite.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel wird die Breite von Button 3 auf "50" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA429
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.Button3Width = 50
End With
End Sub
```

Siehe auch

SameSize-Eigenschaft (Seite 2353)

Button4Width-Eigenschaft (Seite 2123)

Button2Width-Eigenschaft (Seite 2122)

Button1Width-Eigenschaft (Seite 2121)

GroupDisplay-Objekt (Seite 1947)

Button4Width-Eigenschaft

Beschreibung

Legt für das Objekt Sammelanzeige die Breite des Buttons 4 in Pixel fest oder gibt sie zurück.

Wenn die Eigenschaft SameSize auf TRUE gesetzt ist, erhalten alle Buttons dieselbe Breite.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel wird die Breite von Button 4 auf "50" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA430
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.Button4Width = 50
End With
End Sub
```

Siehe auch

Button1Width-Eigenschaft (Seite 2121)

SameSize-Eigenschaft (Seite 2353)

Button3Width-Eigenschaft (Seite 2123)

Button2Width-Eigenschaft (Seite 2122)

GroupDisplay-Objekt (Seite 1947)

ButtonColor-Eigenschaft

Beschreibung

Legt für das Objekt Slider die Farbe des Schiebers fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "SliderConfiguration()" greift auf die Eigenschaften des Sliders zu. In diesem Beispiel wird die Farbe des Schiebers auf "Gelb" gesetzt:

```
Sub SliderConfiguration()
```

```
'VBA431
Dim objSlider As HMISlider
Set objSlider = ActiveDocument.HMIObjects.AddHMIObject("SliderObject1", "HMISlider")
With objSlider
.ButtonColor = RGB(255, 255, 0)
End With
End Sub
```

Siehe auch

Slider-Objekt (Seite 2025)

C

Caption-Eigenschaft

Beschreibung

TRUE, wenn das Applikations- oder Bildfenster in Runtime eine Titelleiste hat. BOOLEAN Schreib-Lese-Zugriff.

Die Caption-Eigenschaft muss auf "True" gesetzt sein, wenn das Applikations- oder Bildfenster Schaltflächen zum Maximieren und Schließen besitzen soll.

Beispiel

Die Prozedur "ApplicationWindowConfig" greift auf die Eigenschaften des Applikationsfensters zu. In diesem Beispiel wird das Applikationsfenster konfiguriert:

```
Sub ApplicationWindowConfig()
'VBA432
Dim objAppWindow As HMIApplicationWindow
Set objAppWindow = ActiveDocument.HMIObjects.AddHMIObject("AppWindow",
"HMIApplicationWindow")
With objAppWindow
.Caption = True
.CloseButton = False
.Height = 200
.Left = 10
.MaximizeButton = True
.Moveable = False
.OnTop = True
.Sizeable = True
.Top = 20
.Visible = True
.Width = 250
.WindowBorder = True
End With
```

```
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

ApplicationWindow-Objekt (Seite 1891)

CaptionText-Eigenschaft

Beschreibung

Legt für das PictureWindow-Objekt den Fenstertitel fest, der in Runtime angezeigt wird oder gibt ihn zurück.

Die Caption-Eigenschaft muss dazu auf "True" gesetzt sein.

Beispiel

Die Prozedur "PictureWindowConfig" greift auf die Eigenschaften des Bildfensters zu. In diesem Beispiel wird das Bildfenster konfiguriert:

```
Sub PictureWindowConfig()
'VBA433
Dim objPicWindow As HMIPictureWindow
Set objPicWindow = ActiveDocument.HMIObjects.AddHMIObject("PicWindow1", "HMIPictureWindow")
With objPicWindow
.AdaptPicture = False
.AdaptSize = False
.Caption = True
.CaptionText = "Picturewindow in runtime"
.OffsetLeft = 5
.OffsetTop = 10
'Replace the picturename "Test.PDL" with the name of
'an existing document from your "GraCS"-Folder of your active project
.PictureName = "Test.PDL"
.ScrollBars = True
.ServerPrefix = ""
.TagPrefix = "Struct."
.UpdateCycle = 5
.Zoom = 100
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

CheckAlarmHigh-Eigenschaft

Beschreibung

TRUE, wenn der Grenzwert "Alarm high" für das BarGraph-Objekt überwacht wird. BOOLEAN Schreib-Lese-Zugriff.

Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften AlarmHigh, ColorAlarmHigh und TypeAlarmHigh festgelegt.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "50" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA434
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeAlarmHigh = False
'Activate monitoring
.CheckAlarmHigh = True
'Set barcolor to "yellow"
.ColorAlarmHigh = RGB(255, 255, 0)
'Set upper limit to "50"
.AlarmHigh = 50
End With
End Sub
```

Siehe auch

TypeAlarmHigh-Eigenschaft (Seite 2393)

ColorAlarmHigh-Eigenschaft (Seite 2138)

AlarmHigh-Eigenschaft (Seite 2073)

BarGraph-Objekt (Seite 1893)

CheckAlarmLow-Eigenschaft

Beschreibung

TRUE, wenn der Grenzwert "Alarm low" für das BarGraph-Objekt überwacht wird. BOOLEAN Schreib-Lese-Zugriff.

Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften AlarmLow, ColorAlarmLow und TypeAlarmLow festgelegt.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "10" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA435
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeAlarmLow = False
'Activate monitoring
.CheckAlarmLow = True
'Set barcolor to "yellow"
.ColorAlarmLow = RGB (255, 255, 0)
'Set lower limit to "10"
.AlarmLow = 10
End With
End Sub
```

Siehe auch

ColorAlarmLow-Eigenschaft (Seite 2139)

TypeAlarmLow-Eigenschaft (Seite 2394)

AlarmLow-Eigenschaft (Seite 2074)

BarGraph-Objekt (Seite 1893)

Checked-Eigenschaft

Beschreibung

TRUE, wenn vor dem benutzerdefinierten Menüeintrag ein Häkchen erscheinen soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "CreateMenuItem()" legt das Menü "Objekte löschen" an und fügt zwei Menüeinträge ("Rechtecke löschen" und "Kreise löschen") hinzu. Der erste Menüeintrag wird zusätzlich mit einem Häkchen versehen:

```
Sub CreateMenuItem()
'VBA436
Dim objMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
'
```

```
'Add new menu "Delete objects" to menubar:
Set objMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DeleteObjects", "Delete objects")
'
'Add two menuitems to the new menu
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(1, "DeleteAllRectangles", "Delete
Rectangles")
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(2, "DeleteAllCircles", "Delete Circles")
With objMenu.MenuItems
.Item("DeleteAllRectangles").Checked = True
End With
End Sub
```

Siehe auch

MenuItems-Eigenschaft (Seite 2298)

Menüs und Symbolleisten konfigurieren (Seite 1628)

CheckLimitHigh4-Eigenschaft

Beschreibung

TRUE, wenn der obere Grenzwert "Reserve 4" des BarGraph-Objektes überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften LimitHigh4, ColorLimitHigh4 und TypeLimitHigh4 festgelegt.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "70" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA437
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeLimitHigh4 = False
'Activate monitoring
.CheckLimitHigh4 = True
'set barcolor to "red"
.ColorLimitHigh4 = RGB(255, 0, 0)
'Set upper limit to "70"
.LimitHigh4 = 70
End With
End Sub
```

Siehe auch

TypeLimitHigh4-Eigenschaft (Seite 2394)

LimitHigh4-Eigenschaft (Seite 2269)

ColorLimitHigh4-Eigenschaft (Seite 2141)

BarGraph-Objekt (Seite 1893)

CheckLimitHigh5-Eigenschaft

Beschreibung

TRUE, wenn der obere Grenzwert "Reserve 5" des BarGraph-Objektes überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften LimitHigh5, ColorLimitHigh5 und TypeLimitHigh5 festgelegt.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "80" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA438
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeLimitHigh5 = False
'Activate monitoring
.CheckLimitHigh5 = True
'set barcolor to "black"
.ColorLimitHigh5 = RGB(0, 0, 0)
'Set upper limit to "80"
.LimitHigh5 = 80
End With
End Sub
```

Siehe auch

ColorLimitHigh5-Eigenschaft (Seite 2142)

TypeLimitHigh5-Eigenschaft (Seite 2395)

LimitHigh4-Eigenschaft (Seite 2269)

BarGraph-Objekt (Seite 1893)

CheckLimitLow4-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Reserve 4" des BarGraph-Objektes überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften LimitLow4, ColorLimitLow4 und TypeLimitLow4 festgelegt.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "5" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA439
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeLimitLow4 = False
'Activate monitoring
.CheckLimitLow4 = True
'Set barcolor to "green"
.ColorLimitLow4 = RGB(0, 255, 0)
'set lower limit to "5"
.LimitLow4 = 5
End With
End Sub
```

Siehe auch

TypeLimitLow4-Eigenschaft (Seite 2396)

LimitLow4-Eigenschaft (Seite 2270)

ColorLimitLow4-Eigenschaft (Seite 2143)

BarGraph-Objekt (Seite 1893)

CheckLimitLow5-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Reserve 5" des BarGraph-Objektes überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften LimitLow5, ColorLimitLow5 und TypeLimitLow5 festgelegt.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "0" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA440
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeLimitLow5 = False
'Activate monitoring
.CheckLimitLow5 = True
'Set barcolor to "white"
.ColorLimitLow5 = RGB(255, 255, 255)
'set lower limit to "0"
.LimitLow5 = 0
End With
End Sub
```

Siehe auch

TypeLimitLow5-Eigenschaft (Seite 2397)

LimitLow5-Eigenschaft (Seite 2271)

ColorLimitLow5-Eigenschaft (Seite 2144)

BarGraph-Objekt (Seite 1893)

CheckToleranceHigh-Eigenschaft

Beschreibung

TRUE, wenn der Grenzwert "Tolerance high" für das BarGraph-Objekt überwacht wird. BOOLEAN Schreib-Lese-Zugriff.

Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften ToleranceHigh, ColorToleranceHigh und TypeToleranceHigh festgelegt.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "45" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
```

```
'VBA441
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeToleranceHigh = False
'Activate monitoring
.CheckToleranceHigh = True
'Set barcolor to "yellow"
.ColorToleranceHigh = RGB(255, 255, 0)
'Set upper limit to "45"
.ToleranceHigh = 45
End With
End Sub
```

Siehe auch

TypeToleranceHigh-Eigenschaft (Seite 2398)

ToleranceHigh-Eigenschaft (Seite 2385)

ColorToleranceHigh-Eigenschaft (Seite 2145)

BarGraph-Objekt (Seite 1893)

CheckToleranceLow-Eigenschaft

Beschreibung

TRUE, wenn der Grenzwert "Tolerance low" für das BarGraph-Objekt überwacht wird. BOOLEAN Schreib-Lese-Zugriff.

Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften ToleranceLow, ColorToleranceLow und TypeToleranceLow festgelegt.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "15" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA442
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeToleranceLow = False
'Activate monitoring
.CheckToleranceLow = True
```

```
'Set barcolor to "yellow"
.ColorToleranceLow = RGB(255, 255, 0)
'Set lower limit to "15"
.ToleranceLow = 15
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

TypeToleranceLow-Eigenschaft (Seite 2398)

ToleranceLow-Eigenschaft (Seite 2386)

ColorToleranceLow-Eigenschaft (Seite 2146)

CheckWarningHigh-Eigenschaft

Beschreibung

TRUE, wenn der Grenzwert "Warning high" für das BarGraph-Objekt überwacht wird. BOOLEAN Schreib-Lese-Zugriff.

Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften WarningHigh, ColorWarningHigh und TypeWarningHigh festgelegt.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "75" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA443
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeWarningHigh = False
'Activate monitoring
.CheckWarningHigh = True
'Set barcolor to "red"
.ColorWarningHigh = RGB(255, 0, 0)
'Set upper limit to "75"
.WarningHigh = 75
End With
End Sub
```

Siehe auch

WarningHigh-Eigenschaft (Seite 2484)

TypeWarningHigh-Eigenschaft (Seite 2399)

ColorWarningHigh-Eigenschaft (Seite 2147)

BarGraph-Objekt (Seite 1893)

CheckWarningLow-Eigenschaft

Beschreibung

TRUE, wenn der Grenzwert "Warning low" für das BarGraph-Objekt überwacht wird. BOOLEAN Schreib-Lese-Zugriff.

Der Grenzwert, die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften WarningLow, ColorWarningLow und TypeWarningLow festgelegt.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "12" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA444
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeWarningLow = False
'Activate monitoring
.CheckWarningLow = True
'Set barcolor to "magenta"
.ColorWarningLow = RGB(255, 0, 255)
'Set lower limit to "12"
.WarningLow = 12
End With
End Sub
```

Siehe auch

WarningLow-Eigenschaft (Seite 2485)

TypeWarningLow-Eigenschaft (Seite 2400)

ColorWarningLow-Eigenschaft (Seite 2148)

BarGraph-Objekt (Seite 1893)

ClearOnError-Eigenschaft

Beschreibung

TRUE, wenn der Feldeintrag des EA-Feldes bei einer Fehleingabe automatisch gelöscht wird. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel soll das EA-Feld bei Fehleingabe gelöscht werden:

```
Sub IOFieldConfiguration()
'VBA445
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.ClearOnError = True
End With
End Sub
```

Siehe auch

IOField-Objekt (Seite 1959)

ClearOnNew-Eigenschaft

Beschreibung

TRUE, wenn der Feldeintrag des EA-Feldes gelöscht wird, sobald das EA-Feld den Fokus erhält. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel wird der Feldeintrag des EA-Feld gelöscht, sobald es den Fokus erhält:.

```
Sub IOFieldConfiguration()
'VBA446
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.ClearOnNew = True
End With
End Sub
```

Siehe auch

IOField-Objekt (Seite 1959)

CloseButton-Eigenschaft

Beschreibung

TRUE, wenn die Objekte ApplicationWindow und PictureWindow in Runtime eine "Schließen"-Schaltfläche besitzen. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "ApplicationWindowConfig" greift auf die Eigenschaften des Applikationsfensters zu. In diesem Beispiel soll das Applikationsfenster in Runtime eine "Schließen"-Schaltfläche besitzen:

```
Sub ApplicationWindowConfig()
'VBA447
Dim objAppWindow As HMIApplicationWindow
Set objAppWindow = ActiveDocument.HMIObjects.AddHMIObject("AppWindow1",
"HMIApplicationWindow")
With objAppWindow
.CloseButton = True
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

ApplicationWindow-Objekt (Seite 1891)

CollectValue-Eigenschaft

Beschreibung

Beinhaltet in Runtime als Eingangswert den jeweils aktuellen Zustand der aktiven Meldeklassen. LONG Schreib-Lese-Zugriff.

Durch Dynamisierung über eine Variable kann beispielsweise der Wert aus den Sammelanzeigen hierarchisch untergeordneter Bilder ermittelt werden.

Beispiel

--

Siehe auch

GroupDisplay-Objekt (Seite 1947)

ColorAlarmHigh-Eigenschaft

Beschreibung

Legt die Balkenfarbe für den Grenzwert "Alarm high" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Die Eigenschaft "CheckAlarmHigh" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "50" ausgelöst werden und die Balkenfarbe auf "rot" wechseln.

```
Sub BarGraphLimitConfiguration()
'VBA449
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeAlarmHigh = False
'Activate monitoring
.CheckAlarmHigh = True
'Set barcolor to "red"
.ColorAlarmHigh = RGB(255, 0, 0)
'Set upper limit to "50"
.AlarmHigh = 50
End With
End Sub
```

Siehe auch

CheckAlarmHigh-Eigenschaft (Seite 2127)

BarGraph-Objekt (Seite 1893)

ColorAlarmLow-Eigenschaft

Beschreibung

Legt die Balkenfarbe für den Grenzwert "Alarm low" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Die Eigenschaft "CheckAlarmLow" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "10" ausgelöst werden und die Balkenfarbe auf "rot" wechseln.

```
Sub BarGraphLimitConfiguration()
'VBA450
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeAlarmLow = False
'Activate monitoring
.CheckAlarmLow = True
'Set barcolor to "red"
.ColorAlarmLow = RGB(255, 0, 0)
'Set lower limit to "10"
.AlarmLow = 10
End With
End Sub
```

Siehe auch

CheckAlarmLow-Eigenschaft (Seite 2127)

BarGraph-Objekt (Seite 1893)

ColorBottom-Eigenschaft

Beschreibung

Legt die Farbe für den unteren/rechten Anschlag des Slider-Objektes fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "SliderConfiguration()" greift auf die Eigenschaften des Sliders zu. In diesem Beispiel wird die Farbe für den unteren/rechten Anschlag auf "Rot" gesetzt:

```
Sub SliderConfiguration()
'VBA451
Dim objSlider As HMISlider
Set objSlider = ActiveDocument.HMIObjects.AddHMIObject("SliderObject1", "HMISlider")
With objSlider
.ColorBottom = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

Slider-Objekt (Seite 2025)

ColorChangeType-Eigenschaft

Beschreibung

TRUE, wenn bei einer Farbänderung beim BarGraph-Objekt (z.B. beim Erreichen eines Grenzwerts) der Farbumschlag segmentweise erfolgen soll. Bei FALSE gilt der Farbumschlag für den gesamten Balken. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel soll der Farbumschlag für den gesamten Balken erfolgen:

```
Sub BarGraphLimitConfiguration()
```

```
'VBA452
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.ColorChangeType = False
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

ColorLimitHigh4-Eigenschaft

Beschreibung

Legt die Farbe für den oberen Grenzwert "Reserve 4" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Die Eigenschaft "CheckLimitHigh4" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "70" ausgelöst werden und die Balkenfarbe auf "rot" wechseln.

```
Sub BarGraphLimitConfiguration()
'VBA453
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeLimitHigh4 = False
'Activate monitoring
.CheckLimitHigh4 = True
'Set barcolor to "red"
.ColorLimitHigh4 = RGB(255, 0, 0)
'Set upper limit to "70"
.LimitHigh4 = 70
```

```
End With
End Sub
```

Siehe auch

CheckLimitHigh4-Eigenschaft (Seite 2129)

BarGraph-Objekt (Seite 1893)

ColorLimitHigh5-Eigenschaft

Beschreibung

Legt die Farbe für den oberen Grenzwert "Reserve 5" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Die Eigenschaft "CheckLimitHigh5" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "80" ausgelöst werden und die Balkenfarbe auf "schwarz" wechseln.

```
Sub BarGraphLimitConfiguration()
'VBA454
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeLimitHigh5 = False
'Activate monitoring
.CheckLimitHigh5 = True
'Set barcolor to "black"
.ColorLimitHigh5 = RGB(0, 0, 0)
'Set upper limit to "80"
.LimitHigh5 = 80
End With
End Sub
```

Siehe auch

CheckLimitHigh5-Eigenschaft (Seite 2130)

BarGraph-Objekt (Seite 1893)

ColorLimitLow4-Eigenschaft

Beschreibung

Legt die Farbe für den unteren Grenzwert "Reserve 4" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Die Eigenschaft "CheckLimitLow4" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "5" ausgelöst werden und die Balkenfarbe auf "grün" wechseln.

```
Sub BarGraphLimitConfiguration()
'VBA455
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeLimitLow4 = False
'Activate monitoring
.CheckLimitLow4 = True
'Set barcolor to "green"
.ColorLimitLow4 = RGB(0, 255, 0)
'Set lower limit to "5"
.LimitLow4 = 5
End With
End Sub
```

Siehe auch

CheckLimitLow4-Eigenschaft (Seite 2131)

BarGraph-Objekt (Seite 1893)

ColorLimitLow5-Eigenschaft

Beschreibung

Legt die Farbe für den unteren Grenzwert "Reserve 5" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Die Eigenschaft "CheckLimitLow5" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "0" ausgelöst werden und die Balkenfarbe auf "weiß" wechseln.

```
Sub BarGraphLimitConfiguration()
'VBA456
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeLimitLow5 = False
'Activate monitoring
.CheckLimitLow5 = True
'Set barcolor to "white"
.ColorLimitLow5 = RGB(255, 255, 255)
'Set lower limit to "0"
.LimitLow5 = 0
End With
End Sub
```

Siehe auch

CheckLimitLow5-Eigenschaft (Seite 2131)

BarGraph-Objekt (Seite 1893)

ColorToleranceHigh-Eigenschaft

Beschreibung

Legt die Farbe für den oberen Grenzwert "Tolerance high" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Die Eigenschaft "CheckToleranceHigh" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "45" ausgelöst werden und die Balken auf "gelb" wechseln

```
Sub BarGraphLimitConfiguration()
'VBA457
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeToleranceHigh = False
'Activate monitoring
.CheckToleranceHigh = True
'Set barcolor to "yellow"
.ColorToleranceHigh = RGB(255, 255, 0)
'Set upper limit to "45"
.ToleranceHigh = 45
End With
End Sub
```

Siehe auch

CheckToleranceHigh-Eigenschaft (Seite 2132)

BarGraph-Objekt (Seite 1893)

ColorToleranceLow-Eigenschaft

Beschreibung

Legt die Farbe für den unteren Grenzwert "Tolerance low" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Die Eigenschaft "CheckToleranceLow" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "12" ausgelöst werden und die Balkenfarbe auf "gelb" wechseln:

```
Sub BarGraphLimitConfiguration()
'VBA458
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeToleranceLow = False
'Activate monitoring
.CheckToleranceLow = True
'Set barcolor to "yellow"
.ColorToleranceLow = RGB(255, 255, 0)
'Set lower limit to "15"
.ToleranceLow = 15
End With
End Sub
```

Siehe auch

CheckToleranceLow-Eigenschaft (Seite 2133)

BarGraph-Objekt (Seite 1893)

ColorTop-Eigenschaft

Beschreibung

Legt die Farbe für den oberen/linken Anschlag des Slider-Objektes fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "SliderConfiguration()" greift auf die Eigenschaften des Sliders zu. In diesem Beispiel wird die Farbe für den oberen/linken Anschlag auf "Orange" gesetzt:

```
Sub SliderConfiguration()
'VBA459
Dim objSlider As HMISlider
Set objSlider = ActiveDocument.HMIObjects.AddHMIObject("SliderObject1", "HMISlider")
With objSlider
.ColorTop = RGB(255, 128, 0)
End With
End Sub
```

Siehe auch

Slider-Objekt (Seite 2025)

ColorWarningHigh-Eigenschaft

Beschreibung

Legt die Farbe für den oberen Grenzwert "Warning high" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Die Eigenschaft "CheckWarningHigh" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "75" ausgelöst werden und die Balkenfarbe auf "rot" wechseln:

```
Sub BarGraphLimitConfiguration()
'VBA460
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeWarningHigh = False
'Activate monitoring
.CheckWarningHigh = True
'Set barcolor to "red"
.ColorWarningHigh = RGB(255, 0, 0)
'Set upper limit to "75"
.WarningHigh = 75
End With
End Sub
```

Siehe auch

CheckWarningHigh-Eigenschaft (Seite 2134)

BarGraph-Objekt (Seite 1893)

ColorWarningLow-Eigenschaft

Beschreibung

Legt die Farbe für den unteren Grenzwert "Warning low" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Die Eigenschaft "CheckWarningLow" muss den Wert TRUE haben, wenn sich die Balkenfarbe bei Erreichen des Grenzwertes ändern soll.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut"

gesetzt. Der Alarm soll bei einem Wert von "12" ausgelöst werden und die Balkenfarbe auf "magenta" wechseln:

```
Sub BarGraphLimitConfiguration()
'VBA461
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeWarningLow = False
'Activate monitoring
.CheckWarningLow = True
'Set barcolor to "magenta"
.ColorWarningLow = RGB(255, 0, 255)
'Set lower limit to "12"
.WarningLow = 12
End With
End Sub
```

Siehe auch

CheckWarningLow-Eigenschaft (Seite 2135)

BarGraph-Objekt (Seite 1893)

CommonVBSCode-Eigenschaft

Beschreibung

Legt für das aktive Bild den übergeordneten, gemeinsamen Deklarationsbereich der Aktionen fest oder gibt ihn zurück.

Im Aktionseditor des Graphics Designers projektieren Sie Aktionen an Ereignisse und Eigenschaften. Im Deklarationsbereich der Aktionen können Sie für ein Prozessbild gemeinsame Variablen deklarieren sowie Funktionen und Prozeduren erstellen. In Runtime kann jede VBS-Aktion auf diese Variablen, Funktionen und Prozeduren zugreifen, wenn das Bild aktiv ist.

Wenn Sie "CommonVBSCode" setzen, wird der String in die Deklarationsbereiche "Ereignis" und "Eigenschaft" im Aktionseditor kopiert. Code, der dort bereits steht, wird überschrieben. Setzen Sie daher "CommonVBSCode" zuerst, bevor Sie die untergeordneten Deklarationsbereiche mit "CommonVBSEventArea" oder "CommonVBSPropertyArea" setzen.

Beispiel

Im folgenden Beispiel wird im aktiven Bild eine Variable deklariert, die allen Bildobjekten gemeinsam ist. Anschließend wird der gemeinsame Deklarationsbereich ausgegeben:

```
Sub DefineTagInActiveDocument
```

```
ActiveDocument.CommonVBSCode = "DIM actionIsdone" & vbCrLf
MsgBox ActiveDocument.CommonVBSCode
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

CommonVBSEventArea-Eigenschaft

Beschreibung

Legt für das aktive Bild den Deklarationsbereich "Ereignis" der Aktionen fest oder gibt ihn zurück.

Im Aktionseditor des Graphics Designers projektieren Sie Aktionen z. B. an Ereignisse. Dafür können Sie im Deklarationsbereich "Ereignis" für ein Prozessbild gemeinsame Variablen deklarieren sowie Funktionen und Prozeduren erstellen. In Runtime kann jede VBS-Aktion, die für ein Ereignis projektiert wurde, auf diese Variablen, Funktionen und Prozeduren zugreifen, wenn das Bild aktiv ist.

Wenn Sie "CommonVBSEventArea" setzen, wird der String in den Deklarationsbereich "Ereignis" im Aktionseditor kopiert. Code, der dort bereits steht, wird überschrieben. Lesen Sie daher den z. B. mit "CommonVBSCode" gesetzten Code zuerst aus, bevor Sie den Deklarationsbereich mit "CommonVBSEventArea" setzen.

Beispiel

Im folgenden Beispiel werden im aktiven Bild zwei Variablen deklariert. Anschließend wird der Deklarationsbereich "Ereignis" ausgegeben:

```
Sub DefineTagInActiveDocument
ActiveDocument.CommonVBSCode = "DIM actionIsdone" & vbCrLf
ActiveDocument.CommonVBSEventArea = ActiveDocument.CommonVBSEventArea & "DIM
"eventHasOccurred"
MsgBox ActiveDocument.CommonVBSEventArea
End Sub
```

CommonVBSPropertyArea-Eigenschaft

Beschreibung

Legt für das aktive Bild den Deklarationsbereich "Eigenschaft" der Aktionen fest oder gibt ihn zurück.

Im Aktionseditor des Graphics Designers projektieren Sie Aktionen z. B. an Eigenschaften. Dafür können Sie im Deklarationsbereich "Eigenschaft" für ein Prozessbild gemeinsame

Variablen deklarieren sowie Funktionen und Prozeduren erstellen. In Runtime kann jede VBS-Aktion, die für eine Eigenschaft projektiert wurde, auf diese Variablen, Funktionen und Prozeduren zugreifen, wenn das Bild aktiv ist.

Wenn Sie "CommonVBSPropertyArea" setzen, wird der String in den Deklarationsbereich "Eigenschaft" im Aktionseditor kopiert. Code, der dort bereits steht, wird überschrieben. Lesen Sie daher den z. B. mit "CommonVBSCode" gesetzten Code zuerst aus, bevor Sie den Deklarationsbereich mit "CommonVBSPropertyArea" setzen.

Beispiel

Im folgenden Beispiel werden im aktiven Bild zwei Variablen deklariert. Anschließend wird der Deklarationsbereich "Eigenschaft" ausgegeben:

```
Sub DefineTagInActiveDocument
ActiveDocument.CommonVBSCode = "DIM actionIsdone" & vbCrLf
ActiveDocument. CommonVBSPropertyArea = ActiveDocument.CommonVBSPropertyArea & "DIM
propertyIsChanged"
MsgBox ActiveDocument.CommonVBSPropertyArea
End Sub
```

CommandLine-Eigenschaft

Beschreibung

Gibt den Startparameter als String zurück, wenn die Applikation über Start>Ausführen "Grafexe.exe Startparameter" geöffnet wird. Lese-Zugriff.

Beispiel

In diesem Beispiel wird beim Öffnen des Dokuments eine Meldung mit dem Startparameter ausgegeben.

```
Sub Document_Opened(CancelForwarding As Boolean)
'VBA462
MsgBox Application.Commandline
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

Compiled-Eigenschaft

Beschreibung

TRUE, wenn der Quellcode eines C- oder VB-Skriptes erfolgreich kompiliert werden konnte. BOOLEAN Lese-Zugriff.

Beispiel

Im folgenden Beispiel werden in das aktive Bild ein Button und ein Kreis eingefügt. In Runtime soll bei jedem Klick auf den Button der Radius des Kreises vergrößert werden. Dazu wird ein VB-Skript verwendet:

```
Sub IncreaseCircleRadiusWithVBScript()
'VBA463
Dim objButton As HMIButton
Dim objCircleA As HMICircle
Dim objEvent As HMIEvent
Dim objVBScript As HMIScriptInfo
Dim strCode As String
strCode = "Dim objCircle" & vbCrLf & "Set objCircle = "
strCode = strCode & "hmiRuntime.ActiveScreen.ScreenItems(""CircleVB"")"
strCode = strCode & vbCrLf & "objCircle.Radius = objCircle.Radius + 5"
Set objCircleA = ActiveDocument.HMIObjects.AddHMIObject("CircleVB", "HMICircle")
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objCircleA
.Top = 100
.Left = 100
End With
With objButton
.Top = 10
.Left = 10
.Width = 200
.Text = "Increase Radius"
End With
'On every mouseclick the radius will be increased:
Set objEvent = objButton.Events(1)
Set objVBScript = objButton.Events(1).Actions.AddAction(hmiActionCreationTypeVBScript)
objVBScript.SourceCode = strCode
Select Case objVBScript.Compiled
Case True
MsgBox "Compilation OK!"
Case False
MsgBox "Errors by compilation!"
End Select
End Sub
```

Siehe auch

SourceCode-Eigenschaft (Seite 2372)

ScriptInfo-Objekt (Seite 2021)

ConfigurationFileName-Eigenschaft

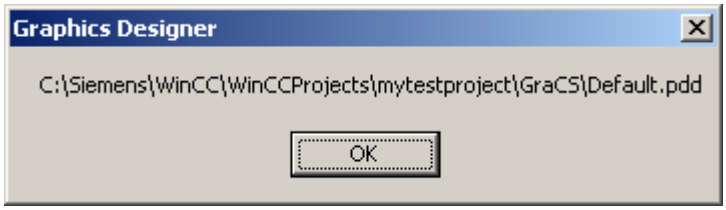
Beschreibung

Gibt Dateinamen und vollständigen Pfad der Konfigurationsdatei des geöffneten Projektes zurück. STRING Lese-Zugriff.

Beispiel

Die Prozedur "ShowConfigurationFileName() gibt den Pfad Konfigurationsdatei des aktuellen Bildes aus:

```
Sub ShowConfigurationFileName()
'VBA464
MsgBox ActiveDocument.Application.ConfigurationFileName
End Sub
```



Siehe auch

Application-Eigenschaft (Seite 2079)

Application-Objekt (Seite 1888)

Count-Eigenschaft

Beschreibung

Gibt die Anzahl der Elemente in der angegebenen Auflistung zurück. Long Lese-Zugriff

Beispiel

Im folgenden Beispiel wird ein neues Bild angelegt, in das ein paar Objekte eingefügt werden. Die Anzahl der eingefügten Objekte wird am Ende ausgegeben:

```
Sub ObjectsInActiveDocument()
'VBA465
Dim objCircle As HMICircle
```

```
Dim objRectangle As HMIRectangle
Dim objDocument As Document
Set objDocument = Application.Documents.Add(hmiOpenDocumentTypeVisible)
Dim iIndex As Integer
iIndex = 1
For iIndex = 1 To 5
Set objCircle = objDocument.HMIObjects.AddHMIObject("Circle" & iIndex, "HMICircle")
Set objRectangle = objDocument.HMIObjects.AddHMIObject("Rectangle" & iIndex,
"HMIRectangle")
With objCircle
.Top = (10 * iIndex)
.Left = (10 * iIndex)
End With
With objRectangle
.Top = ((10 * iIndex) + 50)
.Left = (10 * iIndex)
End With
Next iIndex
MsgBox "There are " & objDocument.HMIObjects.Count & " objects in the document"
End Sub
```

Siehe auch

VariableTriggers-Objekt (Auflistung) (Seite 2061)
Views-Objekt (Auflistung) (Seite 2064)
VariableStateValues-Objekt (Auflistung) (Seite 2058)
ToolbarItems-Objekt (Auflistung) (Seite 2046)
Toolbars-Objekt (Auflistung) (Seite 2041)
SymbolLibraries-Objekt (Auflistung) (Seite 2036)
Selection-Objekt (Auflistung) (Seite 2022)
Properties-Objekt (Auflistung) (Seite 2004)
HMIObjects-Objekt (Auflistung) (Seite 1957)
MenuItems-Objekt (Auflistung) (Seite 1982)
Menus-Objekt (Auflistung) (Seite 1977)
Layers-Objekt (Auflistung) (Seite 1969)
LanguageTexts-Objekt (Auflistung) (Seite 1966)
LanguageFonts-Objekt (Auflistung) (Seite 1963)
GroupedObjects-Objekt (Auflistung) (Seite 1950)
FolderItems-Objekt (Auflistung) (Seite 1941)
Events-Objekt (Auflistung) (Seite 1936)
Documents-Objekt (Auflistung) (Seite 1923)
HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)
DataLanguages-Objekt (Auflistung) (Seite 1915)
ConnectionPoints-Objekt (Auflistung) (Seite 1910)
AnalogResultInfos-Objekt (Auflistung) (Seite 1887)
Actions-Objekt (Auflistung) (Seite 1884)

CurrentDataLanguage-Eigenschaft

Beschreibung

Legt die Projektiersprache fest oder gibt die Sprachenkennung als Dezimalwert zurück. LONG
Schreib-Lese-Zugriff

Beispiel

Die Prozedur "ShowDataLanguage()" gibt die aktuell eingestellte Projektiersprache aus:

```
Sub ShowDataLanguage()
```

```
'VBA466
MsgBox Application.CurrentDataLanguage
End Sub
```

Siehe auch

Application-Eigenschaft (Seite 2079)

DataLanguageChanged-Ereignis (Seite 1752)

Sprachabhängige Projektierung mit VBA (Seite 1626)

CurrentDesktopLanguage-Eigenschaft

Beschreibung

Gibt die Sprachenkennung der aktuell eingestellten Oberflächensprache als Dezimalwert zurück. LONG Lese-Zugriff.

Beispiel

Die Prozedur "ShowDesktopLanguage()" gibt die aktuell eingestellte Oberflächensprache aus:

```
Sub ShowDesktopLanguage()
'VBA467
MsgBox Application.CurrentDesktopLanguage
End Sub
```

Siehe auch

Application-Eigenschaft (Seite 2079)

Application-Objekt (Seite 1888)

DesktopLanguageChanged-Ereignis (Seite 1753)

Sprachabhängige Projektierung mit VBA (Seite 1626)

CursorControl-Eigenschaft

Beschreibung

TRUE, wenn bei aktiviertem Alpha-Cursor-Modus der Cursor nach dem Verlassen des Feldes auf das nächste Feld der TAB-Reihenfolge springt. BOOLEAN Schreib-Lese-Zugriff.

Die Eigenschaft CursorMode muss dazu auf TRUE gesetzt sein.

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel soll der Cursor nach Verlassen des Feldes ins nächste Feld springen. Dazu muss zuvor die Eigenschaft Cursormodus auf TRUE gesetzt werden:

```
Sub IOFieldConfiguration()
'VBA468
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
Application.ActiveDocument.CursorMode = True
With objIOField
.CursorControl = True
End With
End Sub
```

Siehe auch

TabOrderAlpha-Eigenschaft (Seite 2378)

TabOrderSwitch-Eigenschaft (Seite 2376)

CursorMode-Eigenschaft (Seite 2157)

ActiveDocument-Eigenschaft (Seite 2067)

TextList-Objekt (Seite 2037)

IOField-Objekt (Seite 1959)

CursorMode-Eigenschaft

Beschreibung

TRUE, wenn der Modus "Alpha-Cursor" aktiviert werden soll. FALSE, wenn der Modus "Schalt-Cursor" aktiviert werden soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "ActiveDocumentConfiguration()" greift auf die Eigenschaften des aktuellen Bildes im Graphics Designer zu. In diesem Beispiel wird der Modus "Alpha-Cursor" aktiviert:

```
Sub ActiveDocumentConfiguration()
'VBA469
Application.ActiveDocument.CursorMode = True
End Sub
```

Siehe auch

CursorControl-Eigenschaft (Seite 2156)

ActiveDocument-Eigenschaft (Seite 2067)

Documents-Objekt (Auflistung) (Seite 1923)

CustomMenus-Eigenschaft

Beschreibung

Gibt eine Auflistung der vorhandenen benutzerdefinierten Menüs zurück.

Beispiel

Die Prozedur "ShowCustomMenuInformation()" gibt Key und Label aller benutzerdefinierten Menüs des aktuellen Bildes aus:

```
Sub ShowCustomMenuInformation()
'VBA470
Dim strKey As String
Dim strLabel As String
Dim strOutput As String
Dim iIndex As Integer
For iIndex = 1 To ActiveDocument.CustomMenus.Count
strKey = ActiveDocument.CustomMenus(iIndex).Key
strLabel = ActiveDocument.CustomMenus(iIndex).Label
strOutput = strOutput & vbCrLf & "Key: " & strKey & "  Label: " & strLabel
Next iIndex
If 0 = ActiveDocument.CustomMenus.Count Then
strOutput = "There are no custommenus for the document created."
End If
MsgBox strOutput
End Sub
```

Siehe auch

Application-Eigenschaft (Seite 2079)

ActiveDocument-Eigenschaft (Seite 2067)

Menu-Objekt (Seite 1976)

CustomToolbars-Eigenschaft

Beschreibung

Gibt eine Auflistung der vorhandenen benutzerdefinierten Symbolleisten zurück.

Beispiel

Die Prozedur "ShowCustomToolbarInformation()" gibt den Key-Wert aller benutzerdefinierten Symbolleisten des aktuellen Bildes aus:

```
Sub ShowCustomToolbarInformation()
'VBA471
Dim strKey As String
Dim strOutput As String
Dim iIndex As Integer
For iIndex = 1 To ActiveDocument.CustomToolbars.Count
strKey = ActiveDocument.CustomToolbars(iIndex).Key
strOutput = strOutput & vbCrLf & "Key: " & strKey
Next iIndex
If 0 = ActiveDocument.CustomToolbars.Count Then
strOutput = "There are no toolbars created for this document."
End If
MsgBox strOutput
End Sub
```

Siehe auch

Application-Eigenschaft (Seite 2079)

ActiveDocument-Eigenschaft (Seite 2067)

Toolbar-Objekt (Seite 2040)

CycleName-Eigenschaft

Beschreibung

Gibt den Namen des angegebenen Variablentriggers zurück. Lese-Zugriff.

Beispiel

--

Siehe auch

VariableTrigger-Objekt (Seite 2060)

CycleTime-Eigenschaft

Beschreibung

Gibt die Zykluszeit des angegebenen Variablen-Triggers zurück. Lese-Zugriff.

Beispiel

--

Siehe auch

VariableTrigger-Objekt (Seite 2060)

CycleType-Eigenschaft

Beschreibung

Legt die Zyklusart fest oder gibt ihn zurück.

Beispiel

Die Prozedur "DynamicToRadiusOfNewCircle(hmiCircle As IHMICircle)" erzeugt für den Radius eines Kreises eine Dynamik. In diesem Beispiel soll der Radius des Kreises alle zwei Sekunden gesetzt werden:

```
Sub DynamicToRadiusOfNewCircle()
'VBA474
Dim objCircle As hmiCircle
Dim VariableTrigger As HMIVariableTrigger
Set objCircle = Application.ActiveDocument.HMIObjects.AddHMIObject("Circle1", "HMICircle")
Set VariableTrigger = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeVariableDirect,
"NewDynamic1")
VariableTrigger.CycleType = hmiVariableCycleType_2s
End Sub
```

Siehe auch

VariableTrigger-Objekt (Seite 2060)

Dynamisieren von Eigenschaften von Bildern und Objekten (Seite 1692)

D

DataFormat-Eigenschaft

Beschreibung

Legt den Datentyp des Objektes IOField fest oder gibt ihn zurück. Wertebereich von 0 bis 3.

Datentyp	zugeordneter Wert
Binär	0
Dezimal	1
String	2
Hexadezimal	3

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel wird für das EA-Feld der Datentyp "Dezimal" gesetzt:

```
Sub IOFieldConfiguration()
'VBA475
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.DataFormat = 1
End With
End Sub
```

Siehe auch

IOField-Objekt (Seite 1959)

DefaultHMIObjects-Eigenschaft

Beschreibung

Gibt die HMIDefaultObjects-Auflistung zurück.

Beispiel

Die Prozedur "ShowDefaultObjectNames()"gibt alle Objektnamen aus, die in der HMIDefaultObjects-Auflistung enthalten sind:

```
Sub ShowDefaultObjectNames()
'VBA476
Dim strOutput As String
Dim iIndex As Integer
For iIndex = 1 To Application.DefaultHMIObjects.Count
strOutput = strOutput & vbCrLf & Application.DefaultHMIObjects(iIndex).ObjectName
Next iIndex
MsgBox strOutput
End Sub
```

Siehe auch

HMIDefaultObjects-Objekt (Auflistung) (Seite 1951)

DestinationLink-Eigenschaft

Beschreibung

Gibt das Destination-Objekt zurück. Verwenden Sie die DestinationLink-Eigenschaft, um bei einer Direktverbindung das Zielobjekt zu konfigurieren.

Beispiel

Verwenden Sie die DestinationLink-Eigenschaft, um das DestLink-Objekt zurückzugeben. Im folgenden Beispiel wird in Runtime bei einem Mausklick auf den Button die x-Position von "Rectangle_A" in die y-Position von "Rectangle_B" kopiert:

```
Sub DirectConnection()
'VBA477
Dim objButton As HMIButton
Dim objRectangleA As HMIRectangle
Dim objRectangleB As HMIRectangle
Dim objEvent As HMIEvent
Dim objDirConnection As HMIDirectConnection
Set objRectangleA = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_A", "HMIRectangle")
Set objRectangleB = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_B", "HMIRectangle")
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objRectangleA
.Top = 100
.Left = 100
End With
With objRectangleB
.Top = 250
.Left = 400
```

```
.BackColor = RGB(255, 0, 0)
End With
With objButton
.Top = 10
.Left = 10
.Width = 100
.Text = "SetPosition"
End With
'
'Directconnection is initiated by mouseclick:
Set objDirConnection =
objButton.Events(1).Actions.AddAction(hmiActionCreationTypeDirectConnection)
With objDirConnection
'Sourceobject: Property "Top" of Rectangle_A
.SourceLink.Type = hmiSourceTypeProperty
.SourceLink.ObjectName = "Rectangle_A"
.SourceLink.AutomationName = "Top"
'
'Targetobject: Property "Left" of Rectangle_B
.DestinationLink.Type = hmiDestTypeProperty
.DestinationLink.ObjectName = "Rectangle_B"
.DestinationLink.AutomationName = "Left"
End With
End Sub
```

Siehe auch

AutomationName-Eigenschaft (Seite 2082)

ObjectName-Eigenschaft (Seite 2307)

Type-Eigenschaft (Seite 2392)

DirectConnection-Objekt (Seite 1918)

Direction-Eigenschaft

Beschreibung

Legt die Balkenrichtung fest oder gibt sie zurück. BOOLEAN Schreib-Lese-Zugriff.

Slider

Legt die Lage des Slider-Objektes fest oder gibt sie zurück. BOOLEAN Schreib-Lese-Zugriff.

Lage/Balkenachse	Zugeordneter Wert
Vertikal/Negativ	TRUE
Horizontal/Positiv	FALSE

Beispiel

Die Prozedur "SliderConfiguration()" greift auf die Eigenschaften des Sliders zu. In diesem Beispiel wird die Lage des Slider-Objektes auf "vertikal" gesetzt:

```
Sub SliderConfiguration()
'VBA478
Dim objSlider As HMISlider
Set objSlider = ActiveDocument.HMIObjects.AddHMIObject("SliderObject1", "HMISlider")
With objSlider
.Direction = True
End With
End Sub
```

Siehe auch

Slider-Objekt (Seite 2025)

3DBarGraph-Objekt (Seite 1879)

DisableVBAEvents-Eigenschaft

Beschreibung

TRUE, wenn das Event-Handling abgeschaltet ist. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "DisableVBAEvents()" schaltet das Event-Handling aus:

```
Sub DisableVBAEvents()
'VBA479
Application.DisableVBAEvents = False
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

Event-Handling (Seite 1715)

DisplayName-Eigenschaft

Beschreibung

Gibt den Namen des Eigenschaftsattributes zurück. STRING Lese-Zugriff.

So würde der Ausdruck "MsgBox ActiveDocument.HMIObjects("Kreis_1").Properties("Height").DisplayName" als Ergebnis "Höhe" ausgeben.

Beispiel

Die Prozedur "ShowAllObjectDisplayNames()" gibt alle Eigenschaftsattributnamen der Standard-Objekte aus in einer Messagebox aus:

```
Sub ShowAllObjectDisplayNames()
'VBA480
Dim strOutput As String
Dim iIndex1 As Integer
iIndex1 = 1
strOutput = "List of all properties-displaynames from object """ &
Application.DefaultHMIObjects(1).ObjectName & """" & vbCrLf & vbCrLf
For iIndex1 = 1 To Application.DefaultHMIObjects(1).Properties.Count
strOutput = strOutput & Application.DefaultHMIObjects(1).Properties(iIndex1).DisplayName &
" / "
Next iIndex1
MsgBox strOutput
End Sub
```

Siehe auch

Property-Objekt (Seite 2005)

DisplayOptions-Eigenschaft

Beschreibung

Legt die Belegung der Objekte "Button" oder "Rundbutton" fest oder gibt den Wert zurück. Wertebereich von 0 bis 3.

Belegung	zugeordneter Wert
Grafik oder Text	0
Grafik und Text	1
nur Text	2
nur Grafik	3

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu.

In diesem Beispiel wird für den Button die Belegung "Grafik und Text" gesetzt:

```
Sub ButtonConfiguration()
'VBA814
Dim objbutton As HMIButton
Set objbutton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objbutton
.DisplayOptions = 1
End With
End Sub
```

Siehe auch

Button-Objekt (Seite 1898)

DisplayText-Eigenschaft

Beschreibung

Gibt den Wert der Eigenschaft "Label" bzw. "TooltipText" von folgenden Objekten zurück (STRING Lese-Zugriff):

- Menu-Objekt
- MenuItem-Objekt
- ToolbarItem-Objekt

Beispiel

Die Prozedur "ShowLabelTexts()" gibt die alle Beschriftungen des ersten benutzerdefinierten Menüs im aktiven Bild aus:

```
Sub ShowLabelTexts()
'VBA481
Dim objLangText As HMILanguageText
Dim iIndex As Integer
For iIndex = 1 To ActiveDocument.CustomMenus(1).LDLabelTexts.Count
Set objLangText = ActiveDocument.CustomMenus(1).LDLabelTexts(iIndex)
MsgBox objLangText.DisplayName
Next iIndex
End Sub
```

Siehe auch

TooltipText-Eigenschaft (Seite 2387)

Label-Eigenschaft (Seite 2231)

ToolbarItem-Objekt (Seite 2043)

LanguageText-Objekt (Seite 1965)

MenuItem-Objekt (Seite 1979)

Menu-Objekt (Seite 1976)

Documents-Eigenschaft

Beschreibung

Gibt die Documents-Auflistung zurück, die alle geöffneten Bilder enthält. Die Reihenfolge der geöffneten Bilder wird chronologisch festgelegt.

Beispiel

Im folgenden Beispiel werden die Namen aller geöffneten Bilder ausgegeben:

```
Sub ShowDocuments()
'VBA482
Dim colDocuments As Documents
Dim objDocument As Document
Dim strOutput As String
Set colDocuments = Application.Documents
strOutput = "List of all opened documents:" & vbCrLf
For Each objDocument In colDocuments
strOutput = strOutput & vbCrLf & objDocument.Name
Next objDocument
MsgBox strOutput
End Sub
```

Siehe auch

Application-Eigenschaft (Seite 2079)

Application-Objekt (Seite 1888)

Dynamic-Eigenschaft

Beschreibung

Gibt die Dynamisierung einer Eigenschaft zurück.

Beispiel

Verwenden Sie die Dynamic-Eigenschaft, wenn Sie z.B. eine vorhandene Dynamik zurückgeben wollen. Im folgenden Beispiel werden alle eventuell vorhandenen Dynamisierungen von Objekteigeschaften im aktiven Bild ausgegeben:

```
Sub ShowPropertiesDynamicsofAllObjects()
'VBA483
Dim objObject As HMIObject
Dim colObjects As HMIObjects
Dim colProperties As HMIProperties
Dim objProperty As HMIProperty
Dim strOutput As String
Set colObjects = Application.ActiveDocument.HMIObjects
For Each objObject In colObjects
Set colProperties = objObject.Properties
For Each objProperty In colProperties
If 0 <> objProperty.DynamicStateType Then
  strOutput = strOutput & vbCrLf & objObject.ObjectName & " - " & objProperty.DisplayName
& ": Statetype " & objProperty.Dynamic.DynamicStateType
End If
Next objProperty
Next objObject
MsgBox strOutput
End Sub
```

Siehe auch

Property-Objekt (Seite 2005)

E

EditAtOnce-Eigenschaft

Beschreibung

TRUE, wenn beim Anspringen des Feldes mit der Taste <Tab> die Eingabe sofort und ohne weitere Aktion erfolgen kann. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel wird soll beim Anspringen des EA-Feldes die Eingabe sofort erfolgen:

```
Sub IOFieldConfiguration()
'VBA484
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
```

```
With objIOField
.EditAtOnce = True
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

IOField-Objekt (Seite 1959)

ElseCase-Eigenschaft

Beschreibung

Legt den Wert für die dynamisierte Eigenschaft außerhalb der projektierten Wertebereiche fest oder gibt ihn zurück.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert, ein Variablenname vergeben und drei analoge Wertebereiche angelegt:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA485
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.Add 50, 40
.AnalogResultInfos.Add 100, 80
.AnalogResultInfos.ElseCase = 100
End With
End Sub
```

Siehe auch

AnalogResultInfos-Objekt (Auflistung) (Seite 1887)

AnalogResultInfo-Objekt (Seite 1886)

Add-Methode (AnalogResultInfos-Auflistung) (Seite 1777)

Enabled-Eigenschaft

Beschreibung

TRUE, wenn das Menü, der Menüeintrag oder das Symbol aktiviert ist und ausgewählt werden kann. Gilt nur für benutzerdefinierte Menüs und Symbolleisten. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "CreateMenuItem()" legt das Menü "Objekte löschen" an und fügt zwei Menüeinträge ("Rechtecke löschen" und "Kreise löschen") hinzu. In diesem Beispiel wird der zweite Menüpunkt des benutzerdefinierten Menüs "Objekte löschen" ausgegraut und kann im Graphics Designer nicht ausgewählt werden:

```
Sub DisableMenuItem()
'VBA486
Dim objMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
'
'Add a new menu "Delete objects"
Set objMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DeleteObjects", "Delete objects")
'
'Add two menuitems to the new menu
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(1, "DeleteAllRectangles", "Delete
rectangles")
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(2, "DeleteAllCircles", "Delete circles")
'
'Disable menuitem "Delete circles"
With ActiveDocument.CustomMenus("DeleteObjects").MenuItems("DeleteAllCircles")
.Enabled = False
End With
End Sub
```

Siehe auch

ToolbarItem-Objekt (Seite 2043)

MenuItem-Objekt (Seite 1979)

Menu-Objekt (Seite 1976)

Menüs und Symbolleisten konfigurieren (Seite 1628)

EndAngle-Eigenschaft

Beschreibung

Legt bei den Objekten CircularArc, EllipseArc, EllipseSegment und PieSegment das Ende des Objektes fest oder gibt es zurück. Die Angabe erfolgt im Uhrzeigersinn in Grad, beginnend bei 12:00 Uhr.

Beispiel

Die Prozedur "PieSegmentConfiguration()" greift auf die Eigenschaften des Kreissegmentes zu. In diesem Beispiel beginnt das Kreissegmentes bei 40° und endet bei 180°:

```
Sub PieSegmentConfiguration()
'VBA487
Dim objPieSegment As HMIPieSegment
Set objPieSegment = ActiveDocument.HMIObjects.AddHMIObject("PieSegment1", "HMIPieSegment")
With objPieSegment
.StartAngle = 40
.EndAngle = 180
End With
End Sub
```

Siehe auch

StartAngle-Eigenschaft (Seite 2373)

PieSegment-Objekt (Seite 1995)

EllipseSegment-Objekt (Seite 1932)

EllipseArc-Objekt (Seite 1929)

CircularArc-Objekt (Seite 1905)

Events-Eigenschaft

Beschreibung

Gibt die Events-Auflistung zurück. Verwenden Sie die Events-Eigenschaft, um das Ereignis festzulegen, das eine Aktion auslösen soll. Welches Ereignis projektiert werden soll, legen Sie mit der Indexnummer fest:

- Sie projektieren mit VBA eine Aktion an eine Eigenschaft, indem Sie die Eigenschaft "Events(9)" verwenden, wobei der Index "1" für das Ereignis "bei Änderung" steht.
- Sie projektieren mit VBA eine Aktion an ein Objekt, indem Sie die Eigenschaft "Events(Index)" verwenden, wobei "Index" für das auslösende Ereignis steht (siehe Tabelle):

Index	EventType (abhängig vom verwendeten Objekt)
0	hmiEventTypeNotDefined
1	hmiEventTypeMouseClick
2	hmiEventTypeMouseLButtonDown
3	hmiEventTypeMouseLButtonUp
4	hmiEventTypeMouseRButtonDown
5	hmiEventTypeMouseRButtonUp
6	hmiEventTypeKeyboardDown
7	hmiEventTypeKeyboardUp
8	hmiEventTypeFocusEnter
9	hmiEventTypeObjectChange
10	hmiEventTypeOpenPicture
11	hmiEventTypePictureOpen
12	hmiEventTypePictureClose
13	hmiEventTypeObjectDefined
14	hmiEventTypeFocusEnter
15	hmiEventTypeLastTriggerType
16	hmiEventTypeObjSpecificTriggerStart

Beispiel

Im folgenden Beispiel wird in Runtime bei einem Mausklick auf den Button die x-Position von "Rectangle_A" in die y-Position von "Rectangle_B" kopiert:

```
Sub DirectConnection()
'VBA488
Dim objButton As HMIButton
Dim objRectangleA As HMIRectangle
Dim objRectangleB As HMIRectangle
Dim objEvent As HMIEvent
Dim objDirConnection As HMIDirectConnection
Set objRectangleA = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_A", "HMIRectangle")
Set objRectangleB = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_B", "HMIRectangle")
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objRectangleA
.Top = 100
.Left = 100
End With
With objRectangleB
.Top = 250
.Left = 400
.BackColor = RGB(255, 0, 0)
End With
With objButton
.Top = 10
.Left = 10
.Width = 100
.Text = "SetPosition"
```

```
End With
'
'Directconnection is initiated by mouseclick:
Set objDirConnection =
objButton.Events(1).Actions.AddAction(hmiActionCreationTypeDirectConnection)
With objDirConnection
'Sourceobject: Property "Top" of Rectangle_A
.SourceLink.Type = hmiSourceTypeProperty
.SourceLink.ObjectName = "Rectangle_A"
.SourceLink.AutomationName = "Top"
'
'Targetobject: Property "Left" of Rectangle_B
.DestinationLink.Type = hmiDestTypeProperty
.DestinationLink.ObjectName = "Rectangle_B"
.DestinationLink.AutomationName = "Left"
End With
End Sub
```

Siehe auch

Events-Objekt (Auflistung) (Seite 1936)

Projektierung von ereignisgesteuerten Aktionen mit VBA (Seite 1704)

EventType-Eigenschaft

Beschreibung

Gibt die Ereignisart zurück, die an das angegebene Objekt projektiert ist.

Index	EventType (abhängig vom verwendeten Objekt)
0	hmiEventTypeNotDefined
1	hmiEventTypeMouseClick
2	hmiEventTypeMouseLButtonDown
3	hmiEventTypeMouseLButtonUp
4	hmiEventTypeMouseRButtonDown
5	hmiEventTypeMouseRButtonUp
6	hmiEventTypeKeyboardDown
7	hmiEventTypeKeyboardUp
8	hmiEventTypeFocusEnter
9	hmiEventTypeObjectChange
10	hmiEventTypeOpenPicture
11	hmiEventTypePictureOpen
12	hmiEventTypePictureClose
13	hmiEventTypeObjectDefined
14	hmiEventTypeFocusEnter

Index	EventType (abhängig vom verwendeten Objekt)
15	hmiEventTypeLastTriggerType
16	hmiEventTypeObjSpecificTriggerStart

Beispiel

Verwenden Sie die EventType-Eigenschaft, um ein bereits projektiertes Ereignis zu bearbeiten. Im folgenden Beispiel wird zunächst das Ereignis "Mausklick" projektiert, danach aber auf "gedrückt" geändert:

```
Sub AddActionToObjectTypeCScript()
'VBA489
Dim objEvent As HMIEvent
Dim objCScript As HMIScriptInfo
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_AB", "HMICircle")
'
'C-action is initiated by click on object circle
Set objEvent = objCircle.Events(1)
Set objCScript = objEvent.Actions.AddAction(hmiActionCreationTypeCScript)
MsgBox "the type of the projected event is " & objEvent.EventType
End Sub
```

Siehe auch

Events-Objekt (Auflistung) (Seite 1936)

Projektierung von ereignisgesteuerten Aktionen mit VBA (Seite 1704)

Exponent-Eigenschaft

Beschreibung

TRUE, wenn die Zahlendarstellung beim Objekt BarGraph mit Exponenten (z.B."1,00e +000")erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel soll die Zahlendarstellung des Balkens mit Exponenten erfolgen:

```
Sub BarGraphConfiguration()
'VBA490
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.Exponent = True
```

```
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

ExtendedOperation-Eigenschaft

Beschreibung

TRUE, wenn der Schieberegler beim Objekt Slider auf den zugehörigen Endwert (Minimalwert/ Maximalwert) gestellt wird. Dies geschieht bei Mausklick auf den Bereich außerhalb der aktuellen Reglereinstellung. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "SliderConfiguration()" greift auf die Eigenschaften des Sliders zu. In diesem Beispiel wird die Eigenschaft ExtendedOperation auf TRUE gesetzt:

```
Sub SliderConfiguration()
'VBA491
Dim objSlider As HMISlider
Set objSlider = ActiveDocument.HMIObjects.AddHMIObject("SliderObject1", "HMISlider")
With objSlider
.ExtendedOperation = True
End With
End Sub
```

Siehe auch

Slider-Objekt (Seite 2025)

ExtendedZoomingEnable-Eigenschaft

Beschreibung

TRUE, wenn für das ausgewählte Prozessbild in Runtime das Bild durch das Mausrad vergrößert oder verkleinert werden kann. Dies geschieht bei Drücken der Taste <Strg>, während das Mausrad gedreht wird. Wenn das Mausrad vom Handteller weg gedreht wird, vergrößert sich der Zoomfaktor.

BOOLEAN Schreib-Lese-Zugriff.

Voraussetzung für die Nutzung der Zoomfunktion:

- Maustreiber von Logitech oder Microsoft-Intellimouse
- Das Mausrad muss auf "Autoscroll" eingestellt sein.
- In den Rechnereigenschaften muss auf der der Registerkarte "Graphics-Runtime" die Funktion "Erweitertes Zoomen" für alle Prozessbilder eingeschaltet sein.

Beispiel

Die Prozedur "DocConfiguration()" greift auf die Eigenschaften eines Bildes zu.

In diesem Beispiel wird die Eigenschaft ExtendedZoomingEnable auf TRUE gesetzt:

```
Sub DocConfiguration()
'VBA815
Dim objDoc As Document
Set objDoc = ActiveDocument
With objDoc
.ExtendedZoomingEnable = True
End With
End Sub
```

F

FaceplateType-Eigenschaft

Beschreibung

Setzt den Faceplate-Typ der Faceplate-Instanz und gibt seinen Namen zurück. Der Faceplate-Typ ist "Const" und kann daher nur einmal gesetzt werden.

Verwendung

Verwenden Sie die Add-Methode, um ein neues Objekt "Faceplate-Instanz" im Bild anzulegen. Über "Properties.Item(3)" greifen Sie auf die FaceplateType-Eigenschaft zu:

```
Sub FaceplateInstance_and_Properties()
'VBA847
Dim objFaceplateInstance As HMIFaceplateObject

Set objFaceplateInstance = ActiveDocument.HMIObjects.AddHMIObject("Faceplate-Instanz",
"HMIFaceplateObject")
objFaceplateInstance.Properties.Item(3).value = "Faceplate1.fpt"
MsgBox "Faceplate """ & objFaceplateInstance.Properties.Item(3).value & """ is used."

End Sub
```

FillColor-Eigenschaft

Beschreibung

Legt die Farbe des Füllmusters für das Objekt fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel wird die Hintergrundfarbe auf "Gelb" gesetzt:

```
Sub RectangleConfiguration()
'VBA493
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.FillColor = RGB(255, 255, 0)
End With
End Sub
```

Siehe auch

Button-Objekt (Seite 1898)
StaticText-Objekt (Seite 2029)
Slider-Objekt (Seite 2025)
TextList-Objekt (Seite 2037)
RoundRectangle-Objekt (Seite 2018)
RoundButton-Objekt (Seite 2015)
Rectangle-Objekt (Seite 2012)
Polygon-Objekt (Seite 1998)
PieSegment-Objekt (Seite 1995)
OptionGroup-Objekt (Seite 1989)
GroupDisplay-Objekt (Seite 1947)
GraphicObject-Objekt (Seite 1943)
IOField-Objekt (Seite 1959)
EllipseSegment-Objekt (Seite 1932)
Ellipse-Objekt (Seite 1926)
Document-Objekt (Seite 1920)
Circle-Objekt (Seite 1902)
CheckBox-Objekt (Seite 1901)
BarGraph-Objekt (Seite 1893)
3DBarGraph-Objekt (Seite 1879)

Filling-Eigenschaft

Beschreibung

TRUE, wenn das Objekt mit geschlossener Rahmenlinie (z.B. Circle oder Rectangle) gefüllt werden kann (also z.B. den Füllstand eines Tanks darstellt). BOOLEAN Schreib-Lese-Zugriff.

Den Füllstand des Objektes setzen Sie mit der Eigenschaft FillingIndex.

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel kann das Rechteck als Füllstandsanzeige verwendet werden:

```
Sub RectangleConfiguration()
'VBA494
Dim objRectangle As HMIRectangle
```

```
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.Filling = True
End With
End Sub
```

Siehe auch

FillingIndex-Eigenschaft (Seite 2179)

StaticText-Objekt (Seite 2029)

Slider-Objekt (Seite 2025)

RoundRectangle-Objekt (Seite 2018)

RoundButton-Objekt (Seite 2015)

Rectangle-Objekt (Seite 2012)

Polygon-Objekt (Seite 1998)

PieSegment-Objekt (Seite 1995)

OptionGroup-Objekt (Seite 1989)

GraphicObject-Objekt (Seite 1943)

EllipseSegment-Objekt (Seite 1932)

Ellipse-Objekt (Seite 1926)

Circle-Objekt (Seite 1902)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

FillingIndex-Eigenschaft

Beschreibung

Legt den %-Wert (bezogen auf die Höhe des Objekts) fest, zu dem das Objekt mit geschlossener Rahmenlinie (z.B. Circle oder Rectangle) gefüllt wird.

Der Füllstand wird mit der aktuellen Hintergrundfarbe dargestellt. Der nicht gefüllte Hintergrund ist transparent.

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel wird das Rechteck zu 50% gefüllt:

```
Sub RectangleConfiguration()
'VBA495
```

```
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.Filling = True
.FillingIndex = 50
End With
End Sub
```

Siehe auch

PieSegment-Objekt (Seite 1995)

FillColor-Eigenschaft (Seite 2177)

BackColor-Eigenschaft (Seite 2088)

StaticText-Objekt (Seite 2029)

Slider-Objekt (Seite 2025)

RoundRectangle-Objekt (Seite 2018)

RoundButton-Objekt (Seite 2015)

Rectangle-Objekt (Seite 2012)

Polygon-Objekt (Seite 1998)

OptionGroup-Objekt (Seite 1989)

GraphicObject-Objekt (Seite 1943)

EllipseSegment-Objekt (Seite 1932)

Ellipse-Objekt (Seite 1926)

Circle-Objekt (Seite 1902)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

FillingDirection-Eigenschaft

Beschreibung

0 = das Objekt mit geschlossener Rahmenlinie wird von unten nach oben gefüllt.

1 = das Objekt mit geschlossener Rahmenlinie wird von oben nach unten gefüllt.

2 = das Objekt mit geschlossener Rahmenlinie wird von links nach rechts gefüllt.

3 = das Objekt mit geschlossener Rahmenlinie wird von rechts nach links gefüllt.

Schreib-Lese-Zugriff.
Die Füllrichtung des Objektes setzen Sie mit der Eigenschaft "FillingDirection".

Beispiel

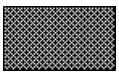
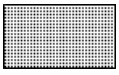
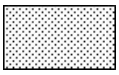
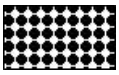
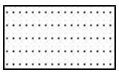
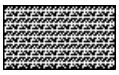
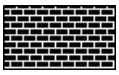
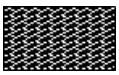
Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel wird das Rechteck von links nach rechts gefüllt.

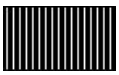
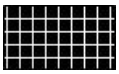
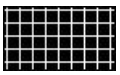
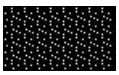
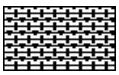
```
Sub RectangleConfiguration()
'VBAxxx
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.FillingDirection = 2
End With
End Sub
```

FillStyle-Eigenschaft

Beschreibung

Legt das Füllmuster für das Objekt fest oder gibt es zurück.

Füllmuster	Wert	Füllmuster	Wert	Füllmuster	Wert
< Transparent >	65536				
< Massiv >	0				
	1048576		196611		196627
	1048577		196612		196628
	1048578		196613		196629
	1048579		196614		196630
	1048832		196615		196631
	1048833		196616		196632
	1048834		196617		196633
	1048835		196618		196634
	131072		196619		196635

Füllmuster	Wert	Füllmuster	Wert	Füllmuster	Wert
	131073		196620		196636
	131074		196621		196637
	131075		196622		196638
	131076		196623		196639
	196608		196624		196640
	196609		196625		196641
	196610		196626		196642

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechtecks zu. In diesem Beispiel wird das Füllmuster auf den Wert "196642" gesetzt:

```
Sub RectangleConfiguration()
'VBA496
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.FillStyle = 196642
End With
End Sub
```

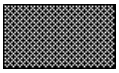
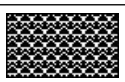
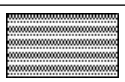
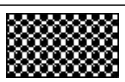
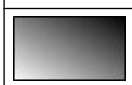
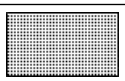
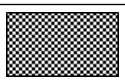
Siehe auch

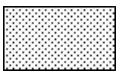
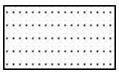
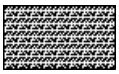
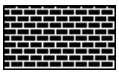
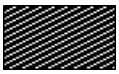
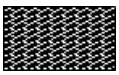
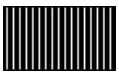
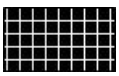
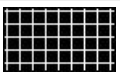
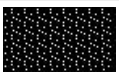
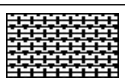
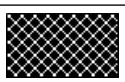
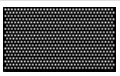
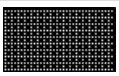
TextList-Objekt (Seite 2037)
StaticText-Objekt (Seite 2029)
Slider-Objekt (Seite 2025)
RoundRectangle-Objekt (Seite 2018)
RoundButton-Objekt (Seite 2015)
Rectangle-Objekt (Seite 2012)
Polygon-Objekt (Seite 1998)
PieSegment-Objekt (Seite 1995)
OptionGroup-Objekt (Seite 1989)
IOField-Objekt (Seite 1959)
GraphicObject-Objekt (Seite 1943)
EllipseSegment-Objekt (Seite 1932)
Ellipse-Objekt (Seite 1926)
Document-Objekt (Seite 1920)
Circle-Objekt (Seite 1902)
CheckBox-Objekt (Seite 1901)
Button-Objekt (Seite 1898)
BarGraph-Objekt (Seite 1893)

FillStyle2-Eigenschaft

Beschreibung

Legt das Füllmuster des Balkens für das Objekt BarGraph fest oder gibt es zurück.

Füllmuster	Wert	Füllmuster	Wert	Füllmuster	Wert
< Transparent >	65536				
< Massiv >	0				
	1048576		196611		196627
	1048577		196612		196628
	1048578		196613		196629
	1048579		196614		196630

Füllmuster	Wert	Füllmuster	Wert	Füllmuster	Wert
	1048832		196615		196631
	1048833		196616		196632
	1048834		196617		196633
	1048835		196618		196634
	131072		196619		196635
	131073		196620		196636
	131074		196621		196637
	131075		196622		196638
	131076		196623		196639
	196608		196624		196640
	196609		196625		196641
	196610		196626		196642

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird das Balkenmuster auf "196642" gesetzt:

```
Sub BarGraphConfiguration()
'VBA497
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.FillStyle2 = 196642
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

FillStyleAlignment-Eigenschaft

Beschreibung

Legt die Ausrichtung des Füllmusters für das Prozessbild fest.

Normal	Das Füllmuster bezieht sich auf das Prozessbild. In Runtime wird es beim Aufziehen des Bildes nicht skaliert.
gestreckt (Fenster)	Das Füllmuster bezieht sich auf das Fenster im Graphics Designer. In Runtime wird es beim Aufziehen des Bildes skaliert.

FlashBackColor-Eigenschaft

Beschreibung

TRUE, wenn das Blinken des Hintergrunds des Objektes aktiviert ist. BOOLEAN Schreib-Lese-Zugriff

Hinweis

Eine Änderung des Attributs deaktiviert nicht automatisch das Attribut "Windows-Stil".

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel wird das Blinken des Hintergrundes aktiviert:

```
Sub RectangleConfiguration()
'VBA498
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.FlashBackColor = True
End With
End Sub
```

Siehe auch

RoundButton-Objekt (Seite 2015)
StaticText-Objekt (Seite 2029)
Slider-Objekt (Seite 2025)
TextList-Objekt (Seite 2037)
RoundRectangle-Objekt (Seite 2018)
Rectangle-Objekt (Seite 2012)
Polygon-Objekt (Seite 1998)
PieSegment-Objekt (Seite 1995)
OptionGroup-Objekt (Seite 1989)
GraphicObject-Objekt (Seite 1943)
IOField-Objekt (Seite 1959)
EllipseSegment-Objekt (Seite 1932)
Ellipse-Objekt (Seite 1926)
Circle-Objekt (Seite 1902)
CheckBox-Objekt (Seite 1901)
Button-Objekt (Seite 1898)
BarGraph-Objekt (Seite 1893)

FlashBorderColor-Eigenschaft

Beschreibung

TRUE, wenn das Blinken der Linie des Objektes aktiviert ist. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel wird das Blinken der Linie aktiviert:

```
Sub RectangleConfiguration()
'VBA499
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.FlashBorderColor = True
End With
End Sub
```

Siehe auch

StaticText-Objekt (Seite 2029)
StatusDisplay-Objekt (Seite 2032)
Slider-Objekt (Seite 2025)
TextList-Objekt (Seite 2037)
RoundRectangle-Objekt (Seite 2018)
RoundButton-Objekt (Seite 2015)
Rectangle-Objekt (Seite 2012)
PolyLine-Objekt (Seite 2001)
Polygon-Objekt (Seite 1998)
PieSegment-Objekt (Seite 1995)
OptionGroup-Objekt (Seite 1989)
Line-Objekt (Seite 1970)
GraphicObject-Objekt (Seite 1943)
IOField-Objekt (Seite 1959)
EllipseSegment-Objekt (Seite 1932)
EllipseArc-Objekt (Seite 1929)
Ellipse-Objekt (Seite 1926)
CircularArc-Objekt (Seite 1905)
Circle-Objekt (Seite 1902)
CheckBox-Objekt (Seite 1901)
Button-Objekt (Seite 1898)

FlashFlashPicture-Eigenschaft

Beschreibung

TRUE, wenn das Blinken des Blinkbildes aktiviert ist. BOOLEAN Schreib-Lese-Zugriff

Beispiel

Die Prozedur "StatusDisplayConfiguration()" greift auf die Eigenschaften der Zustandsanzeige zu. In diesem Beispiel wird das Blinken des Blinkbildes aktiviert:

```
Sub StatusDisplayConfiguration()
'VBA500
Dim objsDisplay As HMIStatusDisplay
```

```
Set objsDisplay = ActiveDocument.HMIObjects.AddHMIObject("StatusDisplay1",
"HMIStatusDisplay")
With objsDisplay
.FlashFlashPicture = True
End With
End Sub
```

Siehe auch

StatusDisplay-Objekt (Seite 2032)

FlashForeColor-Eigenschaft

Beschreibung

TRUE, wenn das Blinken des Textes aktiviert ist. BOOLEAN Schreib-Lese-Zugriff.

Hinweis

Eine Änderung des Attributs deaktiviert nicht automatisch das Attribut "Windows-Stil"

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel wird das Blinken des Textes aktiviert:

```
Sub ButtonConfiguration()
'VBA501
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.FlashForeColor = True
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

IOField-Objekt (Seite 1959)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

FlashPicReferenced-Eigenschaft

Beschreibung

TRUE, wenn das zugeordnete Blinkbild im Objekt Zustandsanzeige gespeichert werden soll. Sonst wird nur der zugehörige Objektverweis gespeichert. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "StatusDisplayConfiguration()" greift auf die Eigenschaften der Zustandsanzeige zu. In diesem Beispiel soll das zugeordnete Bild im Objekt Zustandsanzeige gespeichert werden:

```
Sub StatusDisplayConfiguration()
'VBA502
Dim objStatusDisplay As HMIStatusDisplay
Set objStatusDisplay = ActiveDocument.HMIObjects.AddHMIObject("StatusDisplay1",
"HMIStatusDisplay")
With objStatusDisplay
.FlashPicReferenced = True
End With
End Sub
```

Siehe auch

StatusDisplay-Objekt (Seite 2032)

FlashPicTransColor-Eigenschaft

Beschreibung

Legt fest, welche Farbe des dem Blinkbild zugeordneten Bitmap-Objekts (.bmp, .dib) auf "transparent" gesetzt werden soll oder gibt die Farbe zurück. LONG Schreib-Lese-Zugriff.

Die Farbe wird nur dann auf "transparent" gesetzt, wenn die Eigenschaft "FlashPicUseTransColor" den Wert "True" hat.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "StatusDisplayConfiguration()" greift auf die Eigenschaften der Zustandsanzeige zu. In diesem Beispiel wird die Farbe "Gelb" auf "transparent" gesetzt:

```
Sub StatusDisplayConfiguration()
'VBA503
Dim objStatusDisplay As HMIStatusDisplay
Set objStatusDisplay = ActiveDocument.HMIObjects.AddHMIObject("StatusDisplay1",
"HMIStatusDisplay")
With objStatusDisplay
.FlashPicTransColor = RGB(255, 255, 0)
.FlashPicUseTransColor = True
End With
End Sub
```

Siehe auch

FlashPicUseTransColor-Eigenschaft (Seite 2191)

StatusDisplay-Objekt (Seite 2032)

FlashPicture-Eigenschaft

Beschreibung

Legt das Blinkbild für das Objekt Zustandsanzeige fest oder gibt es zurück.

Das Bild (*.BMP oder *.DIB) muss sich im Verzeichnis "GraCS" des aktuellen Projekts befinden, damit es eingebunden werden kann.

Die Eigenschaft "FlashPicReferenced" legt in diesem Zusammenhang fest, ob das Blinkbild zusammen mit dem Objekt Zustandsanzeige gespeichert oder referenziert wird.

Beispiel

Die Prozedur "StatusDisplayConfiguration()" greift auf die Eigenschaften der Zustandsanzeige zu. In diesem Beispiel wird das Bild "Testbild.BMP" als Blinkbild verwendet :

```
Sub StatusDisplayConfiguration()
'VBA504
Dim objStatusDisplay As HMIStatusDisplay
Set objStatusDisplay = ActiveDocument.HMIObjects.AddHMIObject("StatusDisplay1",
"HMIStatusDisplay")
With objStatusDisplay
'
'To use this example copy a Bitmap-Graphic
'to the "GraCS"-Folder of the actual project.
'Replace the picturename "Testpicture.BMP" with the name of
'the picture you copied
```

```
.FlashPicture = "Testpicture.BMP"
End With
End Sub
```

Siehe auch

FlashPicReferenced-Eigenschaft (Seite 2189)

StatusDisplay-Objekt (Seite 2032)

FlashPicUseTransColor-Eigenschaft

Beschreibung

TRUE, wenn die projektierte Farbe ("FlashPicTransColor"-Eigenschaft) des dem Blinkbild zugeordneten Bitmap-Objektes auf "transparent" gesetzt werden soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "StatusDisplayConfiguration()" greift auf die Eigenschaften der Zustandsanzeige zu. In diesem Beispiel wird die Farbe "Gelb" auf "transparent" gesetzt:

```
Sub StatusDisplayConfiguration()
'VBA505
Dim objStatusDisplay As HMIStatusDisplay
Set objStatusDisplay = ActiveDocument.HMIObjects.AddHMIObject("StatusDisplay1",
"HMIStatusDisplay")
With objStatusDisplay
.FlashPicTransColor = RGB(255, 255, 0)
.FlashPicUseTransColor = True
End With
End Sub
```

Siehe auch

FlashPicTransColor-Eigenschaft (Seite 2189)

StatusDisplay-Objekt (Seite 2032)

FlashRate-Eigenschaft

Beschreibung

Legt die Blinkfrequenz des Objektes GroupDisplay fest oder gibt sie zurück. Wertebereich von 0 bis 2.

Blinkfrequenz	zugeordneter Wert
Langsam (ca. 0,5 Hz)	0
Mittel (ca. 2 Hz)	1
Schnell (ca. 8 Hz)	2

Hinweis

Da es sich beim Blinken um eine softwaretechnische Realisierung handelt, ist die Frequenz system- und hardwareabhängig (Anzahl der Objekte, Prozessor, Speicher, Aktualisierungszeitraster u.a.m).

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel wird die Blinkfrequenz auf "mittel" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA506
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.FlashRate = 1
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

FlashRateBackColor-Eigenschaft

Beschreibung

Legt die Blinkfrequenz für den Objekthintergrund fest oder gibt sie zurück. Wertebereich von 0 bis 2.

Blinkfrequenz	zugeordneter Wert
Langsam (ca. 0,5 Hz)	0
Mittel (ca. 2 Hz)	1
Schnell (ca. 8 Hz)	2

Hinweis

Da es sich beim Blinken um eine softwaretechnische Realisierung handelt, ist die Frequenz system- und hardwareabhängig (Anzahl der Objekte, Prozessor, Speicher, Aktualisierungszeitraster u.a.m).

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel wird die Blinkfrequenz des Hintergrundes auf "mittel" gesetzt:

```
Sub ButtonConfiguration()
'VBA507
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.FlashRateBackColor = 1
End With
End Sub
```

Siehe auch

StaticText-Objekt (Seite 2029)
Slider-Objekt (Seite 2025)
TextList-Objekt (Seite 2037)
RoundRectangle-Objekt (Seite 2018)
RoundButton-Objekt (Seite 2015)
Rectangle-Objekt (Seite 2012)
Polygon-Objekt (Seite 1998)
PieSegment-Objekt (Seite 1995)
OptionGroup-Objekt (Seite 1989)
GraphicObject-Objekt (Seite 1943)
IOField-Objekt (Seite 1959)
EllipseSegment-Objekt (Seite 1932)
Ellipse-Objekt (Seite 1926)
Circle-Objekt (Seite 1902)
CheckBox-Objekt (Seite 1901)
Button-Objekt (Seite 1898)
BarGraph-Objekt (Seite 1893)

FlashRateBorderColor-Eigenschaft

Beschreibung

Legt die Blinkfrequenz für die Linie des Objektes fest oder gibt sie zurück. Wertebereich von 0 bis 2.

Blinkfrequenz	zugeordneter Wert
Langsam (ca. 0,5 Hz)	0
Mittel (ca. 2 Hz)	1
Schnell (ca. 8 Hz)	2

Hinweis

Da es sich beim Blinken um eine softwaretechnische Realisierung handelt, ist die Frequenz system- und hardwareabhängig (Anzahl der Objekte, Prozessor, Speicher, Aktualisierungszeitraster u.a.m).

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel wird die Blinkfrequenz der Linie auf "mittel" gesetzt:

```
Sub ButtonConfiguration()
'VBA508
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.FlashRateBorderColor = 1
End With
End Sub
```

Siehe auch

Slider-Objekt (Seite 2025)

StatusDisplay-Objekt (Seite 2032)

StaticText-Objekt (Seite 2029)

TextList-Objekt (Seite 2037)

RoundRectangle-Objekt (Seite 2018)

RoundButton-Objekt (Seite 2015)

Rectangle-Objekt (Seite 2012)

PolyLine-Objekt (Seite 2001)

Polygon-Objekt (Seite 1998)

PieSegment-Objekt (Seite 1995)

OptionGroup-Objekt (Seite 1989)

Line-Objekt (Seite 1970)

GraphicObject-Objekt (Seite 1943)

IOField-Objekt (Seite 1959)

EllipseSegment-Objekt (Seite 1932)

EllipseArc-Objekt (Seite 1929)

Ellipse-Objekt (Seite 1926)

CircularArc-Objekt (Seite 1905)

Circle-Objekt (Seite 1902)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

FlashRateFlashPic-Eigenschaft

Beschreibung

Legt die Blinkfrequenz für die Statusanzeige fest oder gibt sie zurück. Wertebereich von 0 bis 2.

Blinkfrequenz	zugeordneter Wert
Langsam (ca. 0,5 Hz)	0
Mittel (ca. 2 Hz)	1
Schnell (ca. 8 Hz)	2

Hinweis

Da es sich beim Blinken um eine softwaretechnische Realisierung handelt, ist die Frequenz system- und hardwareabhängig (Anzahl der Objekte, Prozessor, Speicher, Aktualisierungszeitraster u.a.m).

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Statusanzeige zu. In diesem Beispiel wird die Blinkfrequenz des Blinkbildes auf "mittel" gesetzt:

```
Sub StatusDisplayConfiguration()
'VBA509
Dim objStatusDisplay As HMIStatusDisplay
Set objStatusDisplay = ActiveDocument.HMIObjects.AddHMIObject("StatusDisplay1",
"HMIStatusDisplay")
With objStatusDisplay
.FlashRateFlashPic = 1
End With
End Sub
```

Siehe auch

StatusDisplay-Objekt (Seite 2032)

FlashRateForeColor-Eigenschaft

Beschreibung

Legt die Blinkfrequenz für die Beschriftung des Objektes fest oder gibt sie zurück. Wertebereich von 0 bis 2.

Blinkfrequenz	zugeordneter Wert
Langsam (ca. 0,5 Hz)	0
Mittel (ca. 2 Hz)	1
Schnell (ca. 8 Hz)	2

Hinweis

Da es sich beim Blinken um eine softwaretechnische Realisierung handelt, ist die Frequenz system- und hardwareabhängig (Anzahl der Objekte, Prozessor, Speicher, Aktualisierungszeitraster u.a.m).

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel wird die Blinkfrequenz der Beschriftung auf "mittel" gesetzt:

```
Sub ButtonConfiguration()
'VBA510
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.FlashRateForeColor = 1
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

IOField-Objekt (Seite 1959)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

Folder-Eigenschaft

Beschreibung

Gibt einen Ordner aus der Bausteinbibliothek zurück.

Beispiel

Die Prozedur "ShowFolderItems()" greift auf die Symbolbibliotheken zu. In diesem Beispiel werden alle Ordnernamen der globalen Symbolbibliothek und Projektsymbolbibliothek ausgegeben:

```
Sub ShowFolderItems()
'VBA511
Dim colFolderItems As HMIFolderItems
Dim objFolderItem As HMIFolderItem
Dim iAnswer As Integer
Dim iMaxFolder As Integer
Dim iMaxSymbolLib As Integer
Dim iSymbolLibIndex As Integer
Dim iSubFolderIndex As Integer
Dim strSubFolderName As String
Dim strFolderItemName As String
'To determine the number of symbollibraries:
iMaxSymbolLib = Application.SymbolLibraries.Count
iSymbolLibIndex = 1
For iSymbolLibIndex = 1 To iMaxSymbolLib
With Application.SymbolLibraries(iSymbolLibIndex)
Set colFolderItems = .FolderItems
'
'To determine the number of folders in actual symbollibrary:
iMaxFolder = .FolderItems.Count
MsgBox "Number of FolderItems in " & .Name & " : " & iMaxFolder
'
'Output of all subfoldernames from actual folder:
For Each objFolderItem In colFolderItems
  iSubFolderIndex = 1
  For iSubFolderIndex = 1 To iMaxFolder
    strFolderItemName = objFolderItem.DisplayName
    If 0 <> objFolderItem.Folder.Count Then
      strSubFolderName = objFolderItem.Folder(iSubFolderIndex).DisplayName
      iAnswer = MsgBox("SymbolLibrary: " & .Name & vbCrLf & "act. Folder: " &
strFolderItemName & vbCrLf & "act. Subfolder: " & strSubFolderName, vbOKCancel)
      '
      'If "Cancel" is clicked, continued with next FolderItem
      If vbCancel = iAnswer Then
        Exit For
      End If
    Else
      MsgBox "There are no subfolders in " & objFolderItem.DisplayName
      Exit For
    End If
  Next iSubFolderIndex
```

```
Next objFolderItem
End With
Next iSymbolLibIndex
End Sub
```

Siehe auch

SymbolLibraries-Objekt (Auflistung) (Seite 2036)

SymbolLibrary-Objekt (Seite 2035)

FolderItems-Objekt (Auflistung) (Seite 1941)

FolderItem-Objekt (Seite 1939)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

FolderItems-Eigenschaft

Beschreibung

Gibt eine Auflistung zurück, die alle Ordner der Symbolbibliothek enthält.

Beispiel

Die Prozedur "ShowFolderItems()" greift auf die Symbolbibliotheken zu. In diesem Beispiel werden alle Ordnernamen der globalen Symbolbibliothek und Projektsymbolbibliothek ausgegeben:

```
Sub ShowFolderItems()
'VBA512
Dim colFolderItems As HMIFolderItems
Dim objFolderItem As HMIFolderItem
Dim iAnswer As Integer
Dim iMaxFolder As Integer
Dim iMaxSymbolLib As Integer
Dim iSymbolLibIndex As Integer
Dim iSubFolderIndex As Integer
Dim strSubFolderName As String
Dim strFolderItemName As String
'To determine the number of symbollibraries:
iMaxSymbolLib = Application.SymbolLibraries.Count
iSymbolLibIndex = 1
For iSymbolLibIndex = 1 To iMaxSymbolLib
With Application.SymbolLibraries(iSymbolLibIndex)
Set colFolderItems = .FolderItems
'
'To determine the number of folders in actual symbollibrary:
iMaxFolder = .FolderItems.Count
MsgBox "Number of FolderItems in " & .Name & " : " & iMaxFolder
'
```

```
'Output of all subfoldernames from actual folder:
For Each objFolderItem In colFolderItems
  iSubFolderIndex = 1
  For iSubFolderIndex = 1 To iMaxFolder
    strFolderItemName = objFolderItem.DisplayName
    If 0 <> objFolderItem.Folder.Count Then
      strSubFolderName = objFolderItem.Folder(iSubFolderIndex).DisplayName
      iAnswer = MsgBox("SymbolLibrary: " & .Name & vbCrLf & "act. Folder: " &
strFolderItemName & vbCrLf & "act. Subfolder: " & strSubFolderName, vbOKCancel)
      '
      'If "Cancel" is clicked, continued with next FolderItem
      If vbCancel = iAnswer Then
        Exit For
      End If
    Else
      MsgBox "There are no subfolders in " & objFolderItem.DisplayName
      Exit For
    End If
  Next iSubFolderIndex
Next objFolderItem
End With
Next iSymbolLibIndex
End Sub
```

Siehe auch

FolderItem-Objekt (Seite 1939)

SymbolLibraries-Objekt (Auflistung) (Seite 2036)

SymbolLibrary-Objekt (Seite 2035)

FolderItems-Objekt (Auflistung) (Seite 1941)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

FontBold-Eigenschaft

Beschreibung

TRUE, wenn der Text im Objekt das Schriftattribut "fett" erhält. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel das Schriftattribut auf "fett" gesetzt:

```
Sub ButtonConfiguration()
'VBA513
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.FontBold = True
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

IOField-Objekt (Seite 1959)

GroupDisplay-Objekt (Seite 1947)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

BarGraph-Objekt (Seite 1893)

FontFamily-Eigenschaft

Beschreibung

Legt die sprachabhängige Schriftart fest oder gibt sie zurück.

Beispiel

Im folgenden Beispiel werden für einen Button die Schriftattribute für Französisch und Englisch gesetzt:

```
Sub ExampleForLanguageFonts()
'VBA492
Dim colLangFonts As HMILanguageFonts
Dim objButton As HMIButton
Dim iStartLangID As Integer
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
iStartLangID = Application.CurrentDataLanguage
With objButton
.Text = "Command"
```

```
.Width = 100
End With
Set colLangFonts = objButton.LDFonts
'
'To do typesettings for french:
With colLangFonts.ItemByLCID(1036)
.Family = "Courier New"
.Bold = True
.Italic = False
.Underlined = True
.Size = 12
End With
'
'To do typesettings for english:
With colLangFonts.ItemByLCID(1033)
.Family = "Times New Roman"
.Bold = False
.Italic = True
.Underlined = False
.Size = 14
End With
With objButton
Application.CurrentDataLanguage = 1036
.Text = "Command"
MsgBox "Datalanguage is changed in french"
Application.CurrentDataLanguage = 1033
.Text = "Command"
MsgBox "Datalanguage is changed in english"
Application.CurrentDataLanguage = iStartLangID
MsgBox "Datalanguage is changed back to startlanguage."
End With
End Sub
```

Siehe auch

Underlined-Eigenschaft (Seite 2402)

Size-Eigenschaft (Seite 2368)

Parent-Eigenschaft (Seite 2317)

Italic-Eigenschaft (Seite 2224)

LanguageID-Eigenschaft (Seite 2232)

Bold-Eigenschaft (Seite 2106)

Application-Eigenschaft (Seite 2079)

LanguageFont-Objekt (Seite 1962)

FontItalic-Eigenschaft

Beschreibung

TRUE, wenn der Text im Objekt das Schriftattribut "kursiv" erhält. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel das Schriftattribut auf "kursiv" gesetzt:

```
Sub ButtonConfiguration()
'VBA514
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.FontItalic = True
End With
End Sub
```

Siehe auch

StaticText-Objekt (Seite 2029)

TextList-Objekt (Seite 2037)

OptionGroup-Objekt (Seite 1989)

IOField-Objekt (Seite 1959)

GroupDisplay-Objekt (Seite 1947)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

BarGraph-Objekt (Seite 1893)

FontName-Eigenschaft

Beschreibung

Legt die Schriftart des Textes im Objekt fest oder gibt sie zurück.

Dabei stehen Ihnen alle in Windows installierten Schriftarten zur Verfügung.

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel als Schriftart Arial gesetzt:

```
Sub ButtonConfiguration()
'VBA515
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.FontName = "Arial"
End With
End Sub
```

Siehe auch

CheckBox-Objekt (Seite 1901)

TextList-Objekt (Seite 2037)

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

IOField-Objekt (Seite 1959)

GroupDisplay-Objekt (Seite 1947)

Button-Objekt (Seite 1898)

BarGraph-Objekt (Seite 1893)

FontSize-Eigenschaft

Beschreibung

Legt die Schriftgröße des Textes im Objekt in Punkt fest oder gibt sie zurück.

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel die Schriftgröße auf 10 Punkt gesetzt:

```
Sub ButtonConfiguration()
'VBA516
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.FONTSIZE = 10
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

IOField-Objekt (Seite 1959)

GroupDisplay-Objekt (Seite 1947)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

BarGraph-Objekt (Seite 1893)

FontUnderline-Eigenschaft

Beschreibung

TRUE, wenn der Text im Objekt das Schriftattribut "unterstrichen" erhält. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel das Schriftattribut auf "unterstrichen" gesetzt:

```
Sub ButtonConfiguration()
'VBA517
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.FontUnderline = True
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

IOField-Objekt (Seite 1959)

GroupDisplay-Objekt (Seite 1947)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

BarGraph-Objekt (Seite 1893)

ForeColor-Eigenschaft

Beschreibung

Legt die Farbe des Textes im Objekt fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel die Schriftfarbe auf "rot" gesetzt:

```
Sub ButtonConfiguration()
'VBA518
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.ForeColor = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

Button-Objekt (Seite 1898)

TextList-Objekt (Seite 2037)

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

IOField-Objekt (Seite 1959)

GroupDisplay-Objekt (Seite 1947)

CheckBox-Objekt (Seite 1901)

BarGraph-Objekt (Seite 1893)

ForeFlashColorOff-Eigenschaft

Beschreibung

Legt die Farbe des Textes für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel die Schriftfarbe für den Blinkzustand "Aus" auf "rot" gesetzt:

```
Sub ButtonConfiguration()
'VBA519
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.ForeFlashColorOff = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

CheckBox-Objekt (Seite 1901)

TextList-Objekt (Seite 2037)

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

IOField-Objekt (Seite 1959)

GroupDisplay-Objekt (Seite 1947)

Button-Objekt (Seite 1898)

BarGraph-Objekt (Seite 1893)

ForeFlashColorOn-Eigenschaft

Beschreibung

Legt die Farbe des Textes für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel die Schriftfarbe für den Blinkzustand "Ein" auf "weiß" gesetzt:

```
Sub ButtonConfiguration()
'VBA520
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.ForeFlashColorOn = RGB(255, 255, 255)
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

IOField-Objekt (Seite 1959)

GroupDisplay-Objekt (Seite 1947)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

BarGraph-Objekt (Seite 1893)

G-H

GlobalColorScheme-Eigenschaft

Beschreibung

Legt fest, ob die im globalen Farbschema des aktuellen Designs definierten Farben für dieses Objekt verwendet werden.

ja	Übernimmt die Farben aus dem für diesen Objekttyp festgelegten globalen Farbschema.
nein	Übernimmt die Farben aus dem für diesen Objekttyp unter "Farben" eingestellten eigenen Farbschema.

Beispiel

--

GlobalShadow-Eigenschaft

Beschreibung

Legt fest, ob das Objekt mit der im aktiven Design festgelegten Schattierung dargestellt wird.

ja	Übernimmt die für diesen Objekttyp festgelegte globale Schattierung.
nein	Kein Schatten.

Beispiel

--

Grid-Eigenschaft

Beschreibung

TRUE, wenn beim aktiven Bild das Raster eingeschaltet ist. BOOLEAN Schreib-Lese-Zugriff.

Das Raster ist nur während der Projektierungsphase sichtbar.

Beispiel

Die Prozedur "ActiveDocumentConfiguration()" greift auf die Eigenschaften des aktuellen Bildes im Graphics Designer zu. In diesem Beispiel wird das Raster für das aktive Bild eingeschaltet:

```
Sub ActiveDocumentConfiguration()
'VBA521
Application.ActiveDocument.Grid = True
End Sub
```

Siehe auch

GridWidth-Eigenschaft (Seite 2211)

GridHeight-Eigenschaft (Seite 2210)

GridColor-Eigenschaft (Seite 2210)

ActiveDocument-Eigenschaft (Seite 2067)

Application-Eigenschaft (Seite 2079)

Document-Objekt (Seite 1920)

Application-Objekt (Seite 1888)

GridColor-Eigenschaft

Beschreibung

Legt während der Projektierungsphase die Farbe des Rasters fest oder gibt sie zurück. Die Eigenschaft Grid muss auf TRUE gesetzt sein, damit das Raster angezeigt wird. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "ActiveDocumentConfiguration()" greift auf die Eigenschaften des aktuellen Bildes im Graphics Designer zu. In diesem Beispiel wird die Rasterfarbe für das aktive Bild auf "Blau" gesetzt:

```
Sub ActiveDocumentConfiguration()
'VBA522
Application.ActiveDocument.Grid = True
Application.ActiveDocument.GridColor = RGB(0, 0, 255)
End Sub
```

Siehe auch

Grid-Eigenschaft (Seite 2209)

ActiveDocument-Eigenschaft (Seite 2067)

Application-Eigenschaft (Seite 2079)

Document-Objekt (Seite 1920)

Application-Objekt (Seite 1888)

GridHeight-Eigenschaft

Beschreibung

Legt während der Projektierungsphase die Rasterhöhe (in Pixel) im aktuellen Bild fest oder gibt sie zurück. Die Eigenschaft Grid muss auf TRUE gesetzt sein, damit das Raster angezeigt wird.

Beispiel

Die Prozedur "ActiveDocumentConfiguration()" greift auf die Eigenschaften des aktuellen Bildes im Graphics Designer zu. In diesem Beispiel wird die Rasterhöhe für das aktive Bild auf "8" gesetzt:

```
Sub ActiveDocumentConfiguration()
'VBA523
Application.ActiveDocument.Grid = True
Application.ActiveDocument.GridHeight = 8
End Sub
```

Siehe auch

GridWidth-Eigenschaft (Seite 2211)

Grid-Eigenschaft (Seite 2209)

ActiveDocument-Eigenschaft (Seite 2067)

Application-Eigenschaft (Seite 2079)

Document-Objekt (Seite 1920)

Application-Objekt (Seite 1888)

GridWidth-Eigenschaft

Beschreibung

Legt während der Projektierungsphase die Rasterweite (in Pixel) im aktuellen Bild fest oder gibt sie zurück. Die Eigenschaft Grid muss auf TRUE gesetzt sein, damit das Raster angezeigt wird.

Beispiel

Die Prozedur "ActiveDocumentConfiguration()" greift auf die Eigenschaften des aktuellen Bildes im Graphics Designer zu. In diesem Beispiel wird die Rasterweite für das aktive Bild auf "8" gesetzt:

```
Sub ActiveDocumentConfiguration()
'VBA524
Application.ActiveDocument.Grid = True
Application.ActiveDocument.GridWidth = 8
End Sub
```

Siehe auch

Grid-Eigenschaft (Seite 2209)
GridHeight-Eigenschaft (Seite 2210)
ActiveDocument-Eigenschaft (Seite 2067)
Application-Eigenschaft (Seite 2079)
Document-Objekt (Seite 1920)
Application-Objekt (Seite 1888)

GroupParent-Eigenschaft

Beschreibung

Gibt das übergeordnete Objekt des angegebenen Gruppen-Objektes zurück. Nur-Lese-Zugriff.

Beispiel

--

Siehe auch

Group-Objekt (Seite 1946)
ActiveDocument-Eigenschaft (Seite 2067)
GroupedObjects-Objekt (Auflistung) (Seite 1950)
Document-Objekt (Seite 1920)
Application-Objekt (Seite 1888)

GroupedHMIObjects-Eigenschaft

Beschreibung

Gibt eine Auflistung zurück, die alle Objekte der aktuellen Gruppe enthält.

Beispiel

In diesem Beispiel wird aus Objekten das Gruppen-Objekt "Group1" erzeugt. Danach wird dem Gruppen-Objekt ein Ellipsensegment hinzugefügt:

```
Sub CreateGroup()
'VBA526
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
```

```
Dim objEllipseSegment As HMIEllipseSegment
Dim objGroup As HMIGroup
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
Set objGroup = ActiveDocument.Selection.CreateGroup
objGroup.ObjectName = "Group1"
Set objEllipseSegment = ActiveDocument.HMIObjects.AddHMIObject("EllipseSegment",
"HMIEllipseSegment")
'
'Add one object to the existing group
objGroup.GroupedHMIObjects.Add ("EllipseSegment")
End Sub
```

Siehe auch

Group-Objekt (Seite 1946)

Height-Eigenschaft

Beschreibung

Legt die Höhe des Objektes (Document, View, Object) in Pixel fest oder gibt sie zurück.

Anmerkung für die Objekte Document und View:

Der voreingestellte Wert entspricht der betriebssystemseitig eingestellten Bildschirmauflösung in vertikaler Richtung. Die Angabe kann größer sein als die aktuelle Bildschirmauflösung. Das Bild kann dann über Rollbalken verschoben werden.

Die maximal einstellbare Bildhöhe beträgt 10000 Pixel.

Beispiel

Die Prozedur "ActiveDocumentConfiguration()" greift auf die Eigenschaften des aktuellen Bildes im Graphics Designer zu. In diesem Beispiel wird die Höhe des aktuellen Bildes auf "1600" gesetzt:

```
Sub ActiveDocumentConfiguration()
'VBA527
```

```
Application.ActiveDocument.Height = 1600
End Sub
```

Siehe auch

View-Objekt (Seite 2062)

HMIObject-Objekt (Seite 1955)

Document-Objekt (Seite 1920)

Hide-Eigenschaft

Beschreibung

TRUE, wenn das angegebene Bild "sichtbar" geöffnet ist. BOOLEAN Schreib-Lese-Zugriff.

Verwenden Sie die Hide-Eigenschaft, um z.B.zu prüfen, ob ein Bild sichtbar oder unsichtbar geöffnet wird. Andere WinCC-Editoren (z.B. CrossReference) öffnen Bilder unsichtbar, d.h. diese werden nicht im Graphics Designer angezeigt. Wenn Sie beispielsweise das DocumentOpened-Ereignis verwenden, könne Sie mit Hilfe der Hide-Eigenschaft die Ausführung des Codes im Ereignis verhindern, indem Sie die Hide-Eigenschaft auf FALSE prüfen.

Mit der Add- und der Open-Methode können Sie festlegen, ob ein Bild sichtbar oder unsichtbar geöffnet werden soll.

Hinweis

Wenn Sie ein Bild auf "unsichtbar" (Hide = FALSE) setzen, können Sie es nur noch über die Documents-Auflistung ansprechen. Im Graphics Designer ist das Bild nicht mehr verfügbar.

Beispiel

Im folgenden Beispiel wird beim Öffnen eines Bildes ausgegeben, ob es sichtbar oder unsichtbar geöffnet wurde:

```
Private Sub Document_Opened(CancelForwarding As Boolean)
'VBA802
MsgBox Me.Hide
End Sub
```

Siehe auch

Open-Methode (Seite 1853)

Add-Methode (Documents-Auflistung) (Seite 1780)

Document-Objekt (Seite 1920)

HiddenInput-Eigenschaft

Beschreibung

TRUE, wenn der Eingabewert während der Eingabe nicht angezeigt wird. Für jedes Zeichen wird ein * angezeigt. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel soll die Eingabe verdeckt erfolgen:

```
Sub IOFieldConfiguration()
'VBA528
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.HiddenInput = True
End With
End Sub
```

Siehe auch

IOField-Objekt (Seite 1959)

HMIObjects-Eigenschaft

Beschreibung

Gibt eine Auflistung zurück, die alle Objekte im angegebenen Bild enthält.

Um ein Element aus der HMIObjects-Auflistung zurückzugeben, können Sie entweder die Indexnummer oder den Objektnamen verwenden.

Beispiel

Verwenden Sie die Methode "AddHMIObject(ObjectName, ProgID)", um ein neues Objekt in ein Bild einzufügen. :

```
Sub AddCircle()
'VBA529
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("my Circle", "HMICircle")
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

Hotkey-Eigenschaft

Beschreibung

Legt beim Objekt Button die Funktionstaste für die Mausbedienung fest oder gibt sie zurück.

Funktionstaste	zugeordneter Wert
F1	112
F2	113
F3	114
F4	115
F5	116
F6	117
F7	118
F8	119
F9	120
F10	121
F11	122
F12	123

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel soll der Button zusätzlich mit der Funktionstaste "F5" ausgelöst werden können:

```
Sub ButtonConfiguration()
'VBA530
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.Hotkey = 116
End With
End Sub
```

Siehe auch

Button-Objekt (Seite 1898)

Hysteresis-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt BarGraph die Anzeige mit Hysterese erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel soll die Anzeige mit Hysterese erfolgen:

```
Sub BarGraphConfiguration()
'VBA531
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.Hysteresis = True
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

HysteresisRange-Eigenschaft

Beschreibung

Legt die Hysterese in % des Anzeigewertes fest oder gibt sie zurück

Die Eigenschaft Hysteresis muss den Wert TRUE haben, damit die Hysterese berechnet werden kann.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird die Hysterese auf "4%" gesetzt:

```
Sub BarGraphConfiguration()
'VBA532
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.Hysteresis = True
.HysteresisRange = 4
End With
```

```
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

Hysteresis-Eigenschaft (Seite 2217)

I - K

Icon-Eigenschaft

Beschreibung

Legt für die Schaltfläche einer benutzerdefinierten Symbolleiste das Symbol (*.ICO, vollständiger Pfad und Dateiname) fest oder gibt Pfad und Dateiname zurück.

Pfadangaben

Folgende Pfadangabeformate sind möglich:

- absolut: z.B. "C:\Siemens\WinCC\Icons\myIcon.ICO.
- relativ: Das Ausgangsverzeichnis für die relative Pfadangabe ist das "GraCS"-Verzeichnis des aktuellen Projektes.
- <global>: Bezeichnet den Installationspfad von WinCC. Die Pfadangabe "<global>\Icons \myIcon" entspricht der Pfadangabe unter "absolut".
- <project>: Bezeichnet das aktuelle Projektverzeichnis (siehe Beispiel).

Beispiel

Die Prozedur "CreateToolbar()" erzeugt eine benutzerdefinierte Symbolleiste mit zwei Symbolen:

```
Sub CreateToolbar()
'VBA533
Dim objToolbar As HMIToolbar
Dim objToolbarItem As HMIToolbarItem
Dim strFileWithPath
Set objToolbar = ActiveDocument.CustomToolbars.Add("Tool1_1")
Set objToolbarItem = objToolbar.ToolbarItems.InsertToolbarItem(1, "ti1_1",
"myFirstToolbaritem")
Set objToolbarItem = objToolbar.ToolbarItems.InsertToolbarItem(2, "ti1_2",
"mySecondToolbaritem")
'
'To use this example copy a *.ICO-Graphic
'to the "GraCS"-Folder of the actual project.
'Replace the filename "EZSTART.ICO" in the next commandline
```

```
'with the name of the ICO-Graphic you copied
strFileWithPath = Application.ApplicationDataPath & "EZSTART.ICO"
'
'To assign the symbol-icon to the first toolbaritem
objToolbar.ToolbarItems(1).Icon = strFileWithPath
End Sub
```

Siehe auch

ToolbarItems-Objekt (Auflistung) (Seite 2046)

ToolbarItem-Objekt (Seite 2043)

So fügen Sie ein neues Symbol zur Symbolleiste hinzu (Seite 1639)

So legen Sie eine neue anwendungsspezifische Symbolleiste an (Seite 1637)

IndependentWindow-Eigenschaft

Beschreibung

Legt fest, ob die Darstellung des Bildfensters in Runtime von dem Prozessbild abhängt, in dem das Bildfenster projektiert wurde.

ja	Größe und Position des Bildfensters sind unabhängig vom Prozessbild und nur durch das Attribut "Fenstermodus" festgelegt
nein	Größe und Position des Bildfensters ändern sich mit der Verschiebung oder Skalierung des Prozessbildes

Index-Eigenschaft

Beschreibung

Zustandsanzeige

Legt den Zustand (0 bis 255) fest oder gibt ihn zurück. Für jeden Zustandswert können Sie ein Grund- und ein Blinkbild angeben.

Line-Objekt

Legt den Anfangs- und Endpunkt für eine Linie und damit die Richtung fest. Für jeden Anfangs- und Endpunkt legen Sie mit den Eigenschaften ActualPointLeft und ActualPointTop die Koordinaten fest.

Polygon-Objekt, PolyLine-Objekt und TubePolyline-Objekt

Legt die Nummer des Eckpunkts fest, dessen Positionskoordinaten Sie ändern oder anzeigen lassen wollen, oder gibt sie zurück.

CheckBox- und OptionGroup-Objekt

Legt die Nummer (1 bis 32) des Feldes fest, dessen Text Sie definieren wollen, oder gibt sie zurück.

ComboBox- und ListBox-Objekt

Legt die Nummer (1 bis 32) der Zeile fest, deren Text Sie definieren wollen, oder gibt sie zurück.

Beispiel 1: Linie

Im folgenden Beispiel wird eine Linie in das aktive Bild eingefügt und der Anfangs- und Endpunkt festgelegt:

```
Sub LineAdd()
'VBA682
Dim objLine As HMILine
Dim objEvent As HMIEvent
Set objLine = ActiveDocument.HMIObjects.AddHMIObject("myLine", "HMILine")
With objLine
.BorderColor = RGB(255, 0, 0)
.index = hmiLineIndexTypeStartPoint
.ActualPointLeft = 12
.ActualPointTop = 34
.index = hmiLineIndexTypeEndPoint
.ActualPointLeft = 74
.ActualPointTop = 64
End With
End Sub
```

Beispiel 2: Polygonzug

Damit dieses Beispiel funktioniert, fügen Sie einen Polygonzug "Polygonzug1" in das aktive Bild ein. Die Prozedur "PolyLineCoordsOutput" gibt dann die Koordinaten aller Eckpunkte des Polygonzuges aus:

```
Sub PolyLineCoordsOutput()
'VBA534
Dim iPcIndex As Integer
Dim iPosX As Integer
Dim iPosY As Integer
Dim iIndex As Integer
Dim objPolyLine As HMIPolyLine
Set objPolyLine = Application.ActiveDocument.HMIObjects.AddHMIObject("PolyLine1",
"HMIPolyLine")
'
'Determine number of corners from "PolyLine1":
iPcIndex = objPolyLine.PointCount
'
'Output of x/y-coordinates from every corner:
For iIndex = 1 To iPcIndex
With objPolyLine
```

```
.index = iIndex
iPosX = .ActualPointLeft
iPosY = .ActualPointTop
MsgBox iIndex & ". corner:" & vbCrLf & "x-coordinate: " & iPosX & vbCrLf & "y-coordinate:
" & iPosY
End With
Next iIndex
End Sub
```

Beispiel 3: Check-Box

Die Prozedur "CreateOptionGroup()" erzeugt das Objekt OptionGroup mit vier Optionsfeldern. Jedes Optionsfeld wird mit dem Standardnamen "myCustomText<Nummer>"belegt:

```
Sub CreateOptionGroup()
'VBA535
Dim objRadioBox As HMIOptionGroup
Dim iIndex As Integer
Set objRadioBox = ActiveDocument.HMIObjects.AddHMIObject("RadioBox_1", "HMIOptionGroup")
With objRadioBox
.Height = 100
.Width = 180
.BoxCount = 4
For iIndex = 1 To .BoxCount
.index = iIndex
.Text = "myCustomText" & .index
Next iIndex
End With
End Sub
```

Siehe auch

Line-Objekt (Seite 1970)

FlashPicture-Eigenschaft (Seite 2190)

BasePicture-Eigenschaft (Seite 2100)

ActualPointTop-Eigenschaft (Seite 2069)

ActualPointLeft-Eigenschaft (Seite 2068)

StatusDisplay-Objekt (Seite 2032)

PolyLine-Objekt (Seite 2001)

Polygon-Objekt (Seite 1998)

OptionGroup-Objekt (Seite 1989)

InheritState-Eigenschaft

Beschreibung

Legt fest, ob die Eigenschaften "Anzeige" und "Bedienfreigabe" des Anwenderobjekts für die einzelnen Objekte des Anwenderobjekts vererbbar sind.

InputValue-Eigenschaft

Beschreibung

Definiert den im EA-Feld vom Benutzer eingegebene Wert. Der Wert wird beim Setzen der Eigenschaft nicht im EA-Feld angezeigt.

Wenn Sie wollen, dass der eingegebene Wert nach Bestätigung mit der Taste <Return> im EA-Feld angezeigt wird, projektieren Sie eine Direktverbindung zwischen den Eigenschaften "Eingabewert" und "Ausgabewert". Die Direktverbindung ist nur sinnvoll, wenn am Ausgabewert keine Variablenverbindung projektiert ist, der Benutzer aber trotzdem den eingegebenen Wert abfragen möchte, z.B. über einen Skript.

Beispiel

IsActive-Eigenschaft

Beschreibung

Liefert TRUE zurück, wenn die Kopie des aktuellen Bildes aktiv ist. BOOLEAN Lese-Zugriff.

Beispiel

Die Prozedur "ActiveDocumentConfiguration()" greift auf die Eigenschaften des aktuellen Bildes im Graphics Designer zu. In diesem Beispiel wird eine Kopie des aktuellen Bildes erzeugt und ausgegeben, ob sie aktiv ist.

```
Sub ActiveDocumentConfiguration()
'VBA537
Application.ActiveDocument.Views.Add
'If you comment out the following line
'and recall the procedure, the output of
'the messagebox is different
Application.ActiveDocument.Views(1).Activate
```

```
'
'Output state of copy:
MsgBox Application.ActiveDocument.Views(1).IsActive
End Sub
```

Siehe auch

ActiveDocument-Eigenschaft (Seite 2067)

View-Objekt (Seite 2062)

IsConnectedToProject-Eigenschaft

Beschreibung

Liefert TRUE zurück, wenn die Projektverbindung vorhanden ist. BOOLEAN Lese-Zugriff.

Beispiel

Die Prozedur "ConnectCheck()" prüft, ob eine Projektverbindung besteht und gibt das Ergebnis aus:

```
Sub ConnectCheck()
'VBA538
Dim bCheck As Boolean
Dim strStatus As String
bCheck = Application.IsConnectedToProject
If bCheck = True Then
strStatus = "yes"
Else
strStatus = "no"
End If
MsgBox "Connection to project available: " & strStatus
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

IsDynamicable-Eigenschaft

Beschreibung

TRUE, wenn eine Eigenschaft dynamisiert werden kann. BOOLEAN Lese-Zugriff.

Beispiel

Das Ereignis HMIObjectPropertyChanged tritt immer dann ein, wenn Sie im Graphics Designer eine Objekteigenschaft ändern. In diesem Beispiel werden der Eigenschaftsname und ihr Wert ausgegeben. Zusätzlich wird geprüft, ob die Eigenschaft dynamisiert werden kann:

```
Sub Document_HMIObjectPropertyChanged(ByVal Property As IHMIProperty, CancelForwarding As
Boolean)
'VBA539
Dim objProp As HMIProperty
Dim strStatus As String
Set objProp = Property
'
'Checks whether property is dynamicable
If objProp.IsDynamicable = True Then
strStatus = "yes"
Else
strStatus = "no"
End If
MsgBox "Property: " & objProp.Name & vbCrLf & "Value: " & objProp.value & vbCrLf &
"Dynamicable: " & strStatus
End Sub
```

Mehr zum Thema "Ereignisse" finden Sie unter "VBA-Makros im Graphics Designer ausführen".

Siehe auch

Property-Objekt (Seite 2005)

HMIObject-Objekt (Seite 1955)

HMIObjectPropertyChanged-Ereignis (Seite 1760)

VBA-Makros im Graphics Designer ausführen (Seite 1621)

Italic-Eigenschaft

Beschreibung

TRUE, wenn das Schriftattribut "Kursiv" für den sprachabhängigen Text im Objekt gesetzt ist. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Im folgenden Beispiel werden für einen Button die Schriftattribute für Französisch und Englisch gesetzt:

```
Sub ExampleForLanguageFonts()
'VBA540
```

```
Dim objLangFonts As HMILanguageFonts
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
objButton.Text = "Hello"
Set objLangFonts = objButton.LDFonts
'
'To make fontsettings for french:
With objLangFonts.ItemByLCID(1036)
.Family = "Courier New"
.Bold = True
.Italic = False
.Underlined = True
.Size = 12
End With
'
'To make fontsettings for english:
With objLangFonts.ItemByLCID(1033)
.Family = "Times New Roman"
.Bold = False
.Italic = True
.Underlined = False
.Size = 14
End With
End Sub
```

Siehe auch

Underlined-Eigenschaft (Seite 2402)

Size-Eigenschaft (Seite 2368)

Parent-Eigenschaft (Seite 2317)

LanguageID-Eigenschaft (Seite 2232)

FontFamily-Eigenschaft (Seite 2201)

Bold-Eigenschaft (Seite 2106)

Application-Eigenschaft (Seite 2079)

LanguageFont-Objekt (Seite 1962)

Item-Eigenschaft

Beschreibung

Gibt ein Element aus einer Auflistung zurück. Abhängig vom angegebenen Objekt können Sie entweder die Indexnummer oder den Namen verwenden, um ein bestimmtes Element zurückzugeben.

Beispiel

Dieses Beispiel zeigt beide Fälle der Indizierung. Damit das Beispiel legen Sie ein Gruppenobjekt ("Group1") mit zwei Objekten an. Es gibt die Höhe des zweiten Objekts einer Gruppe aus:

```
Sub GetHeight()
'VBA541
Dim objGroup As HMIGroup
'Next line uses the property "Item" to get a group by name
Set objGroup = ActiveDocument.HMIObjects.Item("Group1")
'Otherwise next line uses index to identify a groupobject
MsgBox "The height of object 2 is: " & objGroup.GroupedHMIObjects.Item(2).Height
End Sub
```

Siehe auch

VariableTriggers-Objekt (Auflistung) (Seite 2061)

VariableStateValues-Objekt (Auflistung) (Seite 2058)

AnalogResultInfos-Objekt (Auflistung) (Seite 1887)

ItemBorderBackColor-Eigenschaft

Beschreibung

Legt die Hintergrundfarbe der Trennlinien in der Auswahlliste des Objektes TextList fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Die Hintergrundfarbe ist nur sichtbar, wenn die Eigenschaft ItemBorderStyle > 0 gesetzt ist.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "TextListConfiguration()" greift auf die Eigenschaften des Objektes TextList zu. In diesem Beispiel wird Hintergrundfarbe der Trennlinien auf "Rot" gesetzt:

```
Sub TextListConfiguration()
'VBA542
Dim objTextList As HMITextList
Set objTextList = ActiveDocument.HMIObjects.AddHMIObject("myTextList", "HMITextList")
With objTextList
```

```
.ItemBorderStyle = 1
.ItemBorderBackColor = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

ItemBorderStyle-Eigenschaft (Seite 2228)

TextList-Objekt (Seite 2037)

ItemBorderColor-Eigenschaft

Beschreibung

Legt die Farbe der Trennlinien in der Auswahlliste des Objektes TextList fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "TextListConfiguration()" greift auf die Eigenschaften des Objektes TextList zu. In diesem Beispiel wird Farbe der Trennlinien auf "Weiß" gesetzt:

```
Sub TextListConfiguration()
'VBA543
Dim objTextList As HMITextList
Set objTextList = ActiveDocument.HMIObjects.AddHMIObject("myTextList", "HMITextList")
With objTextList
.ItemBorderStyle = 1
.ItemBorderColor = RGB(255, 255, 255)
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

ItemBorderStyle-Eigenschaft

Beschreibung

Legt die Trennlinienart der Auswahlliste des Objektes TextList fest oder gibt sie zurück. Wertebereich von 0 bis 4.

Linienart	zugeordneter Wert
———	0
— —	1
- - - - - -	2
— · —	3
—··—	4

Beispiel

Die Prozedur "TextListConfiguration()" greift auf die Eigenschaften des Objektes TextList zu. In diesem Beispiel wird die Trennlinienart auf "1" gesetzt:

```
Sub TextListConfiguration()
'VBA544
Dim objTextList As HMITextList
Set objTextList = ActiveDocument.HMIObjects.AddHMIObject("myTextList", "HMITextList")
With objTextList
.ItemBorderStyle = 1
.ItemBorderBackColor = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

ItemBorderWidth-Eigenschaft

Beschreibung

Legt die Trennlinienstärke der Auswahlliste des Objektes TextList in Pixel fest oder gibt sie zurück.

Beispiel

Die Prozedur "TextListConfiguration()" greift auf die Eigenschaften des Objektes TextList zu. In diesem Beispiel wird die Trennlinienbreite auf "4" gesetzt:

```
'Sub E_628_TextListConfiguration()
```

```
Sub E_629_TextListConfiguration()
'VBA545
Dim objTextList As HMITextList
Set objTextList =
ActiveDocument.HMIObjects.AddHMIObject("myTextList", "HMITextList")
With objTextList
.ItemBorderWidth = 4
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

Key-Eigenschaft

Beschreibung

Gibt die Bezeichnung zurück, die den Eintrag (Menüpunkt oder Symbol) des benutzerdefinierten Menüs oder der benutzerdefinierten Symbolleiste identifiziert. Lese-Zugriff.

Verwenden Sie die Key-Eigenschaft, um zu ermitteln, welcher Eintrag angeklickt wurde. Dazu stehen Ihnen z.B. die Ereignisse "MenuItemClicked" und "ToolBarItemClicked" zur Verfügung.

Beispiel

Die Prozedur "CreateMenuItem()" legt das Menü "Objekte löschen" an und fügt zwei Menüeinträge ("Rechtecke löschen" und "Kreise löschen") hinzu:

```
Sub CreateMenuItem()
'VBA546
Dim objMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
'
'Add new menu "Delete objects" to menubar:
Set objMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DeleteObjects", "Delete objects")
'
'Adds two menuitems to menu "Delete objects"
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(1, "DeleteAllRectangles", "Delete
Rectangles")
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(2, "DeleteAllCircles", "Delete Circles")
End Sub
```

In Verbindung mit dem Ereignis "MenuItemClicked" können Sie die Menüeinträge z.B. mit Prozeduraufrufen verbinden. In diesem Beispiel werden die Namen der Menüeinträge ausgegeben:

```
Sub Document_MenuItemClicked(ByVal MenuItem As IHMIMenuItem)
'VBA547
Dim strClicked As String
Dim objMenuItem As HMIMenuItem
Set objMenuItem = MenuItem
'
'"strClicked can get two values:
'(1) "DeleteAllRectangles" and
'(2) "DeleteAllCircles"
strClicked = objMenuItem.Key
'
'To analyse "strClicked" with "Select Case"
Select Case strClicked
Case "DeleteAllRectangles"
'
'Instead of "MsgBox" a procedurecall (e.g. "Call <Prozedurname>") can stay here
MsgBox "'Delete rectangle' was clicked"
Case "DeleteAllCircles"
MsgBox "'Delete Circles' was clicked"
End Select
End Sub
```

Siehe auch

ToolbarItem-Objekt (Seite 2043)

MenuItem-Objekt (Seite 1979)

InsertToolbarItem-Methode (Seite 1842)

InsertMenuItem-Methode (Seite 1838)

ToolbarItemClicked-Ereignis (Seite 1772)

MenuItemClicked-Ereignis (Seite 1766)

Eigene Menüs und Symbolleisten anlegen (Seite 1629)

L

La

Label-Eigenschaft

Beschreibung

Gibt die Beschriftung des benutzerdefinierten Menüs oder des Menüeintrags in der aktuell eingestellten Sprache zurück. Lese-Zugriff.

Beispiel

Die Prozedur "CreateMenuItem()" legt das Menü "Objekte löschen" an und fügt zwei Menüeinträge ("Rechtecke löschen" und "Kreise löschen") hinzu. In diesem Beispiel werden danach die Beschriftungen ausgegeben:

```
Sub CreateMenuItem()
'VBA548
Dim objMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Dim iIndex As Integer
iIndex = 1
'
'Add new menu "Delete objects" to menubar
Set objMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DeleteObjects", "Delete objects")
'
'Adds two menuitems to menu "Delete objects"
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(1, "DeleteAllRectangles", "Delete
rectangles")
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(2, "DeleteAllCircles", "Delete circles")
MsgBox ActiveDocument.CustomMenus(1).Label
For iIndex = 1 To objMenu.MenuItems.Count
MsgBox objMenu.MenuItems(iIndex).Label
Next iIndex
End Sub
```

Siehe auch

CustomMenus-Eigenschaft (Seite 2158)

MenuItems-Objekt (Auflistung) (Seite 1982)

MenuItem-Objekt (Seite 1979)

Menu-Objekt (Seite 1976)

LanguageID-Eigenschaft

Beschreibung

Gibt die Sprachenkennung der Projektiersprache als Dezimalwert zurück. LONG Lese-Zugriff

Beispiel

Die Prozedur "DataLanguages()" gibt die Projektiersprachen zusammen mit ihrer Sprachenkennung aus:

```
Sub DataLanguages()
'VBA549
Dim colDataLang As HMIDataLanguages
Dim objDataLang As HMIDataLanguage
Dim nLangID As Long
Dim strLangName As String
Dim iAnswer As Integer
Set colDataLang = Application.AvailableDataLanguages
For Each objDataLang In colDataLang
nLangID = objDataLang.LanguageID
strLangName = objDataLang.LanguageName
iAnswer = MsgBox(nLangID & " " & strLangName, vbOKCancel)
If vbCancel = iAnswer Then Exit For
Next objDataLang
End Sub
```

Siehe auch

DataLanguages-Objekt (Auflistung) (Seite 1915)

DataLanguage-Objekt (Seite 1914)

LanguageName-Eigenschaft

Beschreibung

Gibt die Projektiersprache zurück. STRING Lese-Zugriff

Beispiel

Die Prozedur "DataLanguages()" gibt die Projektiersprachen zusammen mit ihrer Sprachenkennung aus:

```
Sub DataLanguages()
'VBA550
Dim colDataLang As HMIDataLanguages
```

```
Dim objDataLang As HMIDataLanguage
Dim nLangID As Long
Dim strLangName As String
Dim iAnswer As Integer
Set colDataLang = Application.AvailableDataLanguages
For Each objDataLang In colDataLang
nLangID = objDataLang.LanguageID
strLangName = objDataLang.LanguageName
iAnswer = MsgBox(nLangID & " " & strLangName, vbOKCancel)
If vbCancel = iAnswer Then Exit For
Next objDataLang
End Sub
```

Siehe auch

DataLanguages-Objekt (Auflistung) (Seite 1915)

DataLanguage-Objekt (Seite 1914)

LanguageSwitch-Eigenschaft

Beschreibung

Legt fest, wo die sprachabhängigen Zuordnungstexte abgelegt werden, oder gibt den Wert zurück. BOOLEAN Schreib-Lese-Zugriff.

TRUE, wenn die Texte in der Textbibliothek verwaltet werden. Übersetzungen in andere Sprachen erfolgen in der Textbibliothek.

FALSE, wenn die Texte direkt im Objekt verwaltet werden. Übersetzungen in andere Sprachen können mittels Text Distributor vorgenommen werden.

Beispiel

Die Prozedur "TextListConfiguration()" greift auf die Eigenschaften des Objektes TextList zu. In diesem Beispiel sollen die Texte in der Textbibliothek verwaltet werden:

```
Sub TextListConfiguration()
'VBA551
Dim objTextList As HMITextList
Set objTextList = ActiveDocument.HMIObjects.AddHMIObject("myTextList", "HMITextList")
With objTextList
.LanguageSwitch = True
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

LastChange-Eigenschaft

Beschreibung

Gibt das Datum der letzten Änderung am aktuellen Bild zurück. LESE-Zugriff.

Beispiel

Die Prozedur "ActiveDocumentConfiguration()" greift auf die Eigenschaften des aktuellen Bildes im Graphics Designer zu. In diesem Beispiel wird die letzte Änderung des aktuellen Bildes ausgegeben:

```
Sub ActiveDocumentConfiguration()
'VBA552
Dim varLastDocChange As Variant
varLastDocChange = Application.ActiveDocument.LastChange
MsgBox "Last changing: " & varLastDocChange
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

Layer-Eigenschaft

Beschreibung

Legt die Ebene (Layer) im Bild fest, in der sich ein Objekt befindet oder gibt sie zurück. Insgesamt stehen 32 Ebenen zur Verfügung, wobei Ebene "0" die unterste und Ebene "31" die oberste Ebene ist.

Innerhalb einer Ebene liegen die zuerst projektierten Objekte im Hintergrund.

Hinweis

Die Zählweise beginnt in VBA bei "1". Ein "objRectangle.Layer = 1" befindet sich demnach in der untersten Ebene.

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel wird das Objekt Rechteck in Ebene "4" eingefügt:

```
Sub RectangleConfiguration()
'VBA553
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.Layer = 4
End With
End Sub
```

Siehe auch

HMIObject-Objekt (Seite 1955)

Ebenen mit VBA bearbeiten (Seite 1659)

Layer00Checked-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt 3D-BarGraph die Grenze 0 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Grenzwert und Darstellung werden mit den Eigenschaften Layer00Value und Layer00Color festgelegt.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel soll die Grenze 0 überwacht werden:

```
Sub HMI3DBarGraphConfiguration()
'VBA554
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer00Checked = True
End With
End Sub
```

Siehe auch

Layer00Value-Eigenschaft (Seite 2237)

Layer00Color-Eigenschaft (Seite 2236)

3DBarGraph-Objekt (Seite 1879)

Layer00Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 0 des Objektes 3DBarGraph fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer00Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 0 die Farbe "Magenta" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA555
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer00Checked = True
.Layer00Color = RGB(255, 0, 255)
End With
End Sub
```

Siehe auch

Layer00Checked-Eigenschaft (Seite 2235)

3DBarGraph-Objekt (Seite 1879)

Layer00Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 0" beim Objekt 3DBarGraph fest oder gibt ihn zurück.

Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer00Checked den Wert TRUE hat.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 0 der Wert "0" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA556
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer00Checked = True
.Layer00Value = 0
End With
End Sub
```

Siehe auch

Layer00Checked-Eigenschaft (Seite 2235)

3DBarGraph-Objekt (Seite 1879)

Layer01Checked-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt 3D-BarGraph die Grenze 1 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Grenzwert und Darstellung werden mit den Eigenschaften Layer01Value und Layer01Color festgelegt.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel soll die Grenze 1 überwacht werden:

```
Sub HMI3DBarGraphConfiguration()
'VBA557
Dim obj3DBar As HMI3DBarGraph
```

```
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer01Checked = True
End With
End Sub
```

Siehe auch

Layer01Value-Eigenschaft (Seite 2239)

Layer01Color-Eigenschaft (Seite 2238)

3DBarGraph-Objekt (Seite 1879)

Layer01Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 1 des Objektes 3DBarGraph fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer01Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 1 die Farbe "Magenta" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA558
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer01Checked = True
.Layer01Color = RGB(255, 0, 255)
End With
End Sub
```

Siehe auch

Layer01Checked-Eigenschaft (Seite 2237)

3DBarGraph-Objekt (Seite 1879)

Layer01Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 1" beim Objekt 3DBarGraph fest oder gibt ihn zurück.

Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer01Checked den Wert TRUE hat.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 1 der Wert "10" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA559
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer01Checked = True
.Layer01Value = 10
End With
End Sub
```

Siehe auch

Layer01Checked-Eigenschaft (Seite 2237)

3DBarGraph-Objekt (Seite 1879)

Layer02Checked-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt 3D-BarGraph die Grenze 2 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Grenzwert und Darstellung werden mit den Eigenschaften Layer02Value und Layer02Color festgelegt.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel soll die Grenze 2 überwacht werden:

```
Sub HMI3DBarGraphConfiguration()
'VBA560
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer02Checked = True
End With
End Sub
```

Siehe auch

Layer02Value-Eigenschaft (Seite 2241)

Layer02Color-Eigenschaft (Seite 2240)

3DBarGraph-Objekt (Seite 1879)

Layer02Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 2 des Objektes 3DBarGraph fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer02Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 2 die Farbe "Magenta" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA561
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
```

```
.Layer02Checked = True
.Layer02Color = RGB(255, 0, 255)
End With
End Sub
```

Siehe auch

Layer02Checked-Eigenschaft (Seite 2239)

3DBarGraph-Objekt (Seite 1879)

Layer02Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 2" beim Objekt 3DBarGraph fest oder gibt ihn zurück.

Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer02Checked den Wert TRUE hat.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 2 der Wert "20" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA562
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer02Checked = True
.Layer02Value = 0
End With
End Sub
```

Siehe auch

Layer02Checked-Eigenschaft (Seite 2239)

3DBarGraph-Objekt (Seite 1879)

Layer03Checked-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt 3D-BarGraph die Grenze 3 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Grenzwert und Darstellung werden mit den Eigenschaften Layer03Value und Layer03Color festgelegt.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel soll die Grenze 3 überwacht werden:

```
Sub HMI3DBarGraphConfiguration()
'VBA563
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer03Checked = True
End With
End Sub
```

Siehe auch

Layer03Value-Eigenschaft (Seite 2243)

Layer03Color-Eigenschaft (Seite 2242)

3DBarGraph-Objekt (Seite 1879)

Layer03Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 3 des Objektes 3DBarGraph fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer03Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 3 die Farbe "Magenta" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
```

```
'VBA564
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer03Checked = True
.Layer03Color = RGB(255, 0, 255)
End With
End Sub
```

Siehe auch

Layer03Checked-Eigenschaft (Seite 2241)

3DBarGraph-Objekt (Seite 1879)

Layer03Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 3" beim Objekt 3DBarGraph fest oder gibt ihn zurück.

Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer03Checked den Wert TRUE hat.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 3 der Wert "30" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA565
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer03Checked = True
.Layer03Value = 30
End With
End Sub
```

Siehe auch

Layer03Checked-Eigenschaft (Seite 2241)

3DBarGraph-Objekt (Seite 1879)

Layer04Checked-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt 3D-BarGraph die Grenze 4 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Grenzwert und Darstellung werden mit den Eigenschaften Layer04Value und Layer04Color festgelegt.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel soll die Grenze 4 überwacht werden:

```
Sub HMI3DBarGraphConfiguration()
'VBA566
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer04Checked = True
End With
End Sub
```

Siehe auch

Layer04Value-Eigenschaft (Seite 2245)

Layer04Color-Eigenschaft (Seite 2244)

3DBarGraph-Objekt (Seite 1879)

Layer04Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 4 des Objektes 3DBarGraph fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer04Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 4 die Farbe "Magenta" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA567
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer04Checked = True
.Layer04Color = RGB(255, 0, 255)
End With
End Sub
```

Siehe auch

Layer04Checked-Eigenschaft (Seite 2244)

3DBarGraph-Objekt (Seite 1879)

Layer04Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 4" beim Objekt 3DBarGraph fest oder gibt ihn zurück.

Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer00Checked den Wert TRUE hat.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 4 der Wert "40" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA568
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer04Checked = True
.Layer04Value = 40
End With
End Sub
```

Siehe auch

Layer00Checked-Eigenschaft (Seite 2235)

3DBarGraph-Objekt (Seite 1879)

Layer05Checked-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt 3D-BarGraph die Grenze 5 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Grenzwert und Darstellung werden mit den Eigenschaften Layer05Value und Layer05Color festgelegt.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel soll die Grenze 5 überwacht werden:

```
Sub HMI3DBarGraphConfiguration()
'VBA569
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer05Checked = True
End With
End Sub
```

Siehe auch

Layer05Value-Eigenschaft (Seite 2247)

Layer05Color-Eigenschaft (Seite 2246)

3DBarGraph-Objekt (Seite 1879)

Layer05Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 5 des Objektes 3DBarGraph fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer05Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 5 die Farbe "Magenta" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA570
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer05Checked = True
.Layer05Color = RGB(255, 0, 255)
End With
End Sub
```

Siehe auch

Layer05Checked-Eigenschaft (Seite 2246)

3DBarGraph-Objekt (Seite 1879)

Layer05Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 5" beim Objekt 3DBarGraph fest oder gibt ihn zurück.

Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer05Checked den Wert TRUE hat.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 5 der Wert "50" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA571
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer05Checked = True
```

```
.Layer05Value = 50
End With
End Sub
```

Siehe auch

Layer05Checked-Eigenschaft (Seite 2246)

3DBarGraph-Objekt (Seite 1879)

Layer06Checked-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt 3D-BarGraph die Grenze 6 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Grenzwert und Darstellung werden mit den Eigenschaften Layer06Value und Layer06Color festgelegt.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel soll die Grenze 6 überwacht werden:

```
Sub HMI3DBarGraphConfiguration()
'VBA572
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer06Checked = True
End With
End Sub
```

Siehe auch

Layer06Value-Eigenschaft (Seite 2249)

Layer06Color-Eigenschaft (Seite 2248)

3DBarGraph-Objekt (Seite 1879)

Layer06Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 6 des Objektes 3DBarGraph fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer06Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 6 die Farbe "Magenta" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA573
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer06Checked = True
.Layer06Color = RGB(255, 0, 255)
End With
End Sub
```

Siehe auch

Layer06Checked-Eigenschaft (Seite 2248)

3DBarGraph-Objekt (Seite 1879)

Layer06Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 6" beim Objekt 3DBarGraph fest oder gibt ihn zurück.

Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer06Checked den Wert TRUE hat.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 6 der Wert "60" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA574
Dim obj3DBar As HMI3DBarGraph
```

```
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer06Checked = True
.Layer06Value = 60
End With
End Sub
```

Siehe auch

Layer06Checked-Eigenschaft (Seite 2248)

3DBarGraph-Objekt (Seite 1879)

Layer07Checked-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt 3D-BarGraph die Grenze 7 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Grenzwert und Darstellung werden mit den Eigenschaften Layer07Value und Layer07Color festgelegt.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel soll die Grenze 7 überwacht werden:

```
Sub HMI3DBarGraphConfiguration()
'VBA575
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer07Checked = True
End With
End Sub
```

Siehe auch

Layer07Value-Eigenschaft (Seite 2251)

Layer07Color-Eigenschaft (Seite 2251)

3DBarGraph-Objekt (Seite 1879)

Layer07Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 7 des Objektes 3DBarGraph fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer07Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 7 die Farbe "Magenta" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA576
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer07Checked = True
.Layer07Color = RGB(255, 0, 255)
End With
End Sub
```

Siehe auch

Layer07Checked-Eigenschaft (Seite 2250)

3DBarGraph-Objekt (Seite 1879)

Layer07Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 7" beim Objekt 3DBarGraph fest oder gibt ihn zurück.

Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer07Checked den Wert TRUE hat.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 7 der Wert "70" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA577
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer07Checked = True
.Layer07Value = 70
End With
End Sub
```

Siehe auch

Layer07Checked-Eigenschaft (Seite 2250)

3DBarGraph-Objekt (Seite 1879)

Layer08Checked-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt 3D-BarGraph die Grenze 8 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Grenzwert und Darstellung werden mit den Eigenschaften Layer08Value und Layer08Color festgelegt.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel soll die Grenze 8 überwacht werden:

```
Sub HMI3DBarGraphConfiguration()
'VBA578
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer08Checked = True
End With
End Sub
```

Siehe auch

Layer08Value-Eigenschaft (Seite 2254)

Layer08Color-Eigenschaft (Seite 2253)

3DBarGraph-Objekt (Seite 1879)

Layer08Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 8 des Objektes 3DBarGraph fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer08Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 8 die Farbe "Magenta" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA579
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer08Checked = True
.Layer08Color = RGB(255, 0, 255)
End With
End Sub
```

Siehe auch

Layer08Checked-Eigenschaft (Seite 2252)

3DBarGraph-Objekt (Seite 1879)

Layer08Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 8" beim Objekt 3DBarGraph fest oder gibt ihn zurück.

Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer08Checked den Wert TRUE hat.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 8 der Wert "80" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA580
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer08Checked = True
.Layer08Value = 80
End With
End Sub
```

Siehe auch

Layer08Checked-Eigenschaft (Seite 2252)

3DBarGraph-Objekt (Seite 1879)

Layer09Checked-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt 3D-BarGraph die Grenze 9 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Grenzwert und Darstellung werden mit den Eigenschaften Layer09Value und Layer09Color festgelegt.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel soll die Grenze 9 überwacht werden:

```
Sub HMI3DBarGraphConfiguration()
'VBA581
Dim obj3DBar As HMI3DBarGraph
```

```
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer09Checked = True
End With
End Sub
```

Siehe auch

Layer09Value-Eigenschaft (Seite 2256)

Layer09Color-Eigenschaft (Seite 2255)

3DBarGraph-Objekt (Seite 1879)

Layer09Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 9 des Objektes 3DBarGraph fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer09Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 9 die Farbe "Magenta" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA582
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer09Checked = True
.Layer09Color = RGB(255, 0, 255)
End With
End Sub
```

Siehe auch

Layer09Checked-Eigenschaft (Seite 2254)

3DBarGraph-Objekt (Seite 1879)

Layer09Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 9" beim Objekt 3DBarGraph fest oder gibt ihn zurück.

Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer09Checked den Wert TRUE hat.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 9 der Wert "90" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA583
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer09Checked = True
.Layer09Value = 90
End With
End Sub
```

Siehe auch

Layer09Checked-Eigenschaft (Seite 2254)

3DBarGraph-Objekt (Seite 1879)

Layer10Checked-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt 3D-BarGraph die Grenze 10 überwacht werden soll. BOOLEAN Schreib-Lese-Zugriff.

Grenzwert und Darstellung werden mit den Eigenschaften Layer10Value und Layer10Color festgelegt.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel soll die Grenze 10 überwacht werden:

```
Sub HMI3DBarGraphConfiguration()
'VBA584
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer10Checked = True
End With
End Sub
```

Siehe auch

Layer10Value-Eigenschaft (Seite 2258)

Layer10Color-Eigenschaft (Seite 2257)

3DBarGraph-Objekt (Seite 1879)

Layer10Color-Eigenschaft

Beschreibung

Legt die Farbe für die Grenze 10 des Objektes 3DBarGraph fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Wenn die Überwachung des Grenzwerts aktiviert ist (Eigenschaft Layer10Checked), erhält der Balken mit dem Erreichen des Grenzwerts die Farbe dieses Attributs.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 10 die Farbe "Magenta" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA585
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
```

```
.Layer10Checked = True
.Layer10Color = RGB(255, 0, 255)
End With
End Sub
```

Siehe auch

Layer10Checked-Eigenschaft (Seite 2256)

3DBarGraph-Objekt (Seite 1879)

Layer10Value-Eigenschaft

Beschreibung

Legt den Wert für "Grenze 10" beim Objekt 3DBarGraph fest oder gibt ihn zurück.

Die Überwachung ist nur dann wirksam, wenn Eigenschaft Layer10Checked den Wert TRUE hat.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird für die Grenze 10 der Wert "100" festgelegt:

```
Sub HMI3DBarGraphConfiguration()
'VBA586
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.Layer10Checked = True
.Layer10Value = 100
End With
End Sub
```

Siehe auch

Layer10Checked-Eigenschaft (Seite 2256)

3DBarGraph-Objekt (Seite 1879)

LayerDecluttering-Eigenschaft

Beschreibung

TRUE, wenn das Ein- und Ausblenden von Objekten abhängig vom eingestellen Minimal- und Maximalzoom einer Ebene aktiviert ist. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Im folgenden Beispiel werden im aktiven Bild die Einstellungen der untersten Ebene konfiguriert:

```
Sub ConfigureSettingsOfLayer()
'VBA587
Dim objLayer As HMILayer
Set objLayer = ActiveDocument.Layers(1)
With objLayer
'configure "Layer 0"
.MinZoom = 10
.MaxZoom = 100
.Name = "Configured with VBA"
End With
'define fade-in and fade-out of objects:
With ActiveDocument
.LayerDecluttering = True
.ObjectSizeDecluttering = True
.SetDeclutterObjectSize 50, 100
End With
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

Ebenen mit VBA bearbeiten (Seite 1659)

Layers-Eigenschaft

Beschreibung

Gibt eine Auflistung zurück, welche die Eigenschaften der Ebenen des aktuellen Bildes enthält.

Hinweis

Wenn die Eigenschaft "Layers" geändert wird, kann sich die Reihenfolge der HMI-Objekte in der HMIObjects-Auflistung ändern.

Beispiel

Die Prozedur "LayerInfo()" gibt von jeder Ebene des aktuellen Bildes den Namen und die eingestellten Zoom-Konfigurationen aus:

```
Sub LayerInfo()
'VBA588
Dim colLayers As HMILayers
Dim objLayer As HMILayer
Dim iAnswer As Integer
Set colLayers = ActiveDocument.Layers
For Each objLayer In colLayers
With objLayer
iAnswer = MsgBox("Layername: " & .Name & vbCrLf & "max. zoom:  " & .MaxZoom & vbCrLf & "min.
zoom:  " & .MinZoom, vbOKCancel)
End With
If vbCancel = iAnswer Then Exit For
Next objLayer
End Sub
```

Siehe auch

Name-Eigenschaft (Seite 2303)

MinZoom-Eigenschaft (Seite 2301)

MaxZoom-Eigenschaft (Seite 2284)

Layers-Objekt (Auflistung) (Seite 1969)

Layer-Objekt (Auflistung) (Seite 1967)

LD - Lo

LDFonts-Eigenschaft

Beschreibung

Gibt eine Auflistung zurück, welche die Sprachkennungen der projektierten Schriftarten enthält.

Beispiel

Verwenden Sie die LDFonts-Eigenschaft, um die LanguageFonts-Auflistung zurückzugeben. Im folgenden Beispiel werden die Sprachkennungen der projektierten Schriftarten ausgegeben:

```
Sub ShowLanguageFont()
'VBA589
```

```
Dim colLanguageFonts As HMILanguageFonts
Dim objLanguageFont As HMILanguageFont
Dim objButton As HMIButton
Dim iMax As Integer
Dim iAnswer As Integer
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
Set colLanguageFonts = objButton.LDFonts
iMax = colLanguageFonts.Count
For Each objLanguageFont In colLanguageFonts
iAnswer = MsgBox("Projected fonts: " & iMax & vbCrLf & "Language-ID: " &
objLanguageFont.LanguageID, vbOKCancel)
If vbCancel = iAnswer Then Exit For
Next objLanguageFont
End Sub
```

Siehe auch

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

LanguageFonts-Objekt (Auflistung) (Seite 1963)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

LDLabelTexts-Eigenschaft

Beschreibung

Gibt eine Auflistung zurück, welche die (fremdsprachigen) Beschriftungen des benutzerdefinierten Menüs oder des Menüeintrags enthält.

Beispiel

Die Prozedur "CreateMenuItem()" legt das Menü "Objekte löschen" an und fügt zwei Menüeinträge ("Rechtecke löschen" und "Kreise löschen") hinzu. In diesem Beispiel werden fremdsprachige Menübeschriftungen angelegt:

```
Sub CreateMenuItem()
'VBA590
Dim objMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Dim objLangText As HMILanguageText
'
'Add new menu "Delete objects" to menubar:
Set objMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DeleteObjects", "Delete objects")
'
'Add two menuitems to the new menu
```

```
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(1, "DeleteAllRectangles", "Delete
rectangles")
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(2, "DeleteAllCircles", "Delete circles")
'
'Define foreign-language labels for menu "Delete objects":
Set objLangText = objMenu.LDLabelTexts.Add(1033, "English_Delete objects")
Set objLangText = objMenu.LDLabelTexts.Add(1032, "Greek_Delete objects")
Set objLangText = objMenu.LDLabelTexts.Add(1034, "Spanish_Delete objects")
Set objLangText = objMenu.LDLabelTexts.Add(1036, "French_Delete objects")
End Sub
```

Die Prozedur "LDLabelInfo()" gibt die projektierten Beschriftungen für das Menü "Objekte löschen" aus:

```
Sub LDLabelInfo()
'VBA591
Dim colLangTexts As HMILanguageTexts
Dim objLangText As HMILanguageText
Dim iAnswer As Integer
'
'Save all labels of menu into collection "colLangTexts":
Set colLangTexts = ActiveDocument.CustomMenus("DeleteObjects").LDLabelTexts
For Each objLangText In colLangTexts
iAnswer = MsgBox(objLangText.DisplayName, vbOKCancel)
If vbCancel = iAnswer Then Exit For
Next objLangText
End Sub
```

Siehe auch

MenuItem-Objekt (Seite 1979)

Menu-Objekt (Seite 1976)

LDNames-Eigenschaft

Beschreibung

Gibt eine Auflistung zurück, welche die fremdsprachigen Bezeichnungen eines Ordners der Bausteinbibliothek oder einer Ebene enthält.

Beispiel

Verwenden Sie die LDNames-Eigenschaft, um die die LanguageTexts-Auflistung zurückzugeben. Im folgenden Beispiel werden alle fremdsprachigen Ebenenbezeichnungen ausgegeben:

Erläuterung: Was zeigt das Beispiel

```
Sub LDLabelInfo()
'VBA592
Dim colLayerLngTexts As HMILanguageTexts
Dim objLayerLngText As HMILanguageText
Dim iIndex As Integer
Dim iAnswer As Integer
Dim strResult As String
iIndex = 1
For iIndex = 1 To ActiveDocument.Layers.Count
'
'Save all labels of layers into collection of "colLayerLngTexts":
Set colLayerLngTexts = ActiveDocument.Layers(iIndex).LDNames
For Each objLayerLngText In colLayerLngTexts
strResult = strResult & vbCrLf & objLayerLngText.LanguageID & " - " &
objLayerLngText.DisplayName
Next objLayerLngText
iAnswer = MsgBox(strResult, vbOKCancel)
strResult = ""
If vbCancel = iAnswer Then Exit For
Next iIndex
End Sub
```

Siehe auch

Layer-Objekt (Seite 1967)

LanguageTexts-Objekt (Auflistung) (Seite 1966)

FolderItem-Objekt (Seite 1939)

LDStatusTexts-Eigenschaft

Beschreibung

Gibt eine Auflistung zurück, welche die (fremdsprachigen) Statuszeilentexte eines benutzerdefinierten Symbol-Icons oder Menüeintrags enthält.

Beispiel

Die Prozedur "CreateMenuItem()" legt das Menü "Delete Objects" an und fügt zwei Menüeinträge ("Delete Rectangles" und "Delete Circles") hinzu. In diesem Beispiel werden fremdsprachige Statuszeilentexte angelegt:

```
Sub CreateMenuItem()
'VBA593
Dim objMenu As HMIMenu
Dim objMenuItem1 As HMIMenuItem
Dim objMenuItem2 As HMIMenuItem
Dim objLangStateText As HMILanguageText
'
```

```
'Add new menu "Delete objects" to menubar:
Set objMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DeleteObjects", "Delete objects")
'
'Add two menuitems to the new menu
Set objMenuItem1 = objMenu.MenuItems.InsertMenuItem(1, "DeleteAllRectangles", "Delete
rectangles")
Set objMenuItem2 = objMenu.MenuItems.InsertMenuItem(2, "DeleteAllCircles", "Delete
circles")
'
'Define foreign-language labels for menuitem "Delete rectangles":
Set objLangStateText = objMenuItem1.LDStatusTexts.Add(1033, "English_Delete rectangles")
Set objLangStateText = objMenuItem1.LDStatusTexts.Add(1032, "Greek_Delete rectangles")
Set objLangStateText = objMenuItem1.LDStatusTexts.Add(1034, "Spanish_Delete rectangles")
Set objLangStateText = objMenuItem1.LDStatusTexts.Add(1036, "French_Delete rectangles")
End Sub
```

Die Prozedur "LDStatusTextInfo()" gibt die projektierten Statuszeilentexte für das Menü "Delete Objects" aus:

```
Sub LDStatusTextInfo()
'VBA594
Dim colMenuItems As HMIMenuItems
Dim objMenuItem As HMIMenuItem
Dim colStatusLngTexts As HMILanguageTexts
Dim objStatusLngText As HMILanguageText
Dim strResult As String
Dim iAnswer As Integer
Set colMenuItems = ActiveDocument.CustomMenus("DeleteObjects").MenuItems
For Each objMenuItem In colMenuItems
strResult = "Statustexts of menuitem """ & objMenuItem.Label & """"
Set colStatusLngTexts = objMenuItem.LDStatusTexts
For Each objStatusLngText In colStatusLngTexts
strResult = strResult & vbCrLf & objStatusLngText.DisplayName
Next objStatusLngText
iAnswer = MsgBox(strResult, vbOKCancel)
If vbCancel = iAnswer Then Exit For
Next objMenuItem
End Sub
```

Siehe auch

ToolbarItem-Objekt (Seite 2043)

MenuItem-Objekt (Seite 1979)

Menu-Objekt (Seite 1976)

LDTexts-Eigenschaft

Beschreibung

Gibt eine Auflistung zurück, welche die (fremdsprachigen) Beschriftungen eines Objektes enthält.

Beispiel

Die Prozedur "LDTextInfo()" gibt die projektierten Beschriftungen für das Objekt Button aus. Damit dieses Beispiel funktioniert, legen Sie im Graphics Designer das Objekt "meinButton" an und projektieren Sie mehrere fremdsprachige Beschriftungen:

```
Sub LDTextInfo()
'VBA595
Dim colLDLngTexts As HMILanguageTexts
Dim objLDLngText As HMILanguageText
Dim objButton As HMIButton
Dim iAnswer As Integer
Set objButton = ActiveDocument.HMIObjects("myButton")
Set colLDLngTexts = objButton.LDTexts
For Each objLDLngText In colLDLngTexts
iAnswer = MsgBox(objLDLngText.DisplayName, vbOKCancel)
If vbCancel = iAnswer Then Exit For
Next objLDLngText
End Sub
```

Siehe auch

Button-Objekt (Seite 1898)

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

CheckBox-Objekt (Seite 1901)

LDTooltipTexts-Eigenschaft

Beschreibung

Gibt eine Auflistung zurück, welche die (fremdsprachigen) Tooltiptexte des benutzerdefinierten Symbols oder des Objektes enthält.

Beispiel

Die Prozedur "CreateToolbar()" erzeugt eine benutzerdefinierte Symbolleiste mit zwei Symbole. Dem ersten Symbol werden zwei fremdsprachige Tooltiptexte zugewiesen:

```
Sub CreateToolbar()
'VBA596
Dim objToolbar As HMIToolbar
Dim objToolbarItem As HMIToolbarItem
Dim objLangText As HMILanguageText
Dim strFileWithPath
'
'Create toolbar with two toolbar-items:
Set objToolbar = ActiveDocument.CustomToolbars.Add("Tool1_1")
Set objToolbarItem = objToolbar.ToolbarItems.InsertToolbarItem(1, "ti1_1",
"myFirstToolbaritem")
Set objToolbarItem = objToolbar.ToolbarItems.InsertToolbarItem(2, "ti1_2",
"mySecondToolbaritem")
'
'In order that the example runs correct copy a *.ICO-Graphic
'into the "GraCS"-Folder of the actual project.
'Replace the filename "EZSTART.ICO" in the next commandline
'with the name of the ICO-Graphic you copied
strFileWithPath = Application.ApplicationDataPath & "EZSTART.ICO"
'
'
'To assign the symbol-icon to the first toolbaritem
objToolbar.ToolbarItems(1).Icon = strFileWithPath
'
'Define foreign-language tooltiptexts
Set objLangText = objToolbar.ToolbarItems(1).LDTooltipTexts.Add(1036, "French_Tooltiptext")
Set objLangText = objToolbar.ToolbarItems(1).LDTooltipTexts.Add(1034,
"Spanish_Tooltiptext")
End Sub
```

Die Prozedur "LDTooltipInfo()" gibt alle projektieren Tooltiptexte des ersten Symbols der ersten benutzerdefinierten Symbolleiste aus:

```
Sub LDTooltipInfo()
'VBA597
Dim colLangTexts As HMILanguageTexts
Dim objLangText As HMILanguageText
Dim iAnswer As Integer
Set colLangTexts = ActiveDocument.CustomToolbars(1).ToolbarItems(1).LDTooltipTexts
For Each objLangText In colLangTexts
iAnswer = MsgBox(objLangText.DisplayName, vbOKCancel)
If vbCancel = iAnswer Then Exit For
Next objLangText
End Sub
```

Siehe auch

ToolbarItem-Objekt (Seite 2043)

HMIObject-Objekt (Seite 1955)

Left-Eigenschaft

Beschreibung

Legt die x-Koordinate des Objektes (gemessen vom linken, oberen Bildrand) in Pixel fest oder gibt sie zurück. Die x-Koordinate bezieht sich auf die Ecke links oben des objektumfassenden Rechteckes.

View-Objekt

Legt die x-Koordinate des Fensters (gemessen vom linken, oberen Rand der Arbeitsfläche des Graphics Designers) in Pixel fest oder gibt sie zurück.

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Rechteckes zu. In diesem Beispiel wird das Rechteck um 40 Pixel nach rechts verschoben:

```
Sub RectangleConfiguration()
'VBA598
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.Left = 40
End With
End Sub
```

Siehe auch

View-Objekt (Seite 2062)

HMIObject-Objekt (Seite 1955)

LeftComma-Eigenschaft

Beschreibung

Legt die Anzahl der Vorkommastellen (0 bis 20) des Objektes BarGraph fest oder gibt sie zurück.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird Anzahl der Vorkommastellen auf "4" gesetzt:

```
Sub BarGraphConfiguration()
'VBA599
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.LeftComma = 4
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

LightEffect-Eigenschaft

Beschreibung

TRUE, wenn der Lichteffekt des Objektes 3DBarGraph eingeschaltet ist. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird der Lichteffekt eingeschaltet:

```
Sub HMI3DBarGraphConfiguration()
'VBA600
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
.LightEffect = True
End With
End Sub
```

Siehe auch

3DBarGraph-Objekt (Seite 1879)

LimitHigh4-Eigenschaft

Beschreibung

Legt beim Objekt BarGraph den oberen Grenzwert für "Reserve 4" fest oder gibt ihn zurück.

Die Eigenschaft CheckLimitHigh4 muss auf TRUE gesetzt sein, damit der Grenzwert "Reserve 4" überwacht werden kann.

Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft TypeLimitHigh4 fest.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "70" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA601
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeLimitHigh4 = False
'Activate monitoring
.CheckLimitHigh4 = True
'Set barcolor to "red"
.ColorLimitHigh4 = RGB(255, 0, 0)
'Set upper limit to "70"
.LimitHigh4 = 70
End With
End Sub
```

Siehe auch

TypeLimitHigh4-Eigenschaft (Seite 2394)

CheckLimitHigh4-Eigenschaft (Seite 2129)

BarGraph-Objekt (Seite 1893)

LimitHigh5-Eigenschaft

Beschreibung

Legt beim Objekt BarGraph den oberen Grenzwert für "Reserve 5" fest oder gibt ihn zurück.

Die Eigenschaft CheckLimitHigh5 muss auf TRUE gesetzt sein, damit der Grenzwert "Reserve 5" überwacht werden kann.

Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft TypeLimitHigh5 fest.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "80" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA602
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeLimitHigh5 = False
'Activate monitoring
.CheckLimitHigh5 = True
'Set barcolor to "black"
.ColorLimitHigh5 = RGB(0, 0, 0)
'Set upper limit to "80"
.LimitHigh4 = 80
End With
End Sub
```

Siehe auch

TypeLimitHigh5-Eigenschaft (Seite 2395)

CheckLimitHigh5-Eigenschaft (Seite 2130)

BarGraph-Objekt (Seite 1893)

LimitLow4-Eigenschaft

Beschreibung

Legt beim Objekt BarGraph den unteren Grenzwert für "Reserve 4" fest oder gibt ihn zurück.

Die Eigenschaft CheckLimitLow4 muss auf TRUE gesetzt sein, damit der Grenzwert "Reserve 4" überwacht werden kann.

Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft TypeLimitLow4 fest.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "5" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA603
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeLimitLow4 = False
'Activate monitoring
.CheckLimitLow4 = True
'Set barcolor to "green"
.ColorLimitLow4 = RGB(0, 255, 0)
'Set lower limit to "5"
.LimitLow4 = 5
End With
End Sub
```

Siehe auch

CheckLimitLow4-Eigenschaft (Seite 2131)

TypeLimitLow4-Eigenschaft (Seite 2396)

BarGraph-Objekt (Seite 1893)

LimitLow5-Eigenschaft

Beschreibung

Legt beim Objekt BarGraph den unteren Grenzwert für "Reserve 5" fest oder gibt ihn zurück.

Die Eigenschaft CheckLimitLow5 muss auf TRUE gesetzt sein, damit der Grenzwert "Reserve 5" überwacht werden kann.

Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft TypeLimitLow5 fest.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "0" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA604
```

```
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis to absolute
.TypeLimitLow5 = False
'Activate monitoring
.CheckLimitLow5 = True
'Set barcolor to "white"
.ColorLimitLow5 = RGB(255, 255, 255)
'Set lower limit to "0"
.LimitLow5 = 0
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

TypeLimitLow5-Eigenschaft (Seite 2397)

CheckLimitLow5-Eigenschaft (Seite 2131)

LimitMax-Eigenschaft

Beschreibung

Legt beim Objekt IOField den oberen Grenzwert als Absolutwert abhängig vom Datenformat fest oder gibt ihn zurück.

Überschreitet der anzuzeigende Wert den oberen Grenzwert, wird er durch eine Folge von *** als nicht darstellbar gekennzeichnet.

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel wird der obere Grenzwert für einen Dezimalwert auf "100" gesetzt:

```
Sub IOFieldConfiguration()
'VBA605
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.DataFormat = 1
.LimitMax = 100
End With
End Sub
```

Siehe auch

DataFormat-Eigenschaft (Seite 2161)

IOField-Objekt (Seite 1959)

LimitMin-Eigenschaft

Beschreibung

Legt beim Objekt IOField den unteren Grenzwert als Absolutwert abhängig vom Datenformat fest oder gibt ihn zurück.

Überschreitet der anzuzeigende Wert den oberen Grenzwert, wird er durch eine Folge von *** als nicht darstellbar gekennzeichnet.

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel wird der obere Grenzwert für einen Dezimalwert auf "0" gesetzt:

```
Sub IOFieldConfiguration()
'VBA606
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.DataFormat = 1
.LimitMin = 0
End With
End Sub
```

Siehe auch

DataFormat-Eigenschaft (Seite 2161)

IOField-Objekt (Seite 1959)

LineJoinStyle-Eigenschaft

Beschreibung

Legt fest, auf welche Weise die Ecken bei einem Rohrpolygon dargestellt werden.

Eckig	Die Rohre sind an den Eckpunkten ohne Abrundungen miteinander verbunden.
Rund	Die Rohre sind an den Eckpunkten außen abgerundet.

Beispiel

ListType-Eigenschaft

Beschreibung

Legt beim Objekt TextList die Listenart fest oder gibt sie zurück. Wertebereich von 0 bis 2.

Listenart	zugeordneter Wert
Dezimal	0
Binär	1
Bit	2

Beispiel

Die Prozedur "TextListConfiguration()" greift auf die Eigenschaften des Objektes TextList zu. In diesem Beispiel wird die Listenart auf "Dezimal" gesetzt:

```
Sub TextListConfiguration()
'VBA607
Dim objTextList As HMITextList
Set objTextList = ActiveDocument.HMIObjects.AddHMIObject("myTextList", "HMITextList")
With objTextList
.ListType = 0
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

LockBackColor-Eigenschaft

Beschreibung

Legt beim Objekt GroupDisplay die Hintergrundfarbe der Schaltfläche für eine gesperrte Mess-Stelle fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Die Eigenschaft LockStatus muss den Wert TRUE haben, damit die Hintergrundfarbe angezeigt werden kann.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel wird die Hintergrundfarbe für eine gesperrte Mess-Stelle auf "rot" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA608
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.LockStatus = True
.LockBackColor = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

LockStatus-Eigenschaft (Seite 2277)

GroupDisplay-Objekt (Seite 1947)

LockedByCreatorID-Eigenschaft

Beschreibung

TRUE, wenn ein Bild durch den SIMATIC Manager erstellt und/oder referenziert wurde. BOOLEAN Lese-Zugriff.

Wenn ein Bild im SIMATIC Manager erstellt wurde, können Sie es später in WinCC bearbeiteten und anschliessend speichern. Sie können das Bild in WinCC allerdings nicht löschen. Dazu verwaltet der SIMATIC Manager für jedes Bild eine Kennung, die sogenannte CreatorID, die in WinCC nicht geändert werden kann.

Sie können das Bild in WinCC bearbeitet, ein Überschreiben des Bildes durch ein WinCC-Bild (LockedByCreatorID = FALSE) wird aber verhindert. Zur Überprüfung sollte bei der SaveAs-Methode eine existierende Datei vor dem Schreiben auf den Wert der LockedByCreatorID-Eigenschaft hin überprüft werden. Wird ein solches Bild mit der SaveAs-Methode in ein neues (noch nicht existierendes) oder ein vorhandenes WinCC-Bild gespeichert, wird die CreatorID nicht mit übertragen.

Beispiel 1

Im folgenden Beispiel wird ein mit dem SIMATIC Manager erstelltes Bild (LockedByCreatorID = TRUE) geöffnet, bearbeitet und gespeichert. Der Wert der LockedByCreatorID-Eigenschaft wird nicht verändert.

```
Sub SaveDocAs_1()
'VBA810
'open an existing file, change it and save it
Dim docOld As Document
Const strFile As String = "Simatic_001.Pdl"
'
Set docOld = Application.Documents.Open(Application.ApplicationDataPath & strFile,
hmiOpenDocumentTypeInvisible)
docOld.Width = docOld.Width + 1
docOld.Save
'
MsgBox "LockedByCreatorID = " & docOld.LOckedByCreatorID, vbOKOnly, "Result"
'
docOld.Close
Set docOld = Nothing
'
End Sub
```

Beispiel 2

In diesem Beispiel wird ein neues Bild mit der SaveAs-Methode als neue Datei gespeichert. Um festzustellen ob das Bild gespeichert werden darf, wird die LockedByCreatorID-Eigenschaft überprüft. In der neuen Datei wird die LockedByCreator-Eigenschaft zurückgesetzt.

```
Sub SaveDocAs_2()
'VBA811
'create a new file and overwrite it to an existing file,
'if it is not 'locked by CreatorID'
Dim docNew As Document
Dim docOld As Document
Const strFile As String = "Simatic_001.Pdl"
'
Set docNew = Application.Documents.Add(hmiOpenDocumentTypeInvisible)
Set docOld = Application.Documents.Open(Application.ApplicationDataPath & strFile,
hmiOpenDocumentTypeInvisible)
'
If docOld.LockedByCreatorID = False Then
docOld.Close
docNew.SaveAs(Application.ApplicationDataPath & strFile)
Else
MsgBox "File cannot be stored (LockedByCreatorID). ", vbOKOnly, "Result"
End If
'
docOld.Close
docNew.Close
```

```
Set docOld = Nothing
Set docNew = Nothing
'
End Sub
```

Siehe auch

SaveAs-Methode (Seite 1865)

Document-Objekt (Seite 1920)

LockStatus-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt GroupDisplay eine gesperrte Mess-Stelle angezeigt werden soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel wird die Hintergrundfarbe für eine gesperrte Mess-Stelle auf "rot" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA609
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.LockStatus = True
.LockBackColor = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

LockText-Eigenschaft

Beschreibung

Legt beim Objekt GroupDisplay die Beschriftung der Schaltflächen für eine gesperrte Mess-Stelle fest.

Die Eigenschaft LockStatus muss den Wert TRUE haben, damit die Beschriftung angezeigt werden kann.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel wird die Beschriftung für eine gesperrte Mess-Stelle auf "gesperrt" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA610
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.LockStatus = True
.LockText = "gesperrt"
End With
End Sub
```

Siehe auch

LockStatus-Eigenschaft (Seite 2277)

GroupDisplay-Objekt (Seite 1947)

LockTextColor-Eigenschaft

Beschreibung

Legt beim Objekt GroupDisplay die Farbe der Schaltflächenbeschriftung für eine gesperrte Mess-Stelle fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Die Eigenschaft LockStatus muss den Wert TRUE haben, damit die Hintergrundfarbe angezeigt werden kann.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel wird die Schaltflächenbeschriftung für eine gesperrte Mess-Stelle auf "gelb" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA611
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.LockStatus = True
.LockTextColor = RGB(0, 255, 255)
End With
End Sub
```

Siehe auch

LockStatus-Eigenschaft (Seite 2277)

GroupDisplay-Objekt (Seite 1947)

LongStrokesBold-Eigenschaft

Beschreibung

TRUE, wenn bei der Darstellung der Skala des Objektes BarGraph die langen Abschnitte fett angezeigt werden sollen. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel sollen die langen Teilstriche nicht fett angezeigt werden:

```
Sub BarGraphConfiguration()
'VBA612
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.LongStrokesBold = False
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

LongStrokesOnly-Eigenschaft

Beschreibung

TRUE, wenn bei der Darstellung der Skala des Objektes BarGraph nur die langen Abschnitte angezeigt werden sollen. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel sollen nur die langen Abschnitte angezeigt werden:

```
Sub BarGraphConfiguration()
'VBA613
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.LongStrokesOnly = True
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

LongStrokesSize-Eigenschaft

Beschreibung

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes.

Beispiel

In diesem Beispiel wird die Länge der Achsabschnitte auf "10" gesetzt:

```
Sub BarGraphConfiguration()
'VBA614
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.LongStrokesSize = 10
End With
End Sub
```

Siehe auch

AxisSection-Eigenschaft (Seite 2087)

BarGraph-Objekt (Seite 1893)

LongStrokesTextEach-Eigenschaft

Beschreibung

Legt fest, welche Abschnitte bei der Darstellung der Skala des Objektes BarGraph beschriftet werden sollen (1 = jeder Abschnitt, 2 = jeder zweite Abschnitt, usw.) oder sie zurück.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird jeder dritte Abschnitt beschriftet:

```
Sub BarGraphConfiguration()
'VBA615
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.LongStrokesTextEach = 3
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

M

Macro-Eigenschaft

Beschreibung

Legt für einen benutzerdefinierten Menüeintrag oder Symbol das VBA-Makro fest, das bei Anwahl ausgeführt werden soll.

Beispiel

Im folgenden Beispiel wird ein benutzerdefiniertes Menü mit zwei Menüeinträgen angelegt, die zwei unterschiedliche VBA-Makros aufrufen:

```
Sub CreateDocumentMenusUsingMacroProperty()
'VBA616
Dim objDocMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Set objDocMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DocMenu1", "Doc_Menu_1")
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(1, "dmItem1_1", "My first menuitem")
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(2, "dmItem1_2", "My second menuitem")
'
'To assign a macro to every menuitem:
With ActiveDocument.CustomMenus("DocMenu1")
.MenuItems("dmItem1_1").Macro = "TestMacro1"
.MenuItems("dmItem1_2").Macro = "TestMacro2"
End With
End Sub
```

Die beiden folgenden Prozeduren können Sie über die Menüeinträge des benutzerdefinierten Menüs "DocMenu1" aufrufen:

```
Sub TestMacro1()
MsgBox "TestMacro1 is executed"
End Sub
```

```
Sub TestMacro2()
MsgBox "TestMacro2 is executed"
End Sub
```

Siehe auch

ToolbarItem-Objekt (Seite 2043)

MenuItem-Objekt (Seite 1979)

So weisen Sie Menüs und Symbolleisten VBA-Makros zu (Seite 1644)

Marker-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt BarGraph die Grenzwerte als Skalenwert angezeigt werden sollen. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel werden die Grenzewerte als Skalenwerte angezeigt:

```
Sub BarGraphConfiguration()
'VBA617
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.Marker = True
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

Max-Eigenschaft

Beschreibung

Legt den Absolutwert bei voller Wertanzeige fest oder gibt ihn zurück.

Ist die Skalenanzeige aktiv, so wird dieser Wert angezeigt.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird der Absolutwert auf "10" gesetzt:

```
Sub BarGraphConfiguration()
'VBA618
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.Max = 10
End With
End Sub
```

Siehe auch

Slider-Objekt (Seite 2025)

BarGraph-Objekt (Seite 1893)

3DBarGraph-Objekt (Seite 1879)

MaximizeButton-Eigenschaft

Beschreibung

TRUE, wenn das Objekt ApplicationWindow in Runtime maximiert werden kann. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "ApplicationWindowConfig" greift auf die Eigenschaften des Applikationsfensters zu. In diesem Beispiel erhält das Applikationsfenster in Runtime die Maximieren-Schaltfläche:

```
Sub ApplicationWindowConfig()
'VBA619
Dim objAppWindow As HMIApplicationWindow
Set objAppWindow = ActiveDocument.HMIObjects.AddHMIObject("AppWindow1",
"HMIApplicationWindow")
With objAppWindow
.MaximizeButton = True
End With
End Sub
```

Siehe auch

ApplicationWindow-Objekt (Seite 1891)

MaxZoom-Eigenschaft

Beschreibung

Legt die maximale Zoomstufe für die Ebene fest oder gibt sie zurück.

Beispiel

Die Prozedur "LayerInfo()" gibt von jeder Ebene des aktuellen Bildes den Namen und die eingestellten Zoom-Konfigurationen aus:

```
Sub LayerInfo()
'VBA620
Dim colLayers As HMILayers
Dim objSingleLayer As HMILayer
Dim iAnswer As Integer
Set colLayers = ActiveDocument.Layers
For Each objSingleLayer In colLayers
With objSingleLayer
iAnswer = MsgBox("Layername: " & .Name & vbCrLf & "Min. zoom:  " & .MinZoom & vbCrLf & "Max.
zoom:  " & .MaxZoom, vbOKCancel)
```

```
End With
If vbCancel = iAnswer Then Exit For
Next objSingleLayer
End Sub
```

Siehe auch

Layer-Objekt (Seite 1967)

Ebenen mit VBA bearbeiten (Seite 1659)

MCGUBackColorOff-Eigenschaft

Beschreibung

Legt für das Objekt GroupDisplay für den Zustand "Gegangen Unquittiert" die Farbe des Hintergrunds für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe für den Blinkzustand "Aus" auf "rot" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA621
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCGUBackColorOff = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCGUBackColorOn-Eigenschaft

Beschreibung

Legt für das Objekt GroupDisplay für den Zustand "Gegangen Unquittiert" die Farbe des Hintergrunds für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe für den Blinkzustand "Ein" auf "weiß" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA622
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCGUBackColorOn = RGB(255, 255, 255)
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCGUBackFlash-Eigenschaft

Beschreibung

TRUE, wenn der Hintergrund des Objektes GroupDisplay beim unquittierten Gehen einer Meldung blinken soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe blinken, wenn eine Meldung unquittiert geht:

```
Sub GroupDisplayConfiguration()
```

```
'VBA623
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCGUBackFlash = True
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCGUTextColorOff-Eigenschaft

Beschreibung

Legt für das Objekt GroupDisplay für den Zustand "Gegangen Unquittiert" die Farbe des Textes für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe des Textes für den Blinkzustand "Aus" auf "blau" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA624
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCGUTextColorOff = RGB(0, 0, 255)
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCGUTextColorOn-Eigenschaft

Beschreibung

Legt für das Objekt GroupDisplay für den Zustand "Gegangen Unquittiert" die Hintergrundfarbe des Textes für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe des Textes für den Blinkzustand "Ein" auf "schwarz" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA625
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCGUTextColorOn = RGB(0, 0, 0)
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCGUTextFlash-Eigenschaft

Beschreibung

TRUE, wenn die Schrift des Objektes GroupDisplay beim unquittierten Gehen einer Meldung blinken soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Schrift blinken, wenn eine Meldung unquittiert geht:

```
Sub GroupDisplayConfiguration()
'VBA626
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCGUTextFlash = True
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCKOBackColorOff-Eigenschaft

Beschreibung

Legt für das Objekt GroupDisplay für den Zustand "Gekommen" die Farbe des Hintergrunds für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe für den Blinkzustand "Aus" auf "rot" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA627
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCKOBackColorOff = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCKOBackColorOn-Eigenschaft

Beschreibung

Legt für das Objekt GroupDisplay für den Zustand "Gekommen" die Farbe des Hintergrunds für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe für den Blinkzustand "Ein" auf "weiß" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA628
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCKOBackColorOn = RGB(255, 255, 255)
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCKOBackFlash-Eigenschaft

Beschreibung

TRUE, wenn der Hintergrund des Objektes GroupDisplay beim Kommen einer Meldung blinken soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe blinken, wenn eine Meldung unquittiert geht:

```
Sub GroupDisplayConfiguration()
'VBA629
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCKOBackFlash = True
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCKOTextColorOff-Eigenschaft

Beschreibung

Legt für das Objekt GroupDisplay für den Zustand "Gekommen" die Farbe des Textes für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe des Textes für den Blinkzustand "Aus" auf "blau" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA630
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCKOTextColorOff = RGB(0, 0, 255)
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCKOTextColorOn-Eigenschaft

Beschreibung

Legt für das Objekt GroupDisplay für den Zustand "Gekommen" die Farbe des Hintergrunds des Textes für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe des Textes für den Blinkzustand "Ein" auf "schwarz" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA631
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCKOTextColorOn = RGB(0, 0, 0)
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCKOTextFlash-Eigenschaft

Beschreibung

TRUE, wenn die Schrift des Objektes GroupDisplay beim Kommen einer Meldung blinken soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Schrift blinken, wenn eine Meldung unquittiert geht:

```
Sub GroupDisplayConfiguration()
'VBA632
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCKOTextFlash = True
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCKQBackColorOff-Eigenschaft

Beschreibung

Legt für das Objekt GroupDisplay für den Zustand "Gegangen Quittiert" die Farbe des Hintergrunds für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe für den Blinkzustand "Aus" auf "rot" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA633
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCKQBackColorOff = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCKQBackColorOn-Eigenschaft

Beschreibung

Legt für das Objekt GroupDisplay für den Zustand "Gegangen Quittiert" die Farbe des Hintergrunds für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe für den Blinkzustand "Ein" auf "weiß" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA634
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCKQBackColorOn = RGB(255, 255, 255)
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCKQBackFlash-Eigenschaft

Beschreibung

TRUE, wenn der Hintergrund des Objektes GroupDisplay beim quittierten Gehen einer Meldung blinken soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe blinken, wenn eine Meldung unquittiert geht:

```
Sub GroupDisplayConfiguration()
'VBA635
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCKQBackFlash = True
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCKQTextColorOff-Eigenschaft

Beschreibung

Legt für das Objekt GroupDisplay für den Zustand "Gegangen Quittiert" die Farbe des Textes für den Blinkzustand "Aus" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe des Textes für den Blinkzustand "Aus" auf "blau" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA636
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCKQTextColorOff = RGB(0, 0, 255)
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCKQTextColorOn-Eigenschaft

Beschreibung

Legt für das Objekt GroupDisplay für den Zustand "Gegangen Quittiert" die Farbe des Hintergrunds des Textes für den Blinkzustand "Ein" fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe des Textes für den Blinkzustand "Ein" auf "schwarz" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA637
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCKQTextColorOn = RGB(0, 0, 0)
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCKQTextFlash-Eigenschaft

Beschreibung

TRUE, wenn die Schrift des Objektes GroupDisplay beim quittierten Gehen einer Meldung blinken soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Schrift blinken, wenn eine Meldung unquittiert geht:

```
Sub GroupDisplayConfiguration()
'VBA638
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MCKQTextFlash = True
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

MCText-Eigenschaft

Beschreibung

Legt beim Objekt GroupDisplay die Beschriftung für die jeweilige Meldeklasse fest oder gibt sie zurück.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel wird für die Meldeklasse "Alarm High" als Beschriftung "Alarm High" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA639
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MessageClass = 0
.MCText = "Alarm High"
End With
End Sub
```

Siehe auch

MessageClass-Eigenschaft (Seite 2300)

GroupDisplay-Objekt (Seite 1947)

MenuItems-Eigenschaft

Beschreibung

Gibt eine Auflistung zurück, welche alle Menüeinträge des benutzerdefinierten Menüs enthält.

Beispiel

Die Prozedur "CreateMenuItem()" legt das Menü "Delete Objects" an und fügt zwei Menüeinträge ("Delete Rectangles" und "Delete Circles") hinzu. In diesem Beispiel werden danach die Beschriftungen ausgegeben:

```
Sub CreateMenuItem()
'VBA640
Dim objMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Dim iIndex As Integer
iIndex = 1
'
'Add new menu "Delete objects" to the menubar:
Set objMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DeleteObjects", "Delete objects")
'
'Add two menuitems to menu "Delete objects"
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(1, "DeleteAllRectangles", "Delete
rectangles")
Set objMenuItem = objMenu.MenuItems.InsertMenuItem(2, "DeleteAllCircles", "Delete circles")
'
'Output label of menu:
MsgBox ActiveDocument.CustomMenus(1).Label
'
'Output labels of all menuitems:
For iIndex = 1 To objMenu.MenuItems.Count
MsgBox objMenu.MenuItems(iIndex).Label
Next iIndex
End Sub
```

Siehe auch

Menu-Objekt (Seite 1976)

MenuItem-Objekt (Seite 1979)

MenuItemType-Eigenschaft

Beschreibung

Gibt den Typ eines benutzerdefinierten Menüeintrages zurück. Lese-Zugriff.

Zurückgegebener Wert	Typ des Menüeintrag
0	Trennstrich (Separator)
1	Untermenü (SubMenu)
2	Menüeintrag (MenuItem)

Beispiel

Die Prozedur "ShowMenuTypes()" gibt die Typen der Menüeinträge für das erste benutzerdefinierte Menü aus:

```
Sub ShowMenuTypes()
'VBA641
Dim iMaxMenuItems As Integer
Dim iMenuItemType As Integer
Dim strMenuItemType As String
Dim iIndex As Integer
iMaxMenuItems = ActiveDocument.CustomMenus(1).MenuItems.Count
For iIndex = 1 To iMaxMenuItems
iMenuItemType = ActiveDocument.CustomMenus(1).MenuItems(iIndex).MenuItemType
Select Case iMenuItemType
Case 0
 strMenuItemType = "Trennstrich (Separator)"
Case 1
 strMenuItemType = "Untermenü (SubMenu)"
Case 2
 strMenuItemType = "Menüeintrag (MenuItem)"
End Select
MsgBox iIndex & ". Menuitemtype: " & strMenuItemType
Next iIndex
End Sub
```

Siehe auch

MenuItem-Objekt (Seite 1979)

Menu-Objekt (Seite 1976)

MessageClass-Eigenschaft

Beschreibung

Legt für das Objekt GroupDisplay die jeweilige Meldeart (Alarm High, Alarm Low, Warnung High, Warnung Low, ...) fest, für welche die Attributeinstellungen "Anzeigetext", "Gekommen -", Gekommen Quittiert -" und "Gegangen Unquittiert -" projektiert werden.

MessageClass	zugeordneter Wert
AlarmHigh	0
AlarmLow	1
WarningHigh	2
WarningLow	3
Toleranz High	4
Toleranz Low	5
AS-Leittechnik Störung	6
AS-Leittechnik Fehler	7
OS-Leittechnik Störung	8

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel die Hintergrundfarbe für den Blinkzustand "Aus" für die Meldeart "AlarmHigh" auf "rot" gesetzt:

```
Sub GroupDisplayConfiguration()
'VBA642
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.MessageClass = 0
.MCGUBackColorOff = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

Min-Eigenschaft

Beschreibung

Legt den Absolutwert bei kleinster Wertanzeige fest oder gibt ihn zurück.

Ist die Skalenanzeige aktiv, so wird dieser Wert angezeigt.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird der Absolutwert auf "1" gesetzt:

```
Sub BarGraphConfiguration()
'VBA643
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.Min = 1
End With
End Sub
```

Siehe auch

Slider-Objekt (Seite 2025)

BarGraph-Objekt (Seite 1893)

3DBarGraph-Objekt (Seite 1879)

MinZoom-Eigenschaft

Beschreibung

Legt die minimale Zoomstufe für die Ebene fest oder gibt sie zurück.

Beispiel

Die Prozedur "LayerInfo()" gibt von jeder Ebene des aktuellen Bildes den Namen und die eingestellten Zoom-Konfigurationen aus:

```
Sub LayerInfo()
'VBA644
Dim colLayers As HMILayers
Dim objLayer As HMILayer
Dim strMaxZoom As String
Dim strMinZoom As String
Dim strLayerName As String
Dim iAnswer As Integer
Set colLayers = ActiveDocument.Layers
For Each objLayer In colLayers
```

```
With objLayer
strMinZoom = .MinZoom
strMaxZoom = .MaxZoom
strLayerName = .Name
iAnswer = MsgBox("Layername: " & strLayerName & vbCrLf & "Min. zoom:  " & strMinZoom &
vbCrLf & "Max. zoom:  " & strMaxZoom, vbOKCancel)
End With
If vbCancel = iAnswer Then Exit For
Next objLayer
End Sub
```

Siehe auch

Layer-Objekt (Seite 1967)

Ebenen mit VBA bearbeiten (Seite 1659)

Modified-Eigenschaft

Beschreibung

TRUE, wenn der Quellcode eines Skriptes oder ein Bild verändert wurde. BOOLEAN Lese-Zugriff.

Beispiel

Im folgenden Beispiel wird geprüft, ob das aktive Bild geändert wurde:

```
Sub CheckModificationOfActiveDocument()
'VBA645
Dim strCheck As String
Dim bModified As Boolean
bModified = ActiveDocument.Modified
Select Case bModified
Case True
strCheck = "Active document is modified"
Case False
strCheck = "Active document is not modified"
End Select
MsgBox strCheck
End Sub
```

Siehe auch

ScriptInfo-Objekt (Seite 2021)

Document-Objekt (Seite 1920)

Moveable-Eigenschaft

Beschreibung

TRUE, wenn die Objekte ApplicationWindow und PictureWindow in Runtime verschoben werden können. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "ApplicationWindowConfig" greift auf die Eigenschaften des Applikationsfensters zu. In diesem Beispiel soll das Applikationsfenster in Runtime verschoben werden können:

```
Sub ApplicationWindowConfig()
'VBA646
Dim objAppWindow As HMIApplicationWindow
Set objAppWindow = ActiveDocument.HMIObjects.AddHMIObject("AppWindow1",
"HMIApplicationWindow")
With objAppWindow
.Moveable = True
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

ApplicationWindow-Objekt (Seite 1891)

N-O

Name-Eigenschaft

Beschreibung

Gibt den Namen des Objektes zurück. STRING Lese-Zugriff.

Beispiel

Die Prozedur "LayerInfo()" gibt von jeder Ebene des aktuellen Bildes den Namen und die eingestellten Zoom-Konfigurationen aus:

```
Sub LayerInfo()
'VBA647
Dim colLayers As HMILayers
Dim objLayer As HMILayer
Dim strMaxZoom As String
```

```
Dim strMinZoom As String
Dim strLayerName As String
Dim iAnswer As Integer
Set colLayers = ActiveDocument.Layers
For Each objLayer In colLayers
With objLayer
strMinZoom = .MinZoom
strMaxZoom = .MaxZoom
strLayerName = .Name
iAnswer = MsgBox("Layername: " & strLayerName & vbCrLf & "Min. zoom:  " & strMinZoom &
vbCrLf & "Max. zoom:  " & strMaxZoom, vbOKCancel)
End With
If vbCancel = iAnswer Then Exit For
Next objLayer
End Sub
```

Siehe auch

Trigger-Objekt (Seite 2047)

SymbolLibrary-Objekt (Seite 2035)

Property-Objekt (Seite 2005)

HMIObject-Objekt (Seite 1955)

Layer-Objekt (Seite 1967)

FolderItem-Objekt (Seite 1939)

Document-Objekt (Seite 1920)

Application-Objekt (Seite 1888)

Name-Eigenschaft (FolderItem)

Beschreibung

Gibt den internen Namen des angegebenen Objektes vom Typ "FolderItem" zurück. Lese-Zugriff.

Beispiel

In diesem Beispiel wird der interne Name des Objektes "PC" ausgegeben, das sich in der Globalen Bausteinbibliothek befindet:

```
Sub ShowInternalNameOfFolderItem()
'VBA536
Dim objGlobalLib As HMISymbolLibrary
Set objGlobalLib = Application.SymbolLibraries(1)
MsgBox objGlobalLib.FolderItems(2).Folder(2).Folder.Item(1).Name
End Sub
```

Siehe auch

FolderItem-Objekt (Seite 1939)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

NegativeValue-Eigenschaft

Beschreibung

Verwenden Sie die BinaryResultInfo-Eigenschaft, um das BinaryResultInfo-Objekt zurückzugeben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert, ein Variablenname vergeben und den beiden binären Wertebereichen die dazugehörigen Eigenschaftswerte zugeordnet:

```
Sub AddDynamicDialogToCircleRadiusTypeBinary()
'VBA648
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_C", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeBool
.BinaryResultInfo.NegativeValue = 20
.BinaryResultInfo.PositiveValue = 40
End With
End Sub
```

Siehe auch

VBA-Referenz (Seite 1735)

PositiveValue-Eigenschaft (Seite 2335)

BinaryResultInfo-Objekt (Seite 1896)

Number-Eigenschaft

Beschreibung

Gibt die Ebenennummer eines Objektes vom Typ "Layer" zurück. Die Zählweise beginnt bei 1. Die erste Ebene namens "Ebene0" gibt demnach den Wert "0" zurück. LESE-Zugriff.

Beispiel

In diesem Beispiel wird der Ebenenname und die Ebenennummer sowie der Index ausgegeben:

```
Sub ShowLayerWithNumbers()
'VBA803
Dim colLayers As HMILayers
Dim objLayer As HMILayer
Dim iAnswer As Integer
Dim iIndex As Integer
iIndex = 1
Set colLayers = ActiveDocument.Layers
For Each objLayer In colLayers
iAnswer = MsgBox("Layername: " & objLayer & vbCrLf & "Layernumber: " & objLayer.Number &
vbCrLf & "Layersindex: " & iIndex, vbOKCancel)
iIndex = iIndex + 1
If vbCancel = iAnswer Then Exit For
Next objLayer
End Sub
```

Siehe auch

Layer-Objekt (Seite 1967)

NumberLines-Eigenschaft

Beschreibung

TextList

Legt für das Objekt "TextList" fest, wieviele Zeilen die Auswahlliste enthalten soll oder gibt den Wert zurück. Wenn die projektierten Zeilen mit ihrer Anzahl, Schriftgröße und Schriftart nicht in die Abmessungen des Objekts passen, erhält die Auswahlliste eine vertikale Bildlaufleiste.

Kombinationsfeld und Listenfeld

Legt für die Objekte "Kombinationsfeld" und "Listenfeld" die Anzahl der Textzeilen fest oder gibt den Wert zurück. Sie können maximal 32.000 Zeilen einstellen.

Der Wert des Attributs "Anzahl Zeilen" gibt gleichzeitig den oberen Grenzwert für das Attribut "Index" in der Eigenschaftsgruppe "Schrift" an. Die Änderung des Werts kann folgende Auswirkungen haben:

- Erhöhung der Anzahl: Neue Zeilen werden unten angefügt. Die Standardbeschriftung des neuen Feldes ändern Sie mit dem Attribut "Text" in der Eigenschaftsgruppe "Schrift".
- Verringerung der Anzahl: Es werden alle Zeilen entfernt, bei denen der Wert des Attributs "Index" höher ist als die neue Anzahl.

Beispiel

Die Prozedur "TextListConfiguration()" greift auf die Eigenschaften des Objektes "TextList" zu. In diesem Beispiel wird eine Auswahlliste erzeugt und die Anzahl der sichtbaren Zeilen auf drei gesetzt:

```
Sub TextListConfiguration()
'VBA649
Dim objTextList As HMITextList
'
'Neue TextListe ins aktuelle Bild einfügen:
Set objTextList = ActiveDocument.HMIObjects.AddHMIObject("myTextList", "HMITextList")
With objTextList
.NumberLines=3
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

ObjectName-Eigenschaft

Beschreibung

Legt abhängig vom gewählten Quell- und Zielobjekttyp der Direktverbindung entweder den Namen der Konstanten, des Objektes oder der Variablen fest oder gibt ihn zurück.

Die beiden Tabellen zeigen Ihnen, wann Sie die ObjectName-Eigenschaft verwenden müssen. Ein "--" bedeutet, dass die Eigenschaft beim Erzeugen des DirectConnection-Objektes mit einem Leerstring (" ") vorbelegt wird.

Quellobjekttyp (SourceLink-Eigenschaft)

Type-Eigenschaft	AutomationName-Eigenschaft	ObjectName-Eigenschaft
hmiSourceTypeConstant	--	Name der Konstanten (z.B. der Bildname)
hmiSourceTypeProperty	Eigenschaft des Quellobjektes (z.B. "Top")	Name des Quellobjektes (z.B. "Rectangle_A")
hmiSourceTypePropertyOfThisObject	--	--
hmiSourceTypeVariableDirect	--	Variablenname
hmiSourceTypeVariableIndirect	--	Variablenname

Zielobjekttyp (DestinationLink-Eigenschaft)

Type-Eigenschaft	AutomationName-Eigenschaft	ObjectName-Eigenschaft
hmiDestTypeProperty	Eigenschaft des Zielobjektes (z.B. "Left")	Name des Zielobjektes (z.B. "Rectangle_A")
hmiDestTypePropertyOfThisObject	--	--
hmiDestTypePropertyOfActualWindow	Eigenschaft des Zielobjektes (z.B. "Left")	--
hmiDestTypeVariableDirect	--	Variablenname
hmiDestTypeVariableIndirect	--	Variablenname
hmiDestTypeDirectMessage	--	Variablenname
hmiDestTypeIndirectMessage	--	Variablenname

Beispiel

Im folgenden Beispiel wird in Runtime bei einem Mausklick auf den Button die x-Position von "Rectangle_A" in die y-Position von "Rectangle_B" kopiert:

```
Sub DirectConnection()
'VBA650
Dim objButton As HMIButton
Dim objRectangleA As HMIRectangle
Dim objRectangleB As HMIRectangle
Dim objEvent As HMIEvent
Dim objDirConnection As HMIDirectConnection
Set objRectangleA = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_A", "HMIRectangle")
Set objRectangleB = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_B", "HMIRectangle")
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objRectangleA
.Top = 100
.Left = 100
End With
With objRectangleB
.Top = 250
.Left = 400
.BackColor = RGB(255, 0, 0)
End With
With objButton
.Top = 10
.Left = 10
.Width = 100
.Text = "SetPosition"
End With
'
'Directconnection is initiated by mouseclick:
Set objDirConnection =
objButton.Events(1).Actions.AddAction(hmiActionCreationTypeDirectConnection)
With objDirConnection
'Sourceobject: Property "Top" of Rectangle_A
.SourceLink.Type = hmiSourceTypeProperty
.SourceLink.ObjectName = "Rectangle_A"
```

```
.SourceLink.AutomationName = "Top"
'
'Targetobject: Property "Left" of Rectangle_B
.DestinationLink.Type = hmiDestTypeProperty
.DestinationLink.ObjectName = "Rectangle_B"
.DestinationLink.AutomationName = "Left"
End With
End Sub
```

Siehe auch

Type-Eigenschaft (Seite 2392)

SourceLink-Eigenschaft (Seite 2370)

DestinationLink-Eigenschaft (Seite 2162)

AutomationName-Eigenschaft (Seite 2082)

SourceLink-Objekt (Seite 2028)

DestLink-Objekt (Seite 1917)

ObjectSizeDecluttering-Eigenschaft

Beschreibung

TRUE, wenn die Objekte des angegebenen Bildes außerhalb von zwei projektierten Größen ausgeblendet werden sollen. BOOLEAN Schreib-Lese-Zugriff.

Den Größenbereich legen Sie mit der SetDeclutterObjectSize-Methode fest.

Beispiel

Im folgenden Beispiel werden im aktiven Bild die Einstellungen der untersten Ebene konfiguriert:

```
Sub ConfigureSettingsOfLayer()
'VBA651
Dim objLayer As HMILayer
Set objLayer = ActiveDocument.Layers(1)
With objLayer
'Configure "Layer 0"
.MinZoom = 10
.MaxZoom = 100
.Name = "Configured with VBA"
End With
'Define fade-in and fade-out of objects:
With ActiveDocument
.LayerDecluttering = True
.ObjectSizeDecluttering = True
.SetDeclutterObjectSize 50, 100
```

```
End With
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

Ebenen mit VBA bearbeiten (Seite 1659)

OffsetLeft-Eigenschaft

Beschreibung

Legt den Abstand des Bildes vom linken Rand des Bildfensters fest oder gibt ihn zurück.

Beispiel

Die Prozedur "PictureWindowConfig" greift auf die Eigenschaften des Bildfensters zu. In diesem Beispiel wird das Bildfenster konfiguriert:

```
Sub PictureWindowConfig()
'VBA652
Dim objPicWindow As HMIPictureWindow
Set objPicWindow = ActiveDocument.HMIObjects.AddHMIObject("PicWindow1", "HMIPictureWindow")
With objPicWindow
.AdaptPicture = False
.AdaptSize = False
.Caption = True
.CaptionText = "Picturewindow in runtime"
.OffsetLeft = 5
.OffsetTop = 10
'Replace the picturename "Test.PDL" with the name of
'an existing document from your "GraCS"-Folder of your active project
.PictureName = "Test.PDL"
.ScrollBars = True
.ServerPrefix = ""
.TagPrefix = "Struct."
.UpdateCycle = 5
.Zoom = 100
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

OffsetTop-Eigenschaft

Beschreibung

Legt den Abstand des Bildes vom oberen Rand des Bildfensters fest oder gibt ihn zurück.

Beispiel

Die Prozedur "PictureWindowConfig" greift auf die Eigenschaften des Bildfensters zu. In diesem Beispiel wird das Bildfenster konfiguriert:

```
Sub PictureWindowConfig()
'VBA653
Dim objPicWindow As HMIPictureWindow
Set objPicWindow = ActiveDocument.HMIObjects.AddHMIObject("PicWindow1", "HMIPictureWindow")
With objPicWindow
.AdaptPicture = False
.AdaptSize = False
.Caption = True
.CaptionText = "Picturewindow in runtime"
.OffsetLeft = 5
.OffsetTop = 10
'Replace the picturename "Test.PDL" with the name of
'an existing document from your "GraCS"-Folder of your active project
.PictureName = "Test.PDL"
.ScrollBars = True
.ServerPrefix = ""
.TagPrefix = "Struct."
.UpdateCycle = 5
.Zoom = 100
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

OnTop-Eigenschaft

Beschreibung

TRUE, wenn das Objekt ApplicationWindow in Runtime immer im Vordergrund bleibt. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "ApplicationWindowConfig" greift auf die Eigenschaften des Applikationsfensters zu. In diesem Beispiel soll das Applikationsfenster in Runtime immer im Vordergrund bleiben:

```
Sub ApplicationWindowConfig()
'VBA654
Dim objAppWindow As HMIApplicationWindow
Set objAppWindow = ActiveDocument.HMIObjects.AddHMIObject("AppWindow1",
"HMIApplicationWindow")
With objAppWindow
.OnTop = True
End With
End Sub
```

Siehe auch

ApplicationWindow-Objekt (Seite 1891)

Operation-Eigenschaft

Beschreibung

TRUE, wenn das Objekt in Runtime bedient werden kann. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

In diesem Beispiel wird der Status der Bedienfreigaben für alle Objekte im aktiven Bild ausgegeben:

```
Sub ShowOperationStatusOfAllObjects()
'VBA655
Dim objObject As HMIObject
Dim bStatus As Boolean
Dim strStatus As String
Dim strName As String
Dim iMax As Integer
Dim iIndex As Integer
Dim iAnswer As Integer
iMax = ActiveDocument.HMIObjects.Count
iIndex = 1
For iIndex = 1 To iMax
strName = ActiveDocument.HMIObjects(iIndex).ObjectName
bStatus = ActiveDocument.HMIObjects(iIndex).Operation
Select Case bStatus
Case True
 strStatus = "yes"
Case False
 strStatus = "no"
```

```
End Select
iAnswer = MsgBox("Object: " & strName & vbCrLf & "Operator-Control enable: " & strStatus,
vbOKCancel)
If vbCancel = iAnswer Then Exit For
Next iIndex
If 0 = iMax Then MsgBox "No objects in the active document."
End Sub
```

Siehe auch

HMIObject-Objekt (Seite 1955)

Document-Objekt (Seite 1920)

OperationMessage-Eigenschaft

Beschreibung

TRUE, wenn bei einer erfolgten Bedienung eine Meldung ausgegeben werden soll. Der Grund für die Bedienung kann nur eingegeben werden, wenn die Eigenschaft "OperationReport" auf "True" gesetzt ist. BOOLEAN Schreib-Lese-Zugriff.

Die Bedienung wird an das Meldesystem gesandt und archiviert. Über das Meldesystem kann eine Meldung z. B. in einer Meldezeile ausgeben werden.

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel soll eine Bedienung an das Meldesystem gesandt werden:

```
Sub IOFieldConfiguration()
'VBA656
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.OperationReport = True
.OperationMessage = True
End With
End Sub
```

Siehe auch

OperationReport-Eigenschaft (Seite 2314)

TextList-Objekt (Seite 2037)

Slider-Objekt (Seite 2025)

OptionGroup-Objekt (Seite 1989)

IOField-Objekt (Seite 1959)

CheckBox-Objekt (Seite 1901)

OperationReport-Eigenschaft

Beschreibung

TRUE, wenn der Grund für eine Bedienung mitprotokolliert werden soll. BOOLEAN Schreib-Lese-Zugriff.

Bei der Bedienung des Objekts in Runtime erscheint dann ein Dialog, in dem der Bediener den Grund der Bedienung als Text eingeben kann. Die Bedienung wird an das Meldesystem gesandt und archiviert.

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel soll eine Bedienung an das Meldesystem gesandt werden:

```
Sub IOFieldConfiguration()
'VBA657
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.OperationReport = True
.OperationMessage = True
End With
End Sub
```

Siehe auch

OperationMessage-Eigenschaft (Seite 2313)

TextList-Objekt (Seite 2037)

Slider-Objekt (Seite 2025)

OptionGroup-Objekt (Seite 1989)

IOField-Objekt (Seite 1959)

CheckBox-Objekt (Seite 1901)

Orientation-Eigenschaft

Beschreibung

TRUE, wenn der Text im Objekt horizontal dargestellt. BOOLEAN Schreib-Lese-Zugriff.

Hinweis

Es ist nur der Text, der entweder horizontal oder vertikal dargestellt wird. Die Lage des Objektes bleibt unverändert.

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel wird der Text vertikal dargestellt:

```
Sub ButtonConfiguration()
'VBA658
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.Width = 150
.Height = 150
.Text = "Text is displayed vertical"
.Orientation = False
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

IOField-Objekt (Seite 1959)

CheckBox-Objekt (Seite 1901)

Button-Objekt (Seite 1898)

OutputFormat-Eigenschaft

Beschreibung

Legt die Darstellung des Ausgabewerts fest oder gibt den eingestellten Wert zurück. Die Darstellung ist abhängig vom Datenformat.

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel wird für das EA-Feld der Datentyp "Dezimal" gesetzt. Der Ausgabewert wird mit zwei Stellen vor dem Komma und drei Nachkommastellen angezeigt werden:

```
Sub IOFieldConfiguration()
'VBA659
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.DataFormat = 1
.OutputFormat = "99,999"
End With
End Sub
```

Siehe auch

DataFormat-Eigenschaft (Seite 2161)

IOField-Objekt (Seite 1959)

OutputValue-Eigenschaft

Beschreibung

Legt die Voreinstellung für den anzuzeigenden Wert fest oder gibt sie zurück.

In Runtime wird dieser Wert verwendet, wenn beim Start des Bildes die zugehörige Variable nicht angeschlossen oder nicht aktualisiert ist.

Beispiel

Die Prozedur "IOFieldConfiguration()" greift auf die Eigenschaften des EA-Feldes zu. In diesem Beispiel wird der Ausgabewert auf "00" gesetzt:

```
Sub IOFieldConfiguration()
'VBA660
Dim objIOField As HMIIOField
Set objIOField = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
With objIOField
.OutputValue = "00"
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

IOField-Objekt (Seite 1959)

P-Q

Parent-Eigenschaft

Beschreibung

Gibt das übergeordnete Objekt des angegebenen Objektes zurück. Lese-Zugriff.

Beispiel

Im folgenden Beispiel wird vom aktiven Bild eine Kopie erzeugt und anschließend der Name des Bildes mit Hilfe der Parent-Eigenschaft ausgegeben:

```
Sub ExampleForParent()
'VBA661
Dim objView As HMIView
Set objView = ActiveDocument.Views.Add
MsgBox objView.Parent.Name
End Sub
```

Siehe auch

Toolbars-Objekt (Auflistung) (Seite 2041)
Menu-Objekt (Seite 1976)
Document-Objekt (Seite 1920)
Views-Objekt (Auflistung) (Seite 2064)
View-Objekt (Seite 2062)
VariableTriggers-Objekt (Auflistung) (Seite 2061)
VariableTrigger-Objekt (Seite 2060)
VariableStateValues-Objekt (Auflistung) (Seite 2058)
VariableStateValue-Objekt (Seite 2057)
Trigger-Objekt (Seite 2047)
ToolbarItems-Objekt (Auflistung) (Seite 2046)
ToolbarItem-Objekt (Seite 2043)
Toolbar-Objekt (Seite 2040)
TextList-Objekt (Seite 2037)
SymbolLibraries-Objekt (Auflistung) (Seite 2036)
SymbolLibrary-Objekt (Seite 2035)
StatusDisplay-Objekt (Seite 2032)
StaticText-Objekt (Seite 2029)
SourceLink-Objekt (Seite 2028)
Slider-Objekt (Seite 2025)
Selection-Objekt (Auflistung) (Seite 2022)
ScriptInfo-Objekt (Seite 2021)
RoundRectangle-Objekt (Seite 2018)
RoundButton-Objekt (Seite 2015)
Rectangle-Objekt (Seite 2012)
Properties-Objekt (Auflistung) (Seite 2004)
Property-Objekt (Seite 2005)
PolyLine-Objekt (Seite 2001)
Polygon-Objekt (Seite 1998)
PictureWindow-Objekt (Seite 1992)
PieSegment-Objekt (Seite 1995)
OptionGroup-Objekt (Seite 1989)
OLEObject-Objekt (Seite 1987)
MenuItems-Objekt (Auflistung) (Seite 1982)
MenuItem-Objekt (Seite 1979)
Menus-Objekt (Auflistung) (Seite 1977)
Line-Objekt (Seite 1970)
Layers-Objekt (Auflistung) (Seite 1969)
Layer-Objekt (Seite 1967)
LanguageTexts-Objekt (Auflistung) (Seite 1966)

PasswordLevel-Eigenschaft

Beschreibung

Legt die Berechtigung für die Bedienung (z.B. keine Eingabe oder keine Auslösung von Aktionen) für das Objekt fest.

PasswordLevel	zugeordneter Wert
<Kein Zugriffsschutz>	0
Benutzerverwaltung	1
Werteingabe	2
Prozessbedienung	3
Bildbearbeitung	4
Bildwechsel	5
Fensteranwahl	6
Hardcopy	7
Meldungen quittieren	8
Meldungen sperren	9
Meldungen freigeben	10
Meldungsbearbeitung	11
Archiv starten	12
Archiv stoppen	13
Archivwertbearbeitung	14
Archivbearbeitung	15
Aktionsbearbeitung	16
Projektmanager	17
Remote aktivieren	1000
Remote projektieren	1001
Nur beobachten	1002

Die Bedienberechtigungen müssen Sie zuvor im User Administrator festlegen.

Beispiel

--

Siehe auch

HMIObject-Objekt (Seite 1955)

Path-Eigenschaft

Beschreibung

Gibt den vollständigen Verzeichnispfad zurück, in dem das angegebene Bild gespeichert ist. Lese-Zugriff.

Beispiel

In diesem Beispiel wird der Verzeichnispfad des aktiven Bildes ausgegeben:

```
Sub ShowDocumentPath()
'VBA663
MsgBox ActiveDocument.Path
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

Pathname-Eigenschaft

Beschreibung

Gibt den internen Zugriffspfad in der Bausteinbibliothek des angegebenen Objektes vom Typ "FolderItem" zurück. Lese-Zugriff.

Beispiel

In diesem Beispiel wird der interne Zugriffspfad des Objektes "PC" ausgegeben, das sich in der Globalen Bausteinbibliothek befindet:

```
Sub ShowInternalNameOfFolderItem()
'VBA664
Dim objGlobalLib As HMISymbolLibrary
Set objGlobalLib = Application.SymbolLibraries(1)
MsgBox objGlobalLib.FolderItems(2).Folder(2).Folder.Item(1).PathName
End Sub
```

Siehe auch

FolderItem-Objekt (Seite 1939)

Zugriff auf die Bausteinbibliothek mit VBA (Seite 1648)

PdlProtection-Eigenschaft

Beschreibung

Vergibt dem Bild (Prozessbild oder Faceplate-Typ) ein Passwort oder löscht das Passwort. Schreib-Zugriff.

Hinweis
Bedeutung des Passwortschutzes

Beachten Sie, dass sich die PdlProtection-Eigenschaft nur auf das Öffnen eines Bildes bezieht.

Beispiele

In diesem Beispiel wird dem aktiven Bild ein Passwort gesetzt:

```
Sub ProtectPicture()
'VBAxxx
ActiveDocument.PdlProtection = "Test123"
End Sub
```

In diesem Beispiel wird das Passwort des aktiven passwortgeschützten Bildes gelöscht:

```
Sub UnprotectPicture()
'VBAxxx
ActiveDocument.PdlProtection = ""
End Sub
```

Hinweis
Nur Schreib-Zugriff

Beachten Sie, dass aus sicherheitstechnischen Gründen das Auslesen des Passworts nicht zulässig ist.

PicDeactReferenced-Eigenschaft

Beschreibung

TRUE, wenn das zugeordnete Bild für den Zustand "Deaktiviert" im Objekt RoundButton gespeichert werden soll. Sonst wird nur der zugehörige Objektverweis gespeichert. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "RoundButtonConfiguration()" greift auf die Eigenschaften des RoundButton zu. In diesem Beispiel wird das dem Zustand "Deaktiviert" zugeordnete Bild referenziert:

```
Sub RoundButtonConfiguration()
'VBA665
Dim objRoundButton As HMIRoundButton
Set objRoundButton = ActiveDocument.HMIObjects.AddHMIObject("RButton1", "HMIRoundButton")
With objRoundButton
.PicDeactReferenced = False
End With
End Sub
```

Siehe auch

RoundButton-Objekt (Seite 2015)

PicDeactTransparent-Eigenschaft

Beschreibung

Legt fest, welche Farbe des zugeordneten Bitmap-Objekts (.bmp, .dib) für den Zustand "Deaktiviert" auf "transparent" gesetzt werden soll oder gibt die Farbe zurück. LONG Schreib-Lese-Zugriff.

Die Farbe wird nur dann auf "transparent" gesetzt, wenn die Eigenschaft "PicDeactUseTransColor" den Wert "True" hat.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "RoundButtonConfiguration()" greift auf die Eigenschaften des RoundButton zu. In diesem Beispiel soll die Farbe "Rot" des zugeordneten Bitmap-Objektes im Zustand "Deaktiviert" transparent dargestellt werden:

```
Sub RoundButtonConfiguration()
'VBA666
Dim objRoundButton As HMIRoundButton
Set objRoundButton = ActiveDocument.HMIObjects.AddHMIObject("RButton1", "HMIRoundButton")
With objRoundButton
.PicDeactTransparent = RGB(255, 0, 0)
.PicDeactUseTransColor = True
End With
End Sub
```

Siehe auch

PicDeactUseTransColor-Eigenschaft (Seite 2323)

RoundButton-Objekt (Seite 2015)

PicDeactUseTransColor-Eigenschaft

Beschreibung

TRUE, wenn die mit der Eigenschaft "PicDeactTransparent" festgelegte Transparentfarbe für den Zustand "Deaktiviert" verwendet werden soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "RoundButtonConfiguration()" greift auf die Eigenschaften des RoundButton zu. In diesem Beispiel soll die Farbe "Rot" des zugeordneten Bitmap-Objektes im Zustand "Deaktiviert" transparent dargestellt werden:

```
Sub RoundButtonConfiguration()
'VBA667
Dim objRoundButton As HMIRoundButton
Set objRoundButton = ActiveDocument.HMIObjects.AddHMIObject("RButton1", "HMIRoundButton")
With objRoundButton
.PicDeactTransparent = RGB(255, 0, 0)
.PicDeactUseTransColor = True
End With
End Sub
```

Siehe auch

PicDeactTransparent-Eigenschaft (Seite 2322)

RoundButton-Objekt (Seite 2015)

PicDownReferenced-Eigenschaft

Beschreibung

TRUE, wenn das zugeordnete Bild für den Zustand "Ein" im Objekt RoundButton gespeichert werden soll. Sonst wird nur der zugehörige Objektverweis gespeichert. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "RoundButtonConfiguration()" greift auf die Eigenschaften des RoundButton zu. In diesem Beispiel wird das dem Zustand "Ein" zugeordnete Bild referenziert:

```
Sub RoundButtonConfiguration()
'VBA668
Dim objRoundButton As HMIRoundButton
Set objRoundButton = ActiveDocument.HMIObjects.AddHMIObject("RButton1", "HMIRoundButton")
With objRoundButton
.PicDownReferenced = False
End With
End Sub
```

Siehe auch

RoundButton-Objekt (Seite 2015)

PicDownTransparent-Eigenschaft

Beschreibung

Legt fest, welche Farbe des zugeordneten Bitmap-Objekts (.bmp, .dib) für den Zustand "Ein" auf "transparent" gesetzt werden soll oder gibt die Farbe zurück. LONG Schreib-Lese-Zugriff.

Die Farbe wird nur dann auf "transparent" gesetzt, wenn die Eigenschaft "PicDownUseTransColor" den Wert "True" hat.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "RoundButtonConfiguration()" greift auf die Eigenschaften des RoundButton zu. In diesem Beispiel soll die Farbe "Gelb" des zugeordneten Bitmap-Objektes im Zustand "Ein" transparent dargestellt werden:

```
Sub RoundButtonConfiguration()
'VBA669
Dim objRoundButton As HMIRoundButton
Set objRoundButton = ActiveDocument.HMIObjects.AddHMIObject("RButton1", "HMIRoundButton")
With objRoundButton
.PicDownTransparent = RGB(255, 255, 0)
.PicDownUseTransColor = True
End With
End Sub
```

Siehe auch

PicDownUseTransColor-Eigenschaft (Seite 2325)

RoundButton-Objekt (Seite 2015)

PicDownUseTransColor-Eigenschaft

Beschreibung

TRUE, wenn die mit der Eigenschaft "PicDownTransparent" festgelegte Transparentfarbe für den Zustand "Ein" verwendet werden soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "RoundButtonConfiguration()" greift auf die Eigenschaften des RoundButton zu. In diesem Beispiel soll die Farbe "Gelb" des zugeordneten Bitmap-Objektes im Zustand "Ein" transparent dargestellt werden:

```
Sub RoundButtonConfiguration()
'VBA670
Dim objRoundButton As HMIRoundButton
Set objRoundButton = ActiveDocument.HMIObjects.AddHMIObject("RButton1", "HMIRoundButton")
With objRoundButton
.PicDownTransparent = RGB(255, 255, 0)
.PicDownUseTransColor = True
End With
End Sub
```

Siehe auch

PicDownTransparent-Eigenschaft (Seite 2324)

RoundButton-Objekt (Seite 2015)

PicReferenced-Eigenschaft

Beschreibung

TRUE, wenn das dem Objekt GraphicObject zugeordnete Bild referenziert und nicht im Objekt gespeichert wird. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "GraphicObjectConfiguration()" greift auf die Eigenschaften des Grafik-Objektes zu. In diesem Beispiel soll das zugeordnete Bild referenziert werden:

```
Sub GraphicObjectConfiguration()
'VBA671
Dim objGraphicObject As HMIGraphicObject
Set objGraphicObject = ActiveDocument.HMIObjects.AddHMIObject("GraphicObject1",
"HMIGraphicObject")
With objGraphicObject
.PicReferenced = True
End With
End Sub
```

Siehe auch

GraphicObject-Objekt (Seite 1943)

PictAlignment-Eigenschaft

Beschreibung

Legt als Attribut "Bildausrichtung" die Position und Skalierung des auf dem Button bzw. Rundbutton platzierten Bildes fest.

zentriert	Das Bild wird in Originalproportionen mittig platziert.
linksbündig	Das Bild wird in Originalproportionen linksbündig auf der linken Seite des Buttons platziert.
rechtsbündig	Das Bild wird in Originalproportionen rechtsbündig auf der rechten Seite des Buttons platziert.
gestreckt	Das Bild wird zum Quadrat skaliert und an die Größe des Buttons angepasst.

PicTransColor-Eigenschaft

Beschreibung

Legt fest, welche Farbe des zugeordneten Bitmap-Objekts (.bmp, .dib) auf "transparent" gesetzt werden soll oder gibt die Farbe zurück. LONG Schreib-Lese-Zugriff.

Die Farbe wird nur dann auf "transparent" gesetzt, wenn die Eigenschaft "PicUseTransColor" den Wert "True" hat.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "GraphicObjectConfiguration()" greift auf die Eigenschaften des Grafik-Objektes zu. In diesem Beispiel soll die Farbe "Blau" des zugeordneten Bitmap-Objektes transparent dargestellt werden:

```
Sub GraphicObjectConfiguration()
'VBA672
Dim objGraphicObject As HMIGraphicObject
Set objGraphicObject = ActiveDocument.HMIObjects.AddHMIObject("GraphicObject1",
"HMIGraphicObject")
With objGraphicObject
.PicTransColor = 16711680
.PicUseTransColor = True
End With
End Sub
```

Siehe auch

GraphicObject-Objekt (Seite 1943)

PictureDeactivated-Eigenschaft

Beschreibung

Legt das Bild fest, das im Zustand "Deaktiviert" angezeigt wird oder gibt den Bildnamen zurück.

Das Bild (*.BMP oder *.DIB) muss sich im Verzeichnis "GraCS" des aktuellen Projekts befinden, damit es eingebunden werden kann.

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Rundbuttons zu. In diesem Beispiel werden die Bilder für die Zustände "Ein" und "Aus" festgelegt:

```
Sub ButtonConfiguration()
'VBA673
Dim objRoundButton As HMIRoundButton
Set objRoundButton = ActiveDocument.HMIObjects.AddHMIObject("RButton1", "HMIRoundButton")
With objRoundButton
'
'Toi use this example copy a Bitmap-Graphic
'to the "GraCS"-Folder of the actual project.
'Replace the picturename "TestPicture1.BMP" with the name of
'the picture you copied
.PictureDeactivated = "TestPicture1.BMP"
End With
End Sub
```

Siehe auch

RoundButton-Objekt (Seite 2015)

PicReferenced-Eigenschaft (Seite 2326)

PictureDown-Eigenschaft

Beschreibung

Legt das Bild fest, das im Zustand "Ein" angezeigt wird oder gibt den Bildnamen zurück.

Das Bild (*.BMP oder *.DIB) muss sich im Verzeichnis "GraCS" des aktuellen Projekts befinden, damit es eingebunden werden kann.

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Rundbuttons zu. In diesem Beispiel werden die Bilder für die Zustände "Ein" und "Aus" festgelegt:

```
Sub ButtonConfiguration()
'VBA674
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
'
'To use this example copy two Bitmap-Graphics
'to the "GraCS"-Folder of the actual project.
'Replace the picturenames "TestPicture1.BMP" and "TestPicture2.BMP"
'with the names of the pictures you copied
.PictureDown = "TestPicture1.BMP"
```

```
.PictureUp = "TestPicture2.BMP"
End With
End Sub
```

Siehe auch

RoundButton-Objekt (Seite 2015)

PictureName-Eigenschaft

Beschreibung

Legt das Bild fest, das im Bildfenster in Runtime angezeigt wird oder gibt den Bildnamen zurück.

Das Bild (*.BMP oder *.DIB) muss sich im Verzeichnis "GraCS" des aktuellen Projekts befinden, damit es eingebunden werden kann.

Beispiel

Die Prozedur "PictureWindowConfig" greift auf die Eigenschaften des Bildfensters zu. In diesem Beispiel wird das Bildfenster konfiguriert:

```
Sub PictureWindowConfig()
'VBA675
Dim objPicWindow As HMIPictureWindow
Set objPicWindow = ActiveDocument.HMIObjects.AddHMIObject("PicWindow1", "HMIPictureWindow")
With objPicWindow
.AdaptPicture = False
.AdaptSize = False
.Caption = True
.CaptionText = "Picturewindow in runtime"
.OffsetLeft = 5
.OffsetTop = 10
'Replace the picturename "Test.PDL" with the name of
'an existing document from your "GraCS"-Folder of your active project
.PictureName = "Test.PDL"
.ScrollBars = True
.ServerPrefix = ""
.TagPrefix = "Struct."
.UpdateCycle = 5
.Zoom = 100
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

PictureUp-Eigenschaft

Beschreibung

Legt das Bild fest, das im Zustand "Aus" angezeigt wird oder gibt den Bildnamen zurück.

Das Bild (*.BMP oder *.DIB) muss sich im Verzeichnis "GraCS" des aktuellen Projekts befinden, damit es eingebunden werden kann.

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel werden die Bilder für die Zustände "Ein" und "Aus" festgelegt:

```
Sub ButtonConfiguration()
'VBA676
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
'
'To use this example copy two Bitmap-Graphics
'to the "GraCS"-Folder of the actual project.
'Replace the picturenames "TestPicture1.BMP" and "TestPicture2.BMP"
'with the names of the pictures you copied
.PictureDown = "TestPicture1.BMP"
.PictureUp = "TestPicture2.BMP"
End With
End Sub
```

Siehe auch

RoundButton-Objekt (Seite 2015)

Button-Objekt (Seite 1898)

PicUpReferenced-Eigenschaft

Beschreibung

TRUE, wenn das zugeordnete Bild für den Zustand "Aus" im Objekt RoundButton gespeichert werden soll. Sonst wird nur der zugehörige Objektverweis gespeichert. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "RoundButtonConfiguration()" greift auf die Eigenschaften des RoundButton zu. In diesem Beispiel wird das dem Zustand "Aus" zugeordnete Bild referenziert:

```
Sub RoundButtonConfiguration()
'VBA677
Dim objRoundButton As HMIRoundButton
Set objRoundButton = ActiveDocument.HMIObjects.AddHMIObject("RButton1", "HMIRoundButton")
With objRoundButton
.PicUpReferenced = False
End With
End Sub
```

Siehe auch

RoundButton-Objekt (Seite 2015)

PicUpTransparent-Eigenschaft

Beschreibung

Legt fest, welche Farbe des zugeordneten Bitmap-Objekts (.bmp, .dib) für den Zustand "Aus" auf "transparent" gesetzt werden soll oder gibt die Farbe zurück. LONG Schreib-Lese-Zugriff.

Die Farbe wird nur dann auf "transparent" gesetzt, wenn die Eigenschaft "PicUpUseTransColor" den Wert "True" hat.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "RoundButtonConfiguration()" greift auf die Eigenschaften des RoundButton zu. In diesem Beispiel soll die Farbe "Blau" des zugeordneten Bitmap-Objektes im Zustand "Aus" transparent dargestellt werden:

```
Sub RoundButtonConfiguration()
'VBA678
Dim objRoundButton As HMIRoundButton
Set objRoundButton = ActiveDocument.HMIObjects.AddHMIObject("RButton1", "HMIRoundButton")
With objRoundButton
.PicUpTransparent = RGB(0, 0, 255)
.PicUpUseTransColor = True
End With
```

```
End Sub
```

Siehe auch

PicUpUseTransColor-Eigenschaft (Seite 2332)

RoundButton-Objekt (Seite 2015)

PicUpUseTransColor-Eigenschaft

Beschreibung

TRUE, wenn die mit der Eigenschaft "PicUpTransparent" festgelegte Transparentfarbe für den Zustand "Aus" verwendet werden soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "RoundButtonConfiguration()" greift auf die Eigenschaften des RoundButton zu. In diesem Beispiel soll die Farbe "Blau" des zugeordneten Bitmap-Objektes im Zustand "Aus" transparent dargestellt werden:

```
Sub RoundButtonConfiguration()
'VBA679
Dim objRoundButton As HMIRoundButton
Set objRoundButton = ActiveDocument.HMIObjects.AddHMIObject("RButton1", "HMIRoundButton")
With objRoundButton
.PicUpTransparent = RGB(0, 0, 255)
.PicUpUseTransColor = True
End With
End Sub
```

Siehe auch

PicUpTransparent-Eigenschaft (Seite 2331)

RoundButton-Objekt (Seite 2015)

PicUseTransColor-Eigenschaft

Beschreibung

TRUE, wenn die mit der Eigenschaft "PicTransColor" festgelegte Transparentfarbe für den Zustand "Deaktiviert" verwendet werden soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "GraphicObjectConfiguration()" greift auf die Eigenschaften des Grafik-Objektes zu. In diesem Beispiel soll die Farbe "Blau" des zugeordneten Bitmap-Objektes transparent dargestellt werden:

```
Sub GraphicObjectConfiguration()
'VBA680
Dim objGraphicObject As HMIGraphicObject
Set objGraphicObject = ActiveDocument.HMIObjects.AddHMIObject("GraphicObject1",
"HMIGraphicObject")
With objGraphicObject
.PicTransColor = RGB(0, 0, 255)
.PicUseTransColor = True
End With
End Sub
```

Siehe auch

PicTransColor-Eigenschaft (Seite 2327)

GraphicObject-Objekt (Seite 1943)

PointCount-Eigenschaft

Beschreibung

Legt bei den Objekten Polygon und PolyLine die Anzahl der Eckpunkte fest oder gibt sie zurück. Jeder Eckpunkt hat Positionskoordinaten und wird über einen Index identifiziert.

Beispiel

Damit dieses Beispiel funktioniert, fügen Sie einen Polygonzug "Polygonzug1" in das aktive Bild ein. Die Prozedur "PolyLineCoordsOutput" gibt dann die Koordinaten aller Eckpunkte des Polygonzuges aus:

```
Sub PolyLineCoordsOutput()
'VBA681
Dim iPcIndex As Integer
Dim iPosX As Integer
Dim iPosY As Integer
Dim iIndex As Integer
Dim objPolyLine As HMIPolyLine
Set objPolyLine = Application.ActiveDocument.HMIObjects.AddHMIObject("PolyLine1",
"HMIPolyLine")

'
'Determine number of corners from "PolyLine1":
iPcIndex = objPolyLine.PointCount
```

```
'
'Output of x/y-coordinates from every corner:
For iIndex = 1 To iPcIndex
With objPolyLine
.index = iIndex
iPosX = .ActualPointLeft
iPosY = .ActualPointTop
MsgBox iIndex & ". corner:" & vbCrLf & "x-coordinate: " & iPosX & vbCrLf & "y-coordinate:
" & iPosY
End With
Next iIndex
End Sub
```

Auflistung der Verlinkung

Siehe auch

Index-Eigenschaft (Seite 2219)

ActualPointTop-Eigenschaft (Seite 2069)

ActualPointLeft-Eigenschaft (Seite 2068)

PolyLine-Objekt (Seite 2001)

Polygon-Objekt (Seite 1998)

Position-Eigenschaft

Beschreibung

Der Wert von Position entscheidet über die Reihenfolge, in der Menüeinträge und Symbol-Icons in benutzerdefinierten Menüs und Symbolleisten oder benutzerdefinierte Menüs in der Menüleiste angeordnet sind. Schreib-Lese-Zugriff.

Ein Wert von "1" bedeutet Position 1 (Anfang).

Beispiel

Im folgenden Beispiel wird die Position aller Menüeinträge des ersten benutzerdefinierten Menüs im aktiven Bild ausgegeben. Damit dieses Beispiel funktioniert, führen Sie das Beispiel unter "InsertSubMenu" zuerst aus.

```
Sub ShowPositionOfCustomMenuItems()
'VBA683
Dim objMenu As HMIMenu
Dim iMaxMenuItems As Integer
Dim iPosition As Integer
Dim iIndex As Integer
Set objMenu = ActiveDocument.CustomMenus(1)
iMaxMenuItems = objMenu.MenuItems.Count
```

```
For iIndex = 1 To iMaxMenuItems
iPosition = objMenu.MenuItems(iIndex).Position
MsgBox "Position of the " & iIndex & ". menuitem: " & iPosition
Next iIndex
End Sub
```

Siehe auch

ToolbarItem-Objekt (Seite 2043)

MenuItem-Objekt (Seite 1979)

Menu-Objekt (Seite 1976)

InsertSubmenu-Methode (Seite 1840)

PositiveValue-Eigenschaft

Beschreibung

Legt den Wert der dynamisierten Eigenschaft fest, wenn die projektierte Variable einen Wert ungleich Null zurückliefert oder gibt den Wert zurück.

Beispiel

Verwenden Sie die BinaryResultInfo-Eigenschaft, um das BinaryResultInfo-Objekt zurückzugeben. Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert, ein Variablenname vergeben und den beiden binären Wertebereichen die dazugehörigen Eigenschaftswerte zugeordnet:

```
Sub AddDynamicDialogToCircleRadiusTypeBool()
'VBA684
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_C", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeBool
.BinaryResultInfo.NegativeValue = 20
.BinaryResultInfo.PositiveValue = 40
End With
End Sub
```

Siehe auch

NegativeValue-Eigenschaft (Seite 2305)

BinaryResultInfo-Objekt (Seite 1896)

VBA-Referenz (Seite 1735)

PredefinedAngels-Eigenschaft

Beschreibung

Legt die Tiefendarstellung beim Objekt 3DBarGraph fest oder gibt sie zurück. Wertebereich von 0 bis 3.

Darstellung	zugeordneter Wert
Kavalier	0
Isometrisch	1
Axonometrisch	2
Frei definiert	3

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel wird die Tiefendarstellung auf "Isometrisch" gesetzt:

```
Sub HMI3DBarGraphConfiguration()
'VBA685
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
'Depth-angle a = 15 degrees
.AngleAlpha = 15
.PredefinedAngles = 1
'Depth-angle b = 45 degrees
.AngleBeta = 45
End With
End Sub
```

Siehe auch

3DBarGraph-Objekt (Seite 1879)

Pressed-Eigenschaft

Beschreibung

TRUE, wenn das Objekt Button oder RoundButton gedrückt ist. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "RoundButtonConfiguration()" greift auf die Eigenschaften des RoundButton zu. In diesem Beispiel wird der RoundButton in den Zustand "gedrückt" gesetzt:

```
Sub RoundButtonConfiguration()
'VBA686
Dim objRoundButton As HMIRoundButton
Set objRoundButton = ActiveDocument.HMIObjects.AddHMIObject("RButton1", "HMIRoundButton")
With objRoundButton
.Pressed = True
End With
End Sub
```

Siehe auch

RoundButton-Objekt (Seite 2015)

Process-Eigenschaft

Beschreibung

Legt die Voreinstellung für den anzuzeigenden Wert fest oder gibt sie zurück.

In Runtime wird dieser Wert verwendet, wenn beim Start des Bildes die zugehörige Variable nicht angeschlossen oder nicht aktualisiert ist.

Beispiel

Die Prozedur "HMI3DBarGraphConfiguration()" greift auf die Eigenschaften des 3DBarGraph-Objektes zu. In diesem Beispiel der Wert auf "100" voreingestellt:

```
Sub HMI3DBarGraphConfiguration()
'VBA687
Dim obj3DBar As HMI3DBarGraph
Set obj3DBar = ActiveDocument.HMIObjects.AddHMIObject("3DBar1", "HMI3DBarGraph")
With obj3DBar
'Depth-angle a = 15 degrees
.AngleAlpha = 15
'Depth-angle b = 45 degrees
```

```
.AngleBeta = 45
.Process = 100
End With
End Sub
```

Siehe auch

Slider-Objekt (Seite 2025)

OptionGroup-Objekt (Seite 1989)

CheckBox-Objekt (Seite 1901)

BarGraph-Objekt (Seite 1893)

3DBarGraph-Objekt (Seite 1879)

ProfileName-Eigenschaft

Beschreibung

Gibt den Namen der angegebenen Anwendung zurück. Lese-Zugriff.

Beispiel

In diesem Beispiel wird der Name der Anwendung "Graphics Designer" ausgegeben:

```
Sub ShowProfileName()
'VBA688
MsgBox Application.ProfileName
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

ProgID-Eigenschaft

Beschreibung

Gibt die ProgID eines ActiveX Controls zurück. STRING Lese-Zugriff.

Beispiel

Im folgenden Beispiel wird das ActiveX Control "WinCC Gauge Control" in das aktive Bild eingefügt. Anschließend wird die ProgID ausgegeben:

```
Sub AddActiveXControl()
'VBA689
Dim objActiveXControl As HMIActiveXControl
Set objActiveXControl = ActiveDocument.HMIObjects.AddActiveXControl("WinCC_Gauge",
"XGAUGE.XGaugeCtrl.1")
With ActiveDocument
.HMIObjects("WinCC_Gauge").Top = 40
.HMIObjects("WinCC_Gauge").Left = 40
MsgBox "ProgID of ActiveX-control: " & .HMIObjects("WinCC_Gauge").ProgID
End With
End Sub
```

Siehe auch

ActiveXControl-Objekt (Seite 1885)

AddActiveXControl-Methode (Seite 1785)

ProjectName-Eigenschaft

Beschreibung

Gibt den Projektnamen zurück. Lese-Zugriff

Beispiel

In diesem Beispiel werden Projektname und -typ des geladenen Projektes ausgegeben:

```
Sub ShowProjectInfo()
'VBA690
Dim iProjectType As Integer
Dim strProjectName As String
Dim strProjectType As String
iProjectType = Application.ProjectType
strProjectName = Application.ProjectName
Select Case iProjectType
Case 0
strProjectType = "Single-User System"
Case 1
strProjectType = "Multi-User System"
Case 2
strProjectType = "Client System"
End Select
MsgBox "Projecttype: " & strProjectType & vbCrLf & "Projectname: " & strProjectName
```

```
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

ProjectType-Eigenschaft

Beschreibung

Gibt den Projekttyp zurück. Wertebereich von 0 bis 2. Lese-Zugriff.

Projekttyp	zugeordneter Wert
Einzelplatzprojekt	0
Mehrplatzprojekt	1
Client-Projekt	2

Beispiel

In diesem Beispiel werden Projektname und -typ des geladenen Projektes ausgegeben:

```
Sub ShowProjectInfo()
'VBA691
Dim iProjectType As Integer
Dim strProjectName As String
Dim strProjectType As String
iProjectType = Application.ProjectType
strProjectName = Application.ProjectName
Select Case iProjectType
Case 0
strProjectType = "Single-User System"
Case 1
strProjectType = "Multi-User System"
Case 2
strProjectType = "Client System"
End Select
MsgBox "Projecttype: " & strProjectType & vbCrLf & "Projectname: " & strProjectName
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

Properties-Eigenschaft

Beschreibung

Gibt eine Properties-Auflistung zurück, die alle Eigenschaften des angegebenen Objektes enthält. Lese-Zugriff.

Um ein Element aus der Properties-Auflistung zurückzugeben, können Sie entweder die Indexnummer oder den VBA-Eigenschaftsnamen verwenden.

Sie müssen die Properties-Eigenschaft verwenden, wenn Sie z.B. auf die Objekteigenschaften von Objekten zugreifen wollen, die sich in einem Gruppen-Objekt befinden.

Beispiel

Beispiele zur Anwendung der Properties-Eigenschaft finden Sie in dieser Dokumentation unter:

- "Objekte mit VBA bearbeiten"
- "Gruppen-Objekte"
- "Anwender-Objekte"

Siehe auch

HMIObject-Objekt (Seite 1955)

Anwender-Objekte (Seite 1687)

Gruppen-Objekte (Seite 1679)

Objekte mit VBA bearbeiten (Seite 1662)

Prototype-Eigenschaft

Beschreibung

Gibt den Funktionskopf eines Skriptes zurück. Der Funktionskopf wird standardmäßig vergeben, wenn kein Quellcode projektiert wird.

Beispiel

Im folgenden Beispiel werden in das aktive Bild ein Button und ein Kreis eingefügt. In Runtime soll bei jedem Klick auf den Button der Radius des Kreises vergrößert werden. In diesem Fall wird nur der Prototyp des VB-Skriptes ausgegeben:

```
Sub ExampleForPrototype()
'VBA692
Dim objButton As HMIButton
Dim objCircleA As HMICircle
```

```
Dim objEvent As HMIEvent
Dim objVBScript As HMIScriptInfo
Set objCircleA = ActiveDocument.HMIObjects.AddHMIObject("CircleA", "HMICircle")
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objCircleA
.Top = 100
.Left = 100
End With
With objButton
.Top = 10
.Left = 10
.Width = 200
.Text = "Increase Radius"
End With
'On every mouseclick the radius have to increase:
Set objEvent = objButton.Events(1)
Set objVBScript = objButton.Events(1).Actions.AddAction(hmiActionCreationTypeVBScript)
MsgBox objVBScript.Prototype
End Sub
```

Siehe auch

ScriptInfo-Objekt (Seite 2021)

QualityCodeStateChecked-Eigenschaft

Beschreibung

TRUE, wenn der Quality Code der angegebenen Variable im Dynamik-Dialog zur Dynamisierung verwendet wird. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA816
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
```

```
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

DynamicDialog-Objekt (Seite 1924)

QualityCodeStateValues-Eigenschaft

Beschreibung

Gibt die QualityCodeStateValues-Auflistung zurück. Verwenden Sie die QualityCodeStateValues-Eigenschaft mit der Item-Eigenschaft, um einem Quality Code-Status einen Wert zuzuordnen, der für die Dynamisierung verwendet wird.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
```

```
'VBA817
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

DynamicDialog-Objekt (Seite 1924)

QualityCodeStateValues-Objekt (Auflistung) (Seite 2009)

R

Radius-Eigenschaft

Beschreibung

Legt bei folgenden Objekten den Radius fest oder gibt ihn zurück:

- Circle: Radius in Pixel (0 bis 10000)
- CircularArc: Radius in Pixel (0 bis 10000)
- PieSegment: Radius in Pixel (0 bis 10000)
- RoundButton: Radius in Pixel (0 bis 10000)

Beispiel

Die Prozedur "PieSegmentConfiguration()" greift auf die Eigenschaften des Kreissegmentes zu. In diesem Beispiel wird der Radius auf "80" gesetzt:

```
Sub PieSegmentConfiguration()
'VBA693
Dim objPieSegment As HMIPieSegment
Set objPieSegment = ActiveDocument.HMIObjects.AddHMIObject("PieSegment1", "HMIPieSegment")
With objPieSegment
.StartAngle = 40
.EndAngle = 180
.Radius = 80
End With
End Sub
```

Siehe auch

RoundButton-Objekt (Seite 2015)

PieSegment-Objekt (Seite 1995)

CircularArc-Objekt (Seite 1905)

Circle-Objekt (Seite 1902)

RadiusHeight-Eigenschaft

Beschreibung

Legt bei elliptischen Objekten (Ellipse, EllipseArc, EllipseSegment) den vertikalen Radius in Pixel (0 bis 5000) fest oder gibt ihn zurück.

Beispiel

Die Prozedur "EllipseConfiguration()" greift auf die Eigenschaften des Objektes Ellipse zu. In diesem Beispiel wird vertikale Radius auf "60" gesetzt:

```
Sub EllipseConfiguration()
'VBA694
Dim objEllipse As HMIEllipse
Set objEllipse = ActiveDocument.HMIObjects.AddHMIObject("Ellipse1", "HMIEllipse")
With objEllipse
.RadiusHeight = 60
.RadiusWidth = 40
End With
End Sub
```

Siehe auch

RadiusWidth-Eigenschaft (Seite 2346)

EllipseSegment-Objekt (Seite 1932)

EllipseArc-Objekt (Seite 1929)

Ellipse-Objekt (Seite 1926)

RadiusWidth-Eigenschaft

Beschreibung

Legt bei elliptischen Objekten (Ellipse, EllipseArc, EllipseSegment) den horizontalen Radius in Pixel (0 bis 5000) fest oder gibt ihn zurück.

Beispiel

Die Prozedur "EllipseConfiguration()" greift auf die Eigenschaften des Objektes Ellipse zu. In diesem Beispiel wird vertikale Radius auf "40" gesetzt:

```
Sub EllipseConfiguration()
'VBA695
Dim objEllipse As HMIEllipse
Set objEllipse = ActiveDocument.HMIObjects.AddHMIObject("Ellipse1", "HMIEllipse")
With objEllipse
.RadiusHeight = 60
.RadiusWidth = 40
End With
End Sub
```

Siehe auch

RadiusHeight-Eigenschaft (Seite 2345)

EllipseSegment-Objekt (Seite 1932)

EllipseArc-Objekt (Seite 1929)

Ellipse-Objekt (Seite 1926)

RangeTo-Eigenschaft

Beschreibung

Legt den analogen Wertebereich fest oder gibt ihn zurück.

Beispiel

Ein Beispiel zur Anwendung der RangeTo-Eigenschaft finden Sie in dieser Dokumentation unter "AnalogResultInfos-Objekt (Auflistung)".

Siehe auch

Value-Eigenschaft (Seite 2408)

AnalogResultInfos-Objekt (Auflistung) (Seite 1887)

AnalogResultInfo-Objekt (Seite 1886)

ReferenceRotationLeft-Eigenschaft

Beschreibung

Legt die x-Koordinate des Referenzpunktes fest, um den das Objekt in Runtime gedreht werden soll oder gibt sie zurück.

Der Wert der x-Koordinate ist relativ zur Objektbreite. Geben Sie den Wert ausgehend von der linken Kante des objektumfassenden Rechteckes in Prozent an.

Beispiel

Die Prozedur "PolyLineConfiguration()" greift auf die Eigenschaften des Objektes PolyLine zu. In diesem Beispiel werden die Koordinaten des Referenzpunktes auf 50% der Objektbreite und 50% der Objekthöhe gesetzt:

```
Sub PolyLineConfiguration()
'VBA696
Dim objPolyLine As HMIPolyLine
Set objPolyLine = ActiveDocument.HMIObjects.AddHMIObject("PolyLine1", "HMIPolyLine")
With objPolyLine
```

```
.ReferenceRotationLeft = 50
.ReferenceRotationTop = 50
End With
End Sub
```

Siehe auch

RotationAngle-Eigenschaft (Seite 2351)

ReferenceRotationTop-Eigenschaft (Seite 2348)

PolyLine-Objekt (Seite 2001)

Polygon-Objekt (Seite 1998)

Line-Objekt (Seite 1970)

ReferenceRotationTop-Eigenschaft

Beschreibung

Legt die y-Koordinate des Referenzpunktes fest, um den das Objekt in Runtime gedreht werden soll oder gibt sie zurück.

Der Wert der y-Koordinate ist relativ zur Objektbreite. Geben Sie den Wert ausgehend von der oberen Kante des objektumfassenden Rechteckes in Prozent an.

Beispiel

Die Prozedur "PolyLineConfiguration()" greift auf die Eigenschaften des Objektes PolyLine zu. In diesem Beispiel werden die Koordinaten des Referenzpunktes auf 50% der Objektbreite und 50% der Objekthöhe gesetzt:

```
Sub PolyLineConfiguration()
'VBA697
Dim objPolyLine As HMIPolyLine
Set objPolyLine = ActiveDocument.HMIObjects.AddHMIObject("PolyLine1", "HMIPolyLine")
With objPolyLine
.ReferenceRotationLeft = 50
.ReferenceRotationTop = 50
End With
End Sub
```

Siehe auch

RotationAngle-Eigenschaft (Seite 2351)

ReferenceRotationLeft-Eigenschaft (Seite 2347)

PolyLine-Objekt (Seite 2001)

Polygon-Objekt (Seite 1998)

Line-Objekt (Seite 1970)

Relevant-Eigenschaft

Beschreibung

TRUE, wenn das Objekt für die Bildung der Sammelanzeige berücksichtigt wird. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel das Objekt für die Bildung der Sammelanzeige berücksichtigt werden:

```
Sub GroupDisplayConfiguration()
'VBA698
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.Relevant = True
End With
End Sub
```

Siehe auch

Group-Objekt (Seite 1946)

ResultType-Eigenschaft

Beschreibung

Legt im Dynamik-Dialog die Art der Auswertung der Wertebereiche fest oder gibt sie zurück.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert, ein Variablenname vergeben und den beiden binären Wertebereichen die dazugehörigen Eigenschaftswerte zugeordnet:

```
Sub AddDynamicDialogToCircleRadiusTypeBinary()
'VBA699
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_C", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeBool
.BinaryResultInfo.NegativeValue = 20
.BinaryResultInfo.PositiveValue = 40
End With
End Sub
```

Siehe auch

DynamicDialog-Objekt (Seite 1924)

RightComma-Eigenschaft

Beschreibung

Legt beim Objekt BarGraph die Anzahl der Nachkommastellen (0 bis 20) fest oder gibt sie zurück.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird die Anzahl der Nachkommastellen auf 4 begrenzt.

```
Sub BarGraphConfiguration()
'VBA700
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.RightComma = 4
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

RotationAngle-Eigenschaft

Beschreibung

Line, Polygon und PolyLine

Legt den Rotationswinkel von folgenden Objekten in Grad fest oder gibt ihn zurück: Line, Polygon, PolyLine.

Das Objekt wird nur in Runtime (ausgehend von der projektierten Ausgangsstellung) im Uhrzeigersinn um den eingegebenen Wert um den Referenzpunkt gedreht dargestellt.

T-Stück

Legt die Orientierung eines T-Stücks in Grad fest oder gibt sie zurück. Das Attribut kann nur vier Werte annehmen:

0	Die Standardlage des T-Stücks in der Gestalt des Buchstabens "T"
90	Das "Bein" des "T"s weist nach links
180	Das "Bein" des "T"s weist nach oben
270	Das "Bein" des "T"s weist nach rechts

Andere Werte werden automatisch zum Modul 360 umgerechnet und auf den nächstliegenden zulässigen Wert auf- oder abgerundet.

Das T-Stück wird im Projekt und in Runtime um den Mittelpunkt gedreht dargestellt.

Beispiel

Die Prozedur "PolyLineConfiguration()" greift auf die Eigenschaften des Objektes PolyLine zu. In diesem Beispiel wird das Objekt in Runtime um 45° gedreht:

```
Sub PolyLineConfiguration()
'VBA701
Dim objPolyLine As HMIPolyLine
Set objPolyLine = ActiveDocument.HMIObjects.AddHMIObject("PolyLine1", "HMIPolyLine")
With objPolyLine
.ReferenceRotationLeft = 50
.ReferenceRotationTop = 50
.RotationAngle = 45
End With
End Sub
```

Siehe auch

ReferenceRotationTop-Eigenschaft (Seite 2348)

ReferenceRotationLeft-Eigenschaft (Seite 2347)

PolyLine-Objekt (Seite 2001)

Polygon-Objekt (Seite 1998)

Line-Objekt (Seite 1970)

RoundCornerHeight-Eigenschaft

Beschreibung

Legt den Eckradius des Objektes RoundRectangle fest oder gibt ihn zurück.

Geben Sie den Wert prozentual zur halben Höhe des Objektes ein.

Beispiel

Die Prozedur "RoundRectangleConfiguration()" greift auf die Eigenschaften des Objektes RoundRectangle zu. In diesem Beispiel wird der Eckradius auf 25% (Höhe) und 50% (Breite) gesetzt:

```
Sub RoundRectangleConfiguration()
'VBA702
Dim objRoundRectangle As HMIRoundRectangle
Set objRoundRectangle = ActiveDocument.HMIObjects.AddHMIObject("RoundRectangle1",
"HMIRoundRectangle")
With objRoundRectangle
.RoundCornerHeight = 25
.RoundCornerWidth = 50
End With
End Sub
```

Siehe auch

RoundCornerWidth-Eigenschaft (Seite 2352)

RoundRectangle-Objekt (Seite 2018)

RoundCornerWidth-Eigenschaft

Beschreibung

Legt den Eckradius des Objektes RoundRectangle fest oder gibt ihn zurück.

Geben Sie den Wert prozentual zur halben Breite des Objektes ein.

Beispiel

Die Prozedur "RoundRectangleConfiguration()" greift auf die Eigenschaften des Objektes RoundRectangle zu. In diesem Beispiel wird der Eckradius auf 25% (Höhe) und 50% (Breite) gesetzt:

```
Sub RoundRectangleConfiguration()
'VBA703
Dim objRoundRectangle As HMIRoundRectangle
Set objRoundRectangle = ActiveDocument.HMIObjects.AddHMIObject("RoundRectangle1",
"HMIRoundRectangle")
With objRoundRectangle
.RoundCornerHeight = 25
.RoundCornerWidth = 50
End With
End Sub
```

Siehe auch

RoundCornerHeight-Eigenschaft (Seite 2352)

RoundRectangle-Objekt (Seite 2018)

S

SameSize-Eigenschaft

Beschreibung

TRUE, wenn alle vier Buttons beim Objekt GroupDisplay die gleiche Größe haben. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel sollen alle vier Buttons die gleiche Größe haben:

```
Sub GroupDisplayConfiguration()
'VBA704
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.SameSize = True
End With
End Sub
```

Siehe auch

GroupDisplay-Objekt (Seite 1947)

ScaleColor-Eigenschaft

Beschreibung

Legt die Farbe der Skala fest oder gibt sie zurück. LONG Schreib Lese-Zugriff.

Damit die Farbe angezeigt wird, muss die Eigenschaft "Scaling" auf "True" gesetzt sein.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird zusätzlich die Skala angezeigt und die Skalenfarbe auf "Rot" gesetzt:

```
Sub BarGraphConfiguration()
'VBA705
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.Scaling = True
.ScaleColor = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

Scaling-Eigenschaft (Seite 2355)

BarGraph-Objekt (Seite 1893)

ScaleTicks-Eigenschaft

Beschreibung

Legt für das BarGraph-Objekt die Anzahl der Skalenabschnitte fest oder gibt sie zurück.

Der Skalenabschnitt ist der Bereich, der von zwei langen Skalenstrichen begrenzt wird. Wenn Sie der Eigenschaft den Wert "0" zuweisen, wird die geeignete Skaleneinteilung automatisch berechnet.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird die Anzahl der Skalenabschnitte auf "10" gesetzt:

```
Sub BarGraphConfiguration()
'VBA706
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.Scaling = True
.ScaleTicks = 10
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

Scaling-Eigenschaft

Beschreibung

TRUE, wenn beim Objekt BarGraph zusätzlich eine Skala zur Darstellung der Werte verwendet wird. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird zusätzlich die Skala angezeigt und die Skalenfarbe auf "Rot" gesetzt:

```
Sub BarGraphConfiguration()
'VBA707
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.Scaling = True
.ScaleColor = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

ScalingMode-Eigenschaft

Beschreibung

Legt fest, in welcher Größe die Objekte der Faceplate-Instanz dargestellt werden.

Default	Wie Skalierungsmodus "proportional"
1 : 1	Der Faceplate-Typ wird in der Faceplate-Instanz in originaler Größe dargestellt. Wenn die Faceplate-Instanz zu klein ist, passt sich die Größe der Faceplate-Instanz der Größe des Faceplate-Typs an.
Proportional	Der Faceplate-Typ wird proportional auf die Größe der Faceplate-Instanz skaliert.

Beispiel

ScalingType-Eigenschaft

Beschreibung

Legt die Art der Balkenskalierung fest oder gibt sie zurück. Wertebereich von 0 bis 2.

Damit die Farbe angezeigt wird, muss die Eigenschaft "Scaling" auf "True" gesetzt sein.

Balkenskalierung	zugeordneter Wert
Linear	0
Logarithmisch	1
Automatisch	2

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird die Balkenskalierung auf "Linear" gesetzt:

```
Sub BarGraphConfiguration()
'VBA708
```

```
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.ScalingType = 0
.Scaling = True
End With
End Sub
```

Siehe auch

Scaling-Eigenschaft (Seite 2355)

BarGraph-Objekt (Seite 1893)

ScriptType-Eigenschaft

Beschreibung

Gibt den Skripttyp (C oder VBS) zurück mit dem eine Eigenschaft oder ein Ereignis dynamisiert wurde. Lese-Zugriff.

Beispiel

Im folgenden Beispiel werden in das aktive Bild ein Button und ein Kreis eingefügt. In Runtime soll bei jedem Klick auf den Button der Radius des Kreises vergrößert werden. In diesem Fall wird der Skripttyp ausgegeben:

```
Sub ExampleForPrototype()
'VBA709
Dim objButton As HMIButton
Dim objCircleA As HMICircle
Dim objEvent As HMIEvent
Dim objVBScript As HMIScriptInfo
Dim strScriptType As String
Set objCircleA = ActiveDocument.HMIObjects.AddHMIObject("CircleA", "HMICircle")
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objCircleA
.Top = 100
.Left = 100
End With
With objButton
.Top = 10
.Left = 10
.Width = 200
.Text = "Increase Radius"
End With
'On every mouseclick the radius have to increase:
Set objEvent = objButton.Events(1)
Set objVBScript = objButton.Events(1).Actions.AddAction(hmiActionCreationTypeVBScript)
Select Case objVBScript.ScriptType
```

```
Case 0
strScriptType = "VB-Skript is used"
Case 1
strScriptType = "C-Skript is used"
End Select
MsgBox strScriptType
End Sub
```

Siehe auch

ScriptInfo-Objekt (Seite 2021)

ScrollBars-Eigenschaft

Beschreibung

TRUE, wenn das Bildfenster in Runtime Bildlaufleisten hat. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "PictureWindowConfig" greift auf die Eigenschaften des Bildfensters zu. In diesem Beispiel wird das Bildfenster konfiguriert:

```
Sub PictureWindowConfig()
'VBA710
Dim objPicWindow As HMIPictureWindow
Set objPicWindow = ActiveDocument.HMIObjects.AddHMIObject("PicWindow1", "HMIPictureWindow")
With objPicWindow
.AdaptPicture = False
.AdaptSize = False
.Caption = True
.CaptionText = "Picturewindow in runtime"
.OffsetLeft = 5
.OffsetTop = 10
'Replace the picturename "Test.PDL" with the name of
'an existing document from your "GraCS"-Folder of your active project
.PictureName = "Test.PDL"
.ScrollBars = True
.ServerPrefix = ""
.TagPrefix = "Struct."
.UpdateCycle = 5
.Zoom = 100
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

ScrollPositionX-Eigenschaft

Beschreibung

Legt die horizontale Verschiebung der Bildlaufleiste in einem Bildfenster mit Rollbalken fest oder gibt den Wert zurück.

Beispiel

Die Prozedur "PictureWindowConfig" greift auf die Eigenschaften des Bildfensters zu. In diesem Beispiel wird das Bildfenster konfiguriert:

```
Sub PictureWindowConfig()
'VBA808
Dim objPicWindow As HMIPictureWindow
Set objPicWindow = ActiveDocument.HMIObjects.AddHMIObject("PicWindow1", "HMIPictureWindow")
With objPicWindow
.AdaptPicture = False
.AdaptSize = False
.Caption = True
.CaptionText = "Picturewindow in runtime"
.OffsetLeft = 5
.OffsetTop = 10
'Replace the picturename "Test.PDL" with the name of
'an existing document from your "GraCS"-Folder of your active project
.PictureName = "Test.PDL"
.ScrollBars = True
.ScrollPositionX = 50
.ScrollPositionY = 50
.ServerPrefix = ""
.TagPrefix = "Struct."
.UpdateCycle = 5
.Zoom = 100
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

ScrollPositionY-Eigenschaft

Beschreibung

Legt die vertikale Verschiebung der Bildlaufleiste in einem Bildfenster mit Rollbalken fest oder gibt den Wert zurück.

Beispiel

Die Prozedur "PictureWindowConfig" greift auf die Eigenschaften des Bildfensters zu. In diesem Beispiel wird das Bildfenster konfiguriert:

```
Sub PictureWindowConfig()
'VBA809
Dim objPicWindow As HMIPictureWindow
Set objPicWindow = ActiveDocument.HMIObjects.AddHMIObject("PicWindow1", "HMIPictureWindow")
With objPicWindow
.AdaptPicture = False
.AdaptSize = False
.Caption = True
.CaptionText = "Picturewindow in runtime"
.OffsetLeft = 5
.OffsetTop = 10
'Replace the picturename "Test.PDL" with the name of
'an existing document from your "GraCS"-Folder of your active project
.PictureName = "Test.PDL"
.ScrollBars = True
.ScrollPositionX = 50
.ScrollPositionY = 50
.ServerPrefix = ""
.TagPrefix = "Struct."
.UpdateCycle = 5
.Zoom = 100
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

ScrollPosX-Eigenschaft

Beschreibung

Legt beim Objekt View die x-Position der Bildlaufleisten fest oder gibt sie zurück.

Beispiel

Im folgenden Beispiel vom aktiven Bild eine Kopie erzeugt und diese aktiviert. Die Position der Bildlaufleisten wird auf 40 (x) und 10 (y) gesetzt:

```
Sub CreateViewAndActivateView()
Dim objView As HMIView
Set objView = ActiveDocument.Views.Add
objView.Activate
objView.ScrollPosX = 40
```

```
objView.ScrollPosY = 10
End Sub
```

Siehe auch

ScrollPosY-Eigenschaft (Seite 2361)

View-Objekt (Seite 2062)

ScrollPosY-Eigenschaft

Beschreibung

Legt beim Objekt View die y-Position der Bildlaufleisten fest oder gibt sie zurück.

Beispiel

Im folgenden Beispiel vom aktiven Bild eine Kopie erzeugt und diese aktiviert. Die Position der Bildlaufleisten wird auf 40 (x) und 10 (y) gesetzt:

```
Sub CreateViewAndActivateView()
Dim objView As HMIView
Set objView = ActiveDocument.Views.Add
objView.Activate
objView.ScrollPosX = 40
objView.ScrollPosY = 10
End Sub
```

Siehe auch

ScrollPosX-Eigenschaft (Seite 2360)

View-Objekt (Seite 2062)

SelBGColor-Eigenschaft

Beschreibung

Legt die Hintergrundfarbe für den ausgewählten Eintrag beim Objekt TextList fest oder gibt ihn zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "TextListConfiguration()" greift auf die Eigenschaften des Objektes TextList zu. In diesem Beispiel wird die Hintergrundfarbe des ausgwählten Eintrages auf "Rot" gesetzt:

```
Sub TextListConfiguration()
Dim objTextList As HMITextList
'
'Neue TextListe ins aktuelle Bild einfügen:
Set objTextList = ActiveDocument.HMIObjects.AddHMIObject("myTextList", "HMITextList")
With objTextList
.SelBGColor = RGB (255, 0, 0)
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

Selected-Eigenschaft

Beschreibung

TRUE, wenn ein Objekt im Bild ausgewählt ist. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Im folgenden Beispiel werden zwei neue Objekte in das aktive Bild eingefügt und anschließend ausgewählt:

```
Sub SelectObjects()
'VBA714
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objGroup As HMIGroup
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
With objCircle
.Top = 40
.Left = 40
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 80
.Selected = True
End With
MsgBox "Objects selected!"
End Sub
```

Siehe auch

HMIObject-Objekt (Seite 1955)

Selection-Eigenschaft

Beschreibung

Gibt eine Auflistung zurück, die alle ausgewählten Objekte im angegebenen Bild enthält.

Um ein Element aus der Selection-Auflistung zurückzugeben, können Sie entweder die Indexnummer oder den Objektnamen verwenden.

Sie können die Selection-Eigenschaft z.B. verwenden, um alle Objekt im Bild auszuwählen.

Beispiel

Im folgenden Beispiel werden alle Objekte im aktiven Bild ausgewählt:

```
Sub SelectAllObjectsInActiveDocument()
'VBA715
ActiveDocument.Selection.SelectAll
End Sub
```

Siehe auch

Selection-Objekt (Auflistung) (Seite 2022)

Document-Objekt (Seite 1920)

SelIndex-Eigenschaft

Beschreibung

Legt den Index fest oder gibt ihn zurück, dessen zugehöriger Text im Kombinationsfeld oder Listenfeld hervorgehoben angezeigt wird.

SelText-Eigenschaft

Beschreibung

Zeigt den mit der Eigenschaft SelIndex festgelegten Text, der im Objekt Kombinationsfeld oder Listenfeld hervorgehoben dargestellt wird. Sie können das Attribut "Selektierter Text" nicht direkt ändern. Sie ändern das Attribut "Selektierter Text" nur indirekt, indem Sie das Attribut "Selektiertes Feld" ändern oder den Text selbst in der Eigenschaftsgruppe "Schrift" ändern.

SelTextColor-Eigenschaft

Beschreibung

Legt die Textfarbe des ausgewählten Eintrages des Objektes TextList fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "TextListConfiguration()" greift auf die Eigenschaften des Objektes TextList zu. In diesem Beispiel wird die Textfarbe des ausgwählten Eintrages auf "Gelb" gesetzt:

```
Sub TextListConfiguration()
'VBA716
Dim objTextList As HMITextList
Set objTextList = ActiveDocument.HMIObjects.AddHMIObject("myTextList", "HMITextList")
With objTextList
.SelTextColor = RGB(255, 255, 0)
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

ServerName-Eigenschaft

Beschreibung

Gibt die Bezeichnung des angegbenen ActiveX Control zurück. Lese-Zugriff.

Beispiel

Im folgenden Beispiel wird das ActiveX Control "WinCC Gauge Control" in das aktive Bild eingefügt und der Name des ActiveX Control ausgegeben:

```
Sub AddActiveXControl()
'VBA717
Dim objActiveXControl As HMIActiveXControl
```

```
Set objActiveXControl = ActiveDocument.HMIObjects.AddActiveXControl("WinCC_Gauge",
"XGAUGE.XGaugeCtrl.1")
With objActiveXControl
.Top = 40
.Left = 60
MsgBox .Properties("ServerName").value
End With
End Sub
```

Siehe auch

ActiveXControl-Objekt (Seite 1885)

ServerPrefix-Eigenschaft

Beschreibung

Legt fest, auf welchen Server das Bild liegt, das in Runtime im Bildfenster angezeigt wrid, oder gibt den Servernamen zurück.

Geben Sie den Servernamen gefolgt von zwei Doppelpunkten an: "<Servername>::". Es wird nicht überpüft, ob der Server tatsächlich vorhanden ist.

Beispiel

Die Prozedur "PictureWindowConfig" greift auf die Eigenschaften des Bildfensters zu. In diesem Beispiel wird das Bildfenster konfiguriert:

```
Sub PictureWindowConfig()
'VBA718
Dim objPicWindow As HMIPictureWindow
Set objPicWindow = ActiveDocument.HMIObjects.AddHMIObject("PicWindow1", "HMIPictureWindow")
With objPicWindow
.AdaptPicture = False
.AdaptSize = False
.Caption = True
.CaptionText = "Picturewindow in runtime"
.OffsetLeft = 5
.OffsetTop = 10
'Replace the picturename "Test.PDL" with the name of
'an existing document from your "GraCS"-Folder of your active project
.PictureName = "Test.PDL"
.ScrollBars = True
.ServerPrefix = "my_Server::"
.TagPrefix = "Struct."
.UpdateCycle = 5
.Zoom = 100
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

ShortCut-Eigenschaft

Beschreibung

Legt eine Tastaturkombination für einen benutzerdefinierten Menüeintrag oder ein benutzerdefiniertes Symbol fest oder gibt sie zurück.

Folgende Tasten sind in Kombination mit <Strg>, <Alt> und <Shift> erlaubt:

- Funktionstasten <F1> bis <F12>
- Buchstaben- und Zifferntasten von <A> bis <Z> und <0> bis <9>.

Nicht unterstützt werden die Tasten des alphanumerischen Tastenblockes, alle Cursor- (z.B. <Bild auf>) sowie übrigen Funktionstasten wie <RETURN> und <ESC>. Es wird nicht zwischen Groß- und Kleinschreibung unterschieden. Tastenkombinationen mit zwei oder mehr Buchstaben oder Ziffern sind nicht zulässig, z.B. "STRG+A+B", jedoch die Kombination mit zwei Zusatztasten, z.B. "STRG+ALT+A".

Hinweise zur Verwendung der ShortCut-Eigenschaft

Die verwendeten Tastaturkombinationen müssen innerhalb der benutzerdefinierten Menüs und Symbolleisten eines Bildes eindeutig sein. Tastaturkombinationen, die Sie mit VBA projektieren, haben Vorrang vor eventuell im Graphics Designer vorhandenen Tastaturkombinationen. Innerhalb der benutzerdefinierten Menüs und Symbolleisten haben die bildspezifischen Tastaturkombinationen Vorrang vor den anwendungsspezifischen Tastaturkombinationen.

Hinweis

Tastaturkombinationen werden nur dann ausgeführt, wenn der Menüeintrag oder das Symbol sichtbar und aktiv ist.

Beispiel

Im folgenden Beispiel wird im aktiven Bild ein benutzerdefiniertes Menü mit zwei Menüeinträgen und einem Untermenü mit zwei Einträgen angelegt. Das Untermenü wird durch eine Trennlinie optisch abgesetzt. Der erste Menüeintrag erhält Tastaturkombination <Strg +Shift+M> zum Aufrufen:

```
Sub CreateDocumentMenus()
'VBA719
Dim objDocMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Dim objSubMenu As HMIMenuItem
'
```

```
'Add menu to menubar:
Set objDocMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DocMenu1", "Doc_Menu_1")
'
'Add menuitems to the new menu:
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(1, "dmItem1_1", "&My first Menuitem")
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(2, "dmItem1_2", "My second Menuitem")
'
'Add seperator to menu:
Set objMenuItem = objDocMenu.MenuItems.InsertSeparator(3, "dSeparator1_3")
'
'Add submenu to the menu:
Set objSubMenu = objDocMenu.MenuItems.InsertSubMenu(4, "dSubMenu1_4", "My first submenu")
'
'Add menuitems to the submenu:
Set objMenuItem = objSubMenu.SubMenu.InsertMenuItem(5, "dmItem1_5", "My first submenuitem")
Set objMenuItem = objSubMenu.SubMenu.InsertMenuItem(6, "dmItem1_6", "My second
submenuitem")
'
ActiveDocument.CustomMenus("DocMenu1").MenuItems(1).ShortCut = "STRG+SHIFT+M"
End Sub
```

Siehe auch

Menüs und Symbolleisten konfigurieren (Seite 1628)

ToolbarItem-Objekt (Seite 2043)

MenuItem-Objekt (Seite 1979)

SignificantMask-Eigenschaft

Beschreibung

Wird in Runtime zur Darstellung der aktiven Meldeklasse mit der höchsten Priorität des Objektes GroupDisplay benötigt.

Der Wert der SignificantMask-Eigenschaft stellt einen systeminternen Ausgabewert dar und erfordert keine spezifische Projektierung durch den Anwender. Die Aktualisierung erfolgt in Runtime duch Anklicken des Objektes.

Beispiel

--

Siehe auch

GroupDisplay-Objekt (Seite 1947)

Size-Eigenschaft

Beschreibung

Legt die Schriftgröße in Punkt einer sprachabhängigen Schriftart fest oder gibt sie zurück.

Beispiel

Im folgenden Beispiel werden für einen Button die Schriftattribute für Französisch und Englisch gesetzt:

```
Sub ExampleForLanguageFonts()
'VBA721
Dim colLangFonts As HMILanguageFonts
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
objButton.Text = "DefText"
Set colLangFonts = objButton.LDFonts
'
'Set font-properties for french:
With colLangFonts.ItemByLCID(1036)
.Family = "Courier New"
.Bold = True
.Italic = False
.Underlined = True
.Size = 12
End With
'
'Set font-properties for english:
With colLangFonts.ItemByLCID(1033)
.Family = "Times New Roman"
.Bold = False
.Italic = True
.Underlined = False
.Size = 14
End With
End Sub
```

Siehe auch

Underlined-Eigenschaft (Seite 2402)

Parent-Eigenschaft (Seite 2317)

LanguageID-Eigenschaft (Seite 2232)

Italic-Eigenschaft (Seite 2224)

FontFamily-Eigenschaft (Seite 2201)

Bold-Eigenschaft (Seite 2106)

Application-Eigenschaft (Seite 2079)

LanguageFont-Objekt (Seite 1962)

Sizeable-Eigenschaft

Beschreibung

TRUE, wenn in Runtime die Größe der Objekte ApplicationWindow und PictureWindow verändert werden kann. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "ApplicationWindowConfig" greift auf die Eigenschaften des Applikationsfensters zu. In diesem Beispiel soll die Größe des Applikationsfensters in Runtime verändert werden kann:

```
Sub ApplicationWindowConfig()
'VBA722
Dim objAppWindow As HMIApplicationWindow
Set objAppWindow = ActiveDocument.HMIObjects.AddHMIObject("AppWindow1",
"HMIApplicationWindow")
With objAppWindow
.Sizeable = True
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

ApplicationWindow-Objekt (Seite 1891)

SmallChange-Eigenschaft

Beschreibung

Legt fest, um wieviel Schritte der Regler mit einem Mausklick verschoben werden kann oder gibt diesen Wert zurück.

Beispiel

Die Prozedur "SliderConfiguration()" greift auf die Eigenschaften des Sliders zu. In diesem Beispiel wird die Anzahl der Schritte auf "4" eingestellt:

```
Sub SliderConfiguration()
'VBA723
Dim objSlider As HMISlider
Set objSlider = ActiveDocument.HMIObjects.AddHMIObject("SliderObject1", "HMISlider")
With objSlider
.SmallChange = 4
```

```
End With
End Sub
```

Siehe auch

Slider-Objekt (Seite 2025)

SnapToGrid-Eigenschaft

Beschreibung

TRUE, wenn Objekte im Bild am (unsichtbaren) Raster ausgerichtet werden. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Im folgenden Beispiel wird die Ausrichtung von Objekten im aktiven Bild am Raster aktiviert:

```
Sub ActivateSnapToGrid()
'VBA724
ActiveDocument.SnapToGrid = True
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

SourceLink-Eigenschaft

Beschreibung

Gibt das Source-Objekt zurück. Verwenden Sie die SourceLink-Eigenschaft, um bei einer Direktverbindung das Quellobjekt zu konfigurieren.

Beispiel

Im folgenden Beispiel wird in Runtime bei einem Mausklick auf den Button die x-Position von "Rectangle_A" in die y-Position von "Rectangle_B" kopiert:

```
Sub DirectConnection()
'VBA725
Dim objButton As HMIButton
Dim objRectangleA As HMIRectangle
```

```
Dim objRectangleB As HMIRectangle
Dim objEvent As HMIEvent
Dim objDirConnection As HMIDirectConnection
Set objRectangleA = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_A", "HMIRectangle")
Set objRectangleB = ActiveDocument.HMIObjects.AddHMIObject("Rectangle_B", "HMIRectangle")
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objRectangleA
.Top = 100
.Left = 100
End With
With objRectangleB
.Top = 250
.Left = 400
.BackColor = RGB(255, 0, 0)
End With
With objButton
.Top = 10
.Left = 10
.Width = 100
.Text = "SetPosition"
End With
'
'Directconnection is initiated by mouseclick:
Set objDirConnection =
objButton.Events(1).Actions.AddAction(hmiActionCreationTypeDirectConnection)
With objDirConnection
'Sourceobject: Property "Top" of Rectangle_A
.SourceLink.Type = hmiSourceTypeProperty
.SourceLink.ObjectName = "Rectangle_A"
.SourceLink.AutomationName = "Top"
'
'Targetobject: Property "Left" of Rectangle_B
.DestinationLink.Type = hmiDestTypeProperty
.DestinationLink.ObjectName = "Rectangle_B"
.DestinationLink.AutomationName = "Left"
End With
End Sub
```

Siehe auch

Type-Eigenschaft (Seite 2392)

ObjectName-Eigenschaft (Seite 2307)

AutomationName-Eigenschaft (Seite 2082)

SourceLink-Objekt (Seite 2028)

DirectConnection-Objekt (Seite 1918)

SourceCode-Eigenschaft

Beschreibung

Legt den Quellcode eines C- oder VB-Skriptes fest oder gibt ihn zurück.

Wenn Sie der SourceCode-Eigenschaft ein C-Skript zuweisen, müssen Sie nur den Programmcode eingeben, der zwischen den geschweiften Klammern ("{}")steht.

Wenn Sie der SourceCode-Eigenschaft ein VB-Skript zuweisen, müssen Sie nur den Programmcode eingeben, der zwischen den Sub- und EndSub-Schlüsselwörtern steht.

Hinweis

Wenn Sie im Programmcode Hochkommata (') oder Anführungszeichen (") verwenden, müssen Sie vor jedes Hochkomma oder Anführungszeichen ein zusätzliches Anführungszeichen setzen, damit der Programmcode im VBA-Editor korrekt interpretiert werden kann.

Die Compiled-Eigenschaft liefert TRUE zurück, wenn der Quellcode erfolgreich kompiliert werden konnte.

Beispiel

Im folgenden Beispiel werden in das aktive Bild ein Button und ein Kreis eingefügt. In Runtime soll bei jedem Klick auf den Button der Radius des Kreises vergrößert werden. Dazu wird ein VB-Skript verwendet:

```
Sub IncreaseCircleRadiusWithVBScript()
'VBA726
Dim objButton As HMIButton
Dim objCircleA As HMICircle
Dim objEvent As HMIEvent
Dim objVBScript As HMIScriptInfo
Dim strCode As String
strCode = "Dim objCircle" & vbCrLf & "Set objCircle = "
strCode = strCode & "hmiRuntime.ActiveScreen.ScreenItems(""CircleVB"")"
strCode = strCode & vbCrLf & "objCircle.Radius = objCircle.Radius + 5"
Set objCircleA = ActiveDocument.HMIObjects.AddHMIObject("CircleVB", "HMICircle")
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
With objCircleA
.Top = 100
.Left = 100
End With
With objButton
.Top = 10
.Left = 10
.Width = 200
.Text = "Increase Radius"
End With
'
'On every mouseclick the radius have to increase:
```

```
Set objEvent = objButton.Events(1)
Set objVBScript = objButton.Events(1).Actions.AddAction(hmiActionCreationTypeVBScript)
objVBScript.SourceCode = strCode
Select Case objVBScript.Compiled
Case True
MsgBox "Compilation ok!"
Case False
MsgBox "Error on compilation!"
End Select
End Sub
```

Siehe auch

Compiled-Eigenschaft (Seite 2152)

ScriptInfo-Objekt (Seite 2021)

StartAngle-Eigenschaft

Beschreibung

Legt bei den Objekten CircularArc, EllipseArc, EllipseSegment und PieSegment den Beginn des Objektes fest oder gibt ihn zurück. Die Angabe erfolgt im Uhrzeigersinn in Grad, beginnend bei 12:00 Uhr.

Beispiel

Die Prozedur "PieSegmentConfiguration()" greift auf die Eigenschaften des Kreissegmentes zu. In diesem Beispiel beginnt das Kreissegmentes bei 40° und endet bei 180°:

```
Sub PieSegmentConfiguration()
'VBA727
Dim PieSegment As HMIPieSegment
Set PieSegment = ActiveDocument.HMIObjects.AddHMIObject("PieSegment1", "HMIPieSegment")
With PieSegment
.StartAngle = 40
.EndAngle = 180
End With
End Sub
```

Siehe auch

EndAngle-Eigenschaft (Seite 2171)

PieSegment-Objekt (Seite 1995)

EllipseSegment-Objekt (Seite 1932)

EllipseArc-Objekt (Seite 1929)

CircularArc-Objekt (Seite 1905)

StatusText-Eigenschaft

Beschreibung

Legt den Text fest, der in der Statuszeile angezeigt wird, wenn Sie mit der Maus über einen benutzerdefinierten Menüeintrag oder benutzerdefiniertes Symbol zeigen, oder gibt ihn zuück.

Beispiel

Im folgenden Beispiel wird im aktiven Bild ein benutzerdefiniertes Menü mit zwei Menüeinträgen und einem Untermenü mit zwei Einträgen angelegt. Das Untermenü wird durch eine Trennlinie optisch abgesetzt. Für jeden Menüeintrag ein Statuszeileneintrag festgelegt:

```
Sub CreateDocumentMenus()
'VBA728
Dim objDocMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Dim objSubMenu As HMIMenuItem
'
'Add menu:
Set objDocMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DocMenu1", "Doc_Menu_1")
'
'Add menuitems to custom-menu:
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(1, "dmItem1_1", "My first menuitem")
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(2, "dmItem1_2", "My second menuitem")
'
'Add seperator to custom-menu:
Set objMenuItem = objDocMenu.MenuItems.InsertSeparator(3, "dSeparator1_3")
'
'Add submenu to custom-menu:
Set objSubMenu = objDocMenu.MenuItems.InsertSubMenu(4, "dSubMenu1_4", "My first submenu")
'
'Add menuitems to submenu:
Set objMenuItem = objSubMenu.SubMenu.InsertMenuItem(5, "dmItem1_5", "My first submenuitem")
Set objMenuItem = objSubMenu.SubMenu.InsertMenuItem(6, "dmItem1_6", "My second
submenuitem")
'
'Assign statustexts to every menuitem
With objDocMenu
.MenuItems(1).StatusText = "My first menuitem"
.MenuItems(2).StatusText = "My second menuitem"
```

```
.MenuItems(4).SubMenu.Item(1).StatusText = "My first submenuitem"
.MenuItems(4).SubMenu.Item(2).StatusText = "My second submenuitem"
End With
End Sub
```

Siehe auch

ToolbarItem-Objekt (Seite 2043)

MenuItem-Objekt (Seite 1979)

SubMenu-Eigenschaft

Beschreibung

Gibt eine MenuItems-Auflistung zurück, wenn das angegebene Objekt vom Typ "SubMenu" ist.

Verwenden Sie SubMenu-Auflistung, wenn Sie in einem benutzerdefinierten Menü ein Untermenü anlegen wollen.

Beispiel

Im folgenden Beispiel wird im aktiven Bild ein benutzerdefiniertes Menü mit zwei Menüeinträgen und einem Untermenü mit zwei Einträgen angelegt. Das Untermenü wird durch eine Trennlinie optisch abgesetzt:

```
Sub CreateDocumentMenus()
'VBA730
Dim objDocMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Dim objSubMenu As HMIMenuItem
'
'Add menu:
Set objDocMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DocMenu1", "Doc_Menu_1")
'
'Add menuitems to custom-menu:
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(1, "dmItem1_1", "My first menuitem")
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(2, "dmItem1_2", "My second menuitem")
'
'Add seperator to custom-menu:
Set objMenuItem = objDocMenu.MenuItems.InsertSeparator(3, "dSeparator1_3")
'
'Add submenu to custom-menu:
Set objSubMenu = objDocMenu.MenuItems.InsertSubMenu(4, "dSubMenu1_4", "My first submenu")
'
'Add menuitems to submenu:
Set objMenuItem = objSubMenu.SubMenu.InsertMenuItem(5, "dmItem1_5", "My first submenuitem")
Set objMenuItem = objSubMenu.SubMenu.InsertMenuItem(6, "dmItem1_6", "My second
submenuitem")
```

```
End Sub
```

Siehe auch

MenuItem-Objekt (Seite 1979)

SymbolLibraries-Eigenschaft

Beschreibung

Gibt eine SymbolLibraries-Auflistung zurück, die Objekte vom Typ "SymbolLibrary" enthält.

Verwenden Sie SymbolLibraries(1), um die "Globale Bibliothek" zurückzugeben. Verwenden Sie SymbolLibraries(2), um die "Projekt Bibliothek" zurückzugeben.

Beispiel

Im folgenden Beispiel werden die Namen der Bibliotheken ausgeben:

```
Sub ShowSymbolLibraries()
'VBA731
Dim colSymbolLibraries As HMISymbolLibraries
Dim objSymbolLibrary As HMISymbolLibrary
Set colSymbolLibraries = Application.SymbolLibraries
For Each objSymbolLibrary In colSymbolLibraries
MsgBox objSymbolLibrary.Name
Next objSymbolLibrary
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

T

TabOrderSwitch-Eigenschaft

Beschreibung

Legt die Position des Objektes in der TAB-Reihenfolge fest oder gibt sie zurück.

Beispiel

In diesem Beispiel werden zwei E/A-Felder in das aktive Bild eingefügt und die TAB-Reihenfolge festgelegt:

```
Sub IOFieldConfig()
'VBA732
Dim objIOField1 As HMIIOField
Dim objIOField2 As HMIIOField
Set objIOField1 = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
Set objIOField2 = ActiveDocument.HMIObjects.AddHMIObject("IOField2", "HMIIOField")
With objIOField1
.Top = 10
.Left = 10
.TabOrderSwitch = 1
End With
With objIOField2
.Top = 100
.Left = 10
.TabOrderSwitch = 2
End With
End Sub
```

Siehe auch

HMIObject-Objekt (Seite 1955)

TabOrderAllHMIObjects-Eigenschaft

Beschreibung

TRUE, wenn alle Objekte in einem Bild in die projektierte TAB-Reihenfolge mit einbezogen werden sollen. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "ConfigureTabOrder()" legt fest, welche Objekte im aktiven Bild in die projektierte TAB-Reihenfolge mit einbezogen werden sollen. In diesem Beispiel werden alle Objekte in die TAB-Reihenfolge mit einbezogen:

```
Sub ConfigureTabOrder()
'VBA733
With ActiveDocument
.TABOrderAllHMIObjects = True
.TABOrderKeyboard = False
.TABOrderMouse = False
.TABOrderOtherAction = False
End With
End Sub
```

Siehe auch

TabOrderOtherAction-Eigenschaft (Seite 2380)

TabOrderMouse-Eigenschaft (Seite 2379)

TabOrderKeyboard-Eigenschaft (Seite 2378)

Document-Objekt (Seite 1920)

TabOrderAlpha-Eigenschaft

Beschreibung

Legt die Position des Objektes in der TAB-Reihenfolge für den Alpha-/Schalt-Cursor fest oder gibt sie zurück.

Beispiel

In diesem Beispiel werden zwei E/A-Felder in das aktive Bild eingefügt und die TAB-Reihenfolge festgelegt:

```
Sub IOFieldConfig()
'VBA734
Dim objIOField1 As HMIIOField
Dim objIOField2 As HMIIOField
Set objIOField1 = ActiveDocument.HMIObjects.AddHMIObject("IOField1", "HMIIOField")
Set objIOField2 = ActiveDocument.HMIObjects.AddHMIObject("IOField2", "HMIIOField")
With objIOField1
.Top = 10
.Left = 10
.TabOrderAlpha = 1
End With
With objIOField2
.Top = 100
.Left = 10
.TabOrderAlpha = 2
End With
End Sub
```

Siehe auch

Document-Objekt (Seite 1920)

TabOrderKeyboard-Eigenschaft

Beschreibung

TRUE, wenn Objekte in die projektierte TAB-Reihenfolge mit einbezogen werden sollen, an die ein Tastaturbedienungsereignis projektiert wurde. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "ConfigureTabOrder()" legt fest, welche Objekte im aktiven Bild in die projektierte TAB-Reihenfolge mit einbezogen werden sollen. In diesem Beispiel werden Objekte mit Tastaturbedienung in die TAB-Reihenfolge mit einbezogen:

```
Sub ConfigureTabOrder()
'VBA735
With ActiveDocument
.TABOrderAllHMIObjects = True
.TABOrderKeyboard = False
.TABOrderMouse = False
.TABOrderOtherAction = False
End With
End Sub
```

Siehe auch

TabOrderOtherAction-Eigenschaft (Seite 2380)

TabOrderMouse-Eigenschaft (Seite 2379)

TabOrderAllHMIObjects-Eigenschaft (Seite 2377)

Document-Objekt (Seite 1920)

TabOrderMouse-Eigenschaft

Beschreibung

TRUE, wenn Objekte in die projektierte TAB-Reihenfolge mit einbezogen werden sollen, an die ein Mausbedienungsereignis projektiert wurde. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "ConfigureTabOrder()" legt fest, welche Objekte im aktiven Bild in die projektierte TAB-Reihenfolge mit einbezogen werden sollen. In diesem Beispiel werden Objekte mit Mausbedienungereignis in die TAB-Reihenfolge mit einbezogen:

```
Sub ConfigureTabOrder()
'VBA736
With ActiveDocument
.TABOrderAllHMIObjects = True
.TABOrderKeyboard = False
.TABOrderMouse = False
.TABOrderOtherAction = False
End With
End Sub
```

Siehe auch

TabOrderOtherAction-Eigenschaft (Seite 2380)

TabOrderKeyboard-Eigenschaft (Seite 2378)

TabOrderAllHMIObjects-Eigenschaft (Seite 2377)

Document-Objekt (Seite 1920)

TabOrderOtherAction-Eigenschaft

Beschreibung

TRUE, wenn Objekte in die projektierte TAB-Reihenfolge mit einbezogen werden sollen, an die ein anderes Ereignis als Maus- oder Tastaturbedienung projektiert wurde. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "ConfigureTabOrder()" legt fest, welche Objekte im aktiven Bild in die projektierte TAB-Reihenfolge mit einbezogen werden sollen. In diesem Beispiel werden Objekte mit anderen Ereignissen als Maus- oder Tastaturbedienung in die TAB-Reihenfolge mit einbezogen:

```
Sub ConfigureTabOrder()
'VBA737
With ActiveDocument
.TABOrderAllHMIObjects = True
.TABOrderKeyboard = False
.TABOrderMouse = False
.TABOrderOtherAction = False
End With
End Sub
```

Siehe auch

TabOrderMouse-Eigenschaft (Seite 2379)

TabOrderKeyboard-Eigenschaft (Seite 2378)

TabOrderAllHMIObjects-Eigenschaft (Seite 2377)

Document-Objekt (Seite 1920)

Tag-Eigenschaft

Beschreibung

Legt für einen benutzerdefinierten Menüeintrag oder ein benutzerdefiniertes Symbol ein Informationstext fest oder gibt diesen zurück. Mit der Tag-Eigenschaft können Sie z.B. kurz beschreiben, was der Menüeintrag bewirkt.

Beispiel

Im folgenden Beispiel wird im aktiven Bild ein benutzerdefiniertes Menü mit zwei Menüeinträgen und einem Untermenü mit zwei Einträgen angelegt. Das Untermenü wird durch eine Trennlinie optisch abgesetzt:

```
Sub CreateDocumentMenus()
'VBA738
Dim objDocMenu As HMIMenu
Dim objMenuItem As HMIMenuItem
Dim objSubMenu As HMIMenuItem
'
'Add menu:
Set objDocMenu = ActiveDocument.CustomMenus.InsertMenu(1, "DocMenu1", "Doc_Menu_1")
'
'Add menuitems to custom-menu:
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(1, "dmItem1_1", "My first menuitem")
Set objMenuItem = objDocMenu.MenuItems.InsertMenuItem(2, "dmItem1_2", "My second menuitem")
'
'Add seperator to custom-menu:
Set objMenuItem = objDocMenu.MenuItems.InsertSeparator(3, "dSeparator1_3")
'
'Add submenu to custom-menu:
Set objSubMenu = objDocMenu.MenuItems.InsertSubMenu(4, "dSubMenu1_4", "My first submenu")
'
'Add menuitems to submenu:
Set objMenuItem = objSubMenu.SubMenu.InsertMenuItem(5, "dmItem1_5", "My first submenuitem")
Set objMenuItem = objSubMenu.SubMenu.InsertMenuItem(6, "dmItem1_6", "My second
submenuitem")
'
'To place an additional information:
With objDocMenu
.MenuItems(1).Tag = "This is the first menuitem"
End With
End Sub
```

Siehe auch

ToolbarItem-Objekt (Seite 2043)

MenuItem-Objekt (Seite 1979)

TagPrefix-Eigenschaft

Beschreibung

Legt den Variablen-Präfix fest, der allen Variablen vorangestellt wird, die im Bildfenster-Objekt enthalten sind oder gibt ihn zurück.

Beispiel:

Im Bildfenster soll das Bild "EinAusgabe" angezeigt werden. Das Bild "EinAusgabe" enthält 3 EA-Felder die an eine Strukturvariable gebunden sind. Die Strukturvariable besteht aus den Elementen EA1, EA2, EA3; jeweils ein Element für jedes EA-Feld.

Im Projekt sind z.B. drei solche Strukturvariablen definiert, mit den Strukturnamen Struct1, Struct2 und Struct3.

Der Variablenprefix ist in diesem Fall der Strukturname mit anschließendem Punkt. Geben Sie als Variablenprefix z.B. Struct2. an (der Punkt ist notwendig, um die Elemente der Strukturvariablen syntaktisch richtig als Strukturelemente anzusprechen), dann werden die EA-Felder im Bild "EinAusgabe" mit den Elementen der Strukturvariablen Struct2 verbunden:

Variablen-Prefix: "Struct2."

- Ausgabewert (erstes EA-Feld): EA1
- Ausgabewert (zweites EA-Feld): EA2
- Ausgabewert (drittes EA-Feld): EA3

Die aktuelle Variablenanbindung im Bildfenster wird damit

- Ausgabewert (erstes EA-Feld): Struct2.EA1
- Ausgabewert (zweites EA-Feld): Struct2.EA2
- Ausgabewert (drittes EA-Feld): Struct2.EA3

Beispiel

Die Prozedur "PictureWindowConfig" greift auf die Eigenschaften des Bildfensters zu. In diesem Beispiel wird das Bildfenster konfiguriert:

```
Sub PictureWindowConfig()
'VBA739
Dim objPicWindow As HMIPictureWindow
Set objPicWindow = ActiveDocument.HMIObjects.AddHMIObject("PicWindow1", "HMIPictureWindow")
With objPicWindow
.AdaptPicture = False
.AdaptSize = False
.Caption = True
.CaptionText = "Picturewindow in runtime"
.OffsetLeft = 5
.OffsetTop = 10
'Replace the picturename "Test.PDL" with the name of
'an existing document from your "GraCS"-Folder of your active project
.PictureName = "Test.PDL"
```

```
.ScrollBars = True
.ServerPrefix = "my_Server::"
.TagPrefix = "Struct."
.UpdateCycle = 5
.Zoom = 100
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

TagScaleParam1-Eigenschaft

Beschreibung

Setzt den Wert1 bei Wertebereich Prozess.

TagScaleParam2-Eigenschaft

Beschreibung

Setzt den Wert2 bei Wertebereich Prozess.

TagScaleParam3-Eigenschaft

Beschreibung

Setzt den Wert3 bei Wertebereich Prozess.

TagScaleParam4-Eigenschaft

Beschreibung

Setzt den Wert4 bei Wertebereich Prozess.

TagStartvaluePersistence-Eigenschaft

Beschreibung

Legt fest, ob eine interne Variable persistent gesetzt ist. Sie können nur interne Variablen als persistent setzen.

Text-Eigenschaft

Beschreibung

Legt die Beschriftung für ein Objekt fest oder gibt sie zurück.

Beispiel

Die Prozedur "ButtonConfiguration()" greift auf die Eigenschaften des Buttons zu. In diesem Beispiel wird die Beschriftung festgelegt:

```
Sub ButtonConfiguration()
'VBA740
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("Button1", "HMIButton")
With objButton
.Text = "Button1"
End With
End Sub
```

Siehe auch

Button-Objekt (Seite 1898)

StaticText-Objekt (Seite 2029)

OptionGroup-Objekt (Seite 1989)

CheckBox-Objekt (Seite 1901)

Toggle-Eigenschaft

Beschreibung

TRUE, wenn der Button oder Rundbutton in Runtime nach dem Betätigen einrasten soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "RoundButtonConfiguration()" greift auf die Eigenschaften des Rundbuttons zu. In diesem Beispiel soll der Rundbutton in Runtime nach dem Drücken einrasten:

```
Sub RoundButtonConfiguration()
'VBA741
Dim objRoundButton As HMIRoundButton
Set objRoundButton = ActiveDocument.HMIObjects.AddHMIObject("RButton1", "HMIRoundButton")
With objRoundButton
.Toggle = True
```

```
End With
End Sub
```

Siehe auch

RoundButton-Objekt (Seite 2015)

ToleranceHigh-Eigenschaft

Beschreibung

Legt den Grenzwert für "Toleranz high" fest oder gibt ihn zurück.

Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft TypeToleranceHigh fest.

Die Überwachung des Grenzwerts ist nur wirksam, wenn die Eigenschaft CheckToleranceHigh auf "True" gesetzt ist.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften von Grenzwerten. In diesem Beispiel wird der Grenzwert für "Toleranz high" konfiguriert:

```
Sub BarGraphLimitConfiguration()
'VBA742
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis = absolute
.TypeToleranceHigh = False
'Activate monitoring
.CheckToleranceHigh = True
'Set barcolor = "yellow"
.ColorToleranceHigh = RGB(255, 255, 0)
'Set upper limit to "40"
.ToleranceHigh = 40
End With
End Sub
```

Siehe auch

TypeToleranceHigh-Eigenschaft (Seite 2398)

CheckToleranceHigh-Eigenschaft (Seite 2132)

BarGraph-Objekt (Seite 1893)

ToleranceLow-Eigenschaft

Beschreibung

Legt den Grenzwert für "Toleranz low" fest oder gibt ihn zurück.

Die Art der Auswertung (prozentual oder absolut) legen Sie mit der Eigenschaft TypeToleranceLow fest.

Die Überwachung des Grenzwerts ist nur wirksam, wenn die Eigenschaft CheckToleranceLow auf "True" gesetzt ist.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften von Grenzwerten. In diesem Beispiel wird der Grenzwert für "Toleranz low" konfiguriert:

```
Sub BarGraphLimitConfiguration()
'VBA743
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis = absolute
.TypeToleranceLow = False
'Activate monitoring
.CheckToleranceLow = True
'Set barcolor = "red"
.ColorToleranceLow = RGB(255, 0, 0)
'Set lower limit to "40"
.ToleranceLow = 40
End With
End Sub
```

Siehe auch

TypeToleranceLow-Eigenschaft (Seite 2398)

CheckToleranceLow-Eigenschaft (Seite 2133)

BarGraph-Objekt (Seite 1893)

ToolbarItems-Eigenschaft

Beschreibung

Gibt eine Auflistung zurück, die alle Elemente (Symbole und Trennlinien) einer benutzerdefinierten Symbolleiste enthält.

Beispiel

Im folgenden Beispiel wird im aktiven Bild eine benutzerdefinierte Symbolleiste zwei Symbole angelegt, die durch eine Trennlinie getrennt sind:

```
Sub AddDocumentSpecificCustomToolbar()
'VBA744
Dim objToolbar As HMIToolbar
Dim objToolbarItem As HMIToolbarItem
Set objToolbar = ActiveDocument.CustomToolbars.Add("DocToolbar")
'
'Add symbol-icon to userdefined toolbar
Set objToolbarItem = objToolbar.ToolbarItems.InsertToolbarItem(1, "tItem1_1", "My first
symbol-icon")
Set objToolbarItem = objToolbar.ToolbarItems.InsertToolbarItem(3, "tItem1_3", "My second
symbol-icon")
Set objToolbarItem = objToolbar.ToolbarItems.InsertSeparator(2, "tSeparator1_2")
End Sub
```

Siehe auch

ToolbarItem-Objekt (Seite 2043)

Toolbar-Objekt (Seite 2040)

TooltipText-Eigenschaft

Beschreibung

Legt den Text fest, der als Tooltip angezeigt wird, wenn Sie mit der Maus über das Objekt (HMIObject, Symbol) fahren oder gibt ihn zurück.

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Objektes Rectangle zu. In diesem Beispiel wird dem Rechteck ein Tooltiptext zugewiesen:

```
Sub RectangleConfiguration()
'VBA745
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.ToolTipText = "This is a rectangle"
End With
End Sub
```

Das folgende Beispiel zeigt, wie Sie die Eigenschaft vor der Dynamisierung initialisieren müssen:

```
Sub Dyn()
'VBA823
Dim objCircle As HMICircle
Dim doc As Document
Dim objDynDialog As HMIDynamicDialog
Set doc = ActiveDocument
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle","HMICircle")
objCircle.ObjectName = "Circle1"
objCircle.BorderColor = RGB(255, 0, 0)
objCircle.BackColor = RGB(0, 255, 0)
objCircle.ToolTipText = "Text"
Set objDynDialog =
objCircle.ToolTipText.CreateDynamic(hmiDynamicCreationTypeDynamicDialog, "'Var'")
End Sub
```

Siehe auch

ToolbarItem-Objekt (Seite 2043)

HMIObject-Objekt (Seite 1955)

So dynamisieren Sie eine Eigenschaft mit dem Dynamik-Dialog (Seite 1697)

Top-Eigenschaft

Beschreibung

Legt die y-Koordinate eines Objektes (gemessen vom linken, oberen Bildrand) in Pixel fest oder gibt sie zurück. Die y-Koordinate bezieht sich auf die Ecke links oben des objektumfassenden Rechteckes.

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Objektes Rectangle zu. In diesem Beispiel wird das Rechteck auf die Position 10/40 gesetzt:

```
Sub RectangleConfiguration()
'VBA746
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
.Left = 10
.Top = 40
End With
End Sub
```

Siehe auch

View-Objekt (Seite 2062)

HMIObject-Objekt (Seite 1955)

TopConnectedObjectName-Eigenschaft

Beschreibung

Gibt den Namen des Ende-Objektes zurück, das der Verbinder verbindet. Lese-Zugriff.

Beispiel:

Ein Beispiel für die Anwendung der BottomConnectedObjectName-Eigenschaft finden Sie in dieser Dokumentation unter "objConnection-Objekt".

Siehe auch

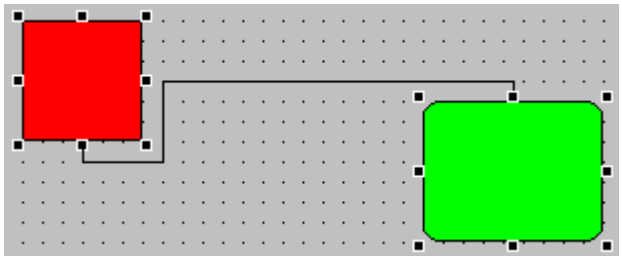
objConnection-Objekt (Seite 1985)

TopConnectedConnectionPointIndex-Eigenschaft

Beschreibung

Gibt die Anschlussstelle am Objekt zurück, mit dem der Verbinder verbunden ist.

Anschlussstelle	zugeordneter Wert
Oben	0
Rechts	1
Unten	2
Links	3



Beispiel

Ein Beispiel für die Anwendung der BottomConnectedObjectName-Eigenschaft finden Sie in dieser Dokumentation unter "objConnection-Objekt".

Siehe auch

objConnection-Objekt (Seite 1985)

Transparency-Eigenschaft

Beschreibung

Legt fest, in welchem Maße das Objekt transparent dargestellt wird. Der Wert zwischen 0 und 100 gibt die Transparenz in Prozent an. Bei einem halb transparenten Objekt scheinen andere Objekte hindurch. Ein 100% transparentes Objekt ist unsichtbar. Auch ein unsichtbares Objekt bleibt in Runtime bedienbar.

Beispiel

```
Sub addTransparentObject()
    'VBA849
    Dim objHMICircle As HMICircle
    Set objHMICircle = ActiveDocument.HMIObjects.AddHMIObject("Circle", "HMICircle")
    objHMICircle.Transparency = 40
End Sub
```

Trend-Eigenschaft

Beschreibung

TRUE, wenn wenn die Tendenz (steigend oder fallend) des zu überwachenden Messwertes mit einem kleinen Pfeil angezeigt werden soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird die Tendenz des Messwertes angezeigt:

```
Sub BarGraphConfiguration()
'VBA747
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.trend = True
End With
End Sub
```

Siehe auch

BarGraph-Objekt (Seite 1893)

TrendColor-Eigenschaft

Beschreibung

Legt die Farbe der Trendanzeige fest oder gibt sie zurück.

Die Trendanzeige stellt die Tendenz (steigend oder fallend) des zu überwachenden Messwertes mit einem kleinen Pfeil dar. Um die Trendanzeige zu aktivieren, muss die Eigenschaft Trend auf "True" gesetzt sein. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird die Tendenz des Messwertes angezeigt; die Trendanzeige wird auf "Rot" gesetzt:

```
Sub BarGraphConfiguration()
'VBA748
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.trend = True
.TrendColor = RGB(255, 0, 0)
End With
End Sub
```

Siehe auch

Trend-Eigenschaft (Seite 2390)

BarGraph-Objekt (Seite 1893)

Trigger-Eigenschaft

Beschreibung

Gibt ein Trigger-Objekt zurück. Verwenden Sie die Trigger-Eigenschaft, wenn Sie eine Eigenschaft mit einem Skript dynamisieren.

Beispiel

In diesem Beispiel wird die Eigenschaft "Radius" einen Kreises mit einem C-Skript dynamisiert (der Rückgabewert setzt den Radius):

```
Sub AddDynamicAsCSkriptToProperty()
'VBA749
Dim objVBScript As HMIScriptInfo
Dim objCircle As HMICircle

Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("myCircle", "HMICircle")
Set objVBScript = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeVBScript)
With objVBScript
.Trigger.Type = hmiTriggerTypeStandardCycle
.Trigger.CycleType = hmiCycleType_2s
.Trigger.Name = "Trigger1"
End With
End Sub
```

Siehe auch

Trigger-Objekt (Seite 2047)

ScriptInfo-Objekt (Seite 2021)

Type-Eigenschaft

Beschreibung

Gibt den Typ eines Objektes zurück oder legt ihn fest.

Der Objekttyp wird entweder als String oder Integer zurückgegeben.

Beispiel

Die Prozedur "RectangleConfiguration()" greift auf die Eigenschaften des Objektes Rectangle zu. In diesem Beispiel wird der Objekttyp ausgegeben:

```
Sub RectangleConfiguration()
'VBA750
Dim objRectangle As HMIRectangle
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("Rectangle1", "HMIRectangle")
With objRectangle
MsgBox "Objecttype: " & .Type
End With
End Sub
```

Siehe auch

Trigger-Objekt (Seite 2047)

ToolbarItem-Objekt (Seite 2043)

SourceLink-Objekt (Seite 2028)

Property-Objekt (Seite 2005)

HMIObject-Objekt (Seite 1955)

FolderItem-Objekt (Seite 1939)

DestLink-Objekt (Seite 1917)

TypeAlarmHigh-Eigenschaft

Beschreibung

TRUE, wenn der obere Grenzwert, bei dem Alarm ausgelöst wird, prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "50" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA751
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis = absolute
.TypeAlarmHigh = False
'Activate monitoring
.CheckAlarmHigh = True
'Set barcolor = "yellow"
.ColorAlarmHigh = RGB(255, 255, 0)
'Set upper limit = "50"
.AlarmHigh = 50
End With
End Sub
```

Siehe auch

ColorAlarmHigh-Eigenschaft (Seite 2138)

CheckAlarmHigh-Eigenschaft (Seite 2127)

AlarmHigh-Eigenschaft (Seite 2073)

BarGraph-Objekt (Seite 1893)

TypeAlarmLow-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert, bei dem Alarm ausgelöst wird, prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "10" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA752
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis = absolute
.TypeAlarmLow = False
'Activate monitoring
.CheckAlarmLow = True
'Set barcolor = "yellow"
.ColorAlarmLow = RGB(255, 255, 0)
'Set lower limit = "10"
.AlarmLow = 10
End With
End Sub
```

Siehe auch

ColorAlarmLow-Eigenschaft (Seite 2139)

CheckAlarmLow-Eigenschaft (Seite 2127)

AlarmLow-Eigenschaft (Seite 2074)

BarGraph-Objekt (Seite 1893)

TypeLimitHigh4-Eigenschaft

Beschreibung

TRUE, wenn der obere Grenzwert "Reserve 4" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "70" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA753
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis = absolute
.TypeLimitHigh4 = False
'Activate monitoring
.CheckLimitHigh4 = True
'Set barcolor = "red"
.ColorLimitHigh4 = RGB(255, 0, 0)
'Set upper limit = "70"
.LimitHigh4 = 70
End With
End Sub
```

Siehe auch

LimitHigh4-Eigenschaft (Seite 2269)

CheckLimitHigh4-Eigenschaft (Seite 2129)

BarGraph-Objekt (Seite 1893)

TypeLimitHigh5-Eigenschaft

Beschreibung

TRUE, wenn der obere Grenzwert "Reserve 5" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "80" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA754
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis = absolute
```

```
.TypeLimitHigh5 = False
'Activate monitoring
.CheckLimitHigh5 = True
'Set barcolor = "black"
.ColorLimitHigh5 = RGB(0, 0, 0)
'Set upper limit = "70"
.LimitHigh5 = 70
End With
End Sub
```

Siehe auch

LimitHigh5-Eigenschaft (Seite 2269)

CheckLimitHigh5-Eigenschaft (Seite 2130)

BarGraph-Objekt (Seite 1893)

TypeLimitLow4-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Reserve 4" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "5" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA755
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis = absolute
.TypeLimitLow4 = False
'Activate monitoring
.CheckLimitLow4 = True
'Set barcolor = "green"
.ColorLimitLow4 = RGB(0, 255, 0)
'Set lower limit = "5"
.LimitLow4 = 5
End With
End Sub
```

Siehe auch

LimitLow4-Eigenschaft (Seite 2270)

ColorLimitLow4-Eigenschaft (Seite 2143)

CheckLimitLow4-Eigenschaft (Seite 2131)

BarGraph-Objekt (Seite 1893)

TypeLimitLow5-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Reserve 5" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "0" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA756
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis = absolute
.TypeLimitLow5 = False
'Activate monitoring
.CheckLimitLow5 = True
'Set barcolor = "white"
.ColorLimitLow5 = RGB(255, 255, 255)
'Set lower limit = "0"
.LimitLow5 = 0
End With
End Sub
```

Siehe auch

LimitLow5-Eigenschaft (Seite 2271)

ColorLimitLow5-Eigenschaft (Seite 2144)

CheckLimitLow5-Eigenschaft (Seite 2131)

BarGraph-Objekt (Seite 1893)

TypeToleranceHigh-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Toleranz high" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften von Grenzwerten. In diesem Beispiel wird der Grenzwert für "Toleranz high" konfiguriert:

```
Sub BarGraphLimitConfiguration()
'VBA757
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis = absolute
.TypeToleranceHigh = False
'Activate monitoring
.CheckToleranceHigh = True
'Set barcolor = "yellow"
.ColorToleranceHigh = RGB(255, 255, 0)
'Set upper limit = "40"
.ToleranceHigh = 40
End With
End Sub
```

Siehe auch

ColorToleranceHigh-Eigenschaft (Seite 2145)

CheckToleranceHigh-Eigenschaft (Seite 2132)

BarGraph-Objekt (Seite 1893)

TypeToleranceLow-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Toleranz low" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften von Grenzwerten. In diesem Beispiel wird der Grenzwert für "Toleranz low" konfiguriert:

```
Sub BarGraphLimitConfiguration()
'VBA758
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis = absolute
.TypeToleranceLow = False
'Activate monitoring
.CheckToleranceLow = True
'Set barcolor = "red"
.ColorToleranceLow = RGB(255, 0, 0)
'Set lower limit = "10"
.ToleranceLow = 10
End With
End Sub
```

Siehe auch

ToleranceLow-Eigenschaft (Seite 2386)

ColorToleranceLow-Eigenschaft (Seite 2146)

CheckToleranceLow-Eigenschaft (Seite 2133)

BarGraph-Objekt (Seite 1893)

TypeWarningHigh-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Warning high" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "75" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA759
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis = absolute
```

```
.TypeWarningHigh = False
'Activate monitoring
.CheckWarningHigh = True
'Set barcolor = "red"
.ColorWarningHigh = RGB(255, 0, 0)
'Set upper limit = "75"
.WarningHigh = 75
End With
End Sub
```

Siehe auch

WarningHigh-Eigenschaft (Seite 2484)

ColorWarningHigh-Eigenschaft (Seite 2147)

CheckWarningHigh-Eigenschaft (Seite 2134)

BarGraph-Objekt (Seite 1893)

TypeWarningLow-Eigenschaft

Beschreibung

TRUE, wenn der untere Grenzwert "Warning low" prozentual ausgewertet wird. FALSE, wenn die Auswertung absolut erfolgen soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "12" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA760
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis = absolute
.TypeWarningLow = False
'Activate monitoring
.CheckWarningLow = True
'Set barcolor = "magenta"
.ColorWarningLow = RGB(255, 0, 255)
'Set lower limit = "12"
.WarningLow = 12
End With
End Sub
```

Siehe auch

WarningLow-Eigenschaft (Seite 2485)

ColorWarningLow-Eigenschaft (Seite 2148)

CheckWarningLow-Eigenschaft (Seite 2135)

BarGraph-Objekt (Seite 1893)

U

UsedLanguage-Eigenschaft

Beschreibung

Mit der UsedLanguage-Eigenschaft stellen Sie die LanguageID ein. Dadurch wird die passende Codepage für den verwendeten Zeichensatz gewählt.

Mit der UsedLanguage-Eigenschaft stellen Sie die passende Codepage für den verwendeten Zeichensatz ein.

LONG Schreib-Lese-Zugriff.

Beispiel

Im folgenden Beispiel wird mit der Eigenschaft "UsedLanguage" und der Sprach-ID "1033" die Codepage auf Englisch-US gesetzt.

```
Sub Create_Cycle_with_Dynamic()
'VBA61

Dim objCScript As HMIScriptInfo
Dim objCircle As HMICircle
Dim strCode As String

strCode = "long lHeight;" & vbCrLf & "int check;" & vbCrLf
strCode = strCode & "GetHeight( ""events.PDL"", ""myCircle"" );" & vbCrLf
strCode = strCode & "lHeight = lHeight+5;" & vbCrLf
strCode = strCode & "check = SetHeight(""events.PDL"", ""myCircle"", lHeight );"
strCode = strCode & vbCrLf & "//Return-Type: BOOL" & vbCrLf
strCode = strCode & "return check;"
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("myCircle", "HMICircle")
'Create dynamic for Property "Height":
Set objCScript = objCircle.Height.CreateDynamic(hmiDynamicCreationTypeCScript)
'
'set Sourcecode and cycletime:
With objCScript
.SourceCode = strCode
.Trigger.Type = hmiTriggerTypeStandardCycle
.Trigger.CycleType = hmiCycleType_2s
```

```
.Trigger.Name = "Trigger1"
'Set language English-US
.UsedLanguage = 1033
End With

End Sub
```

Underlined-Eigenschaft

Beschreibung

TRUE, wenn das Schriftattribut "Unterstrichen" für den sprachabhängigen Text im Objekt gesetzt ist. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Im folgenden Beispiel werden für einen Button die Schriftattribute für Französisch und Englisch gesetzt:

```
Sub ExampleForLanguageFonts()
'VBA761
Dim colLangFonts As HMILanguageFonts
Dim objButton As HMIButton
Set objButton = ActiveDocument.HMIObjects.AddHMIObject("myButton", "HMIButton")
objButton.Text = "DefText"
Set colLangFonts = objButton.LDFonts
'
'Set font-properties for french:
With colLangFonts.ItemByLCID(1036)
.Family = "Courier New"
.Bold = True
.Italic = False
.Underlined = True
.Size = 12
End With
'
'Set font-properties for english:
With colLangFonts.ItemByLCID(1033)
.Family = "Times New Roman"
.Bold = False
.Italic = True
.Underlined = False
.Size = 14
End With
End Sub
```

Siehe auch

Size-Eigenschaft (Seite 2368)

Parent-Eigenschaft (Seite 2317)

LanguageID-Eigenschaft (Seite 2232)

Italic-Eigenschaft (Seite 2224)

FontFamily-Eigenschaft (Seite 2201)

Bold-Eigenschaft (Seite 2106)

Application-Eigenschaft (Seite 2079)

LanguageFont-Objekt (Seite 1962)

UnselBGColor-Eigenschaft

Beschreibung

Legt beim Objekt TextList die Hintergrundfarbe der Auswahlliste bei nicht ausgewählten Einträgen fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "TextListConfiguration()" greift auf die Eigenschaften des Objektes TextList zu. In diesem Beispiel werden die Farben für nicht ausgewählte Einträge in der Auswahlliste festgelegt:

```
Sub TextListConfiguration()
'VBA762
Dim objTextList As HMITextList
Set objTextList = ActiveDocument.HMIObjects.AddHMIObject("myTextList", "HMITextList")
With objTextList
.UnselBGColor = RGB(255, 0, 0)
.UnselTextColor = RGB(0, 0, 0)
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

UnselTextColor-Eigenschaft

Beschreibung

Legt beim Objekt TextList die Textfarbe in der Auswahlliste bei nicht ausgewählten Einträgen fest oder gibt sie zurück. LONG Schreib-Lese-Zugriff.

Ermittlung des Farbwertes

Die Farbe wird im RGB-Format (Rot, Grün, Blau) dargestellt. Geben Sie für jeden der drei RGB-Werte den entsprechenden Dezimalwert an (Wertebereich von 0 bis 255).

Verwenden Sie VBA-Funktion "RGB", um einer Eigenschaft eine Farbe zuzuweisen. Die Farbe "Rot" wird beispielsweise so dargestellt: RGB(255, 0, 0)

Beispiel

Die Prozedur "TextListConfiguration()" greift auf die Eigenschaften des Objektes TextList zu. In diesem Beispiel werden die Farben für nicht ausgewählte Einträge in der Auswahlliste festgelegt:

```
Sub TextListConfiguration()
'VBA763
Dim objTextList As HMITextList
Set objTextList = ActiveDocument.HMIObjects.AddHMIObject("myTextList", "HMITextList")
With objTextList
.UnselBGColor = RGB(255, 0, 0)
.UnselTextColor = RGB(0, 0, 0)
End With
End Sub
```

Siehe auch

TextList-Objekt (Seite 2037)

UpdateCycle-Eigenschaft

Beschreibung

Legt die Art und Häufigkeit für die Aktualisierung des Bildfensters in Runtime fest oder gibt sie zurück.

Aktualisierungszyklus	zugeordneter Wert
bei Änderung	0
250 ms	1
500 ms	2
1 s	3
2 s	4

Aktualisierungszyklus	zugeordneter Wert
5 s	5
10 s	6
1 min	7
5 min	8
10 min	9
1 h	10
Anwenderzyklus1	11
Anwenderzyklus2	12
Anwenderzyklus3	13
Anwenderzyklus4	14
Anwenderzyklus5	15
Bildzyklus	255

Beispiel

Die Prozedur "PictureWindowConfig" greift auf die Eigenschaften des Bildfensters zu. In diesem Beispiel wird das Bildfenster in Runtime alle fünf Sekunden aktualisiert:

```
Sub PictureWindowConfig()
'VBA764
Dim objPicWindow As HMIPictureWindow
Set objPicWindow = ActiveDocument.HMIObjects.AddHMIObject("PicWindow1", "HMIPictureWindow")
With objPicWindow
.UpdateCycle = 5
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

UserValue1-Eigenschaft

Beschreibung

Legt beim Objekt GroupDisplay einen beliebigen Wert fest oder gibt ihn zurück.

Den Wert können Sie z.B. von einem Skript auswerten lassen. In Runtime wird dieser Wert weder ausgewertet noch angezeigt.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel werden vier unterschiedliche Benutzerwerte zugewiesen:

```
Sub GroupDisplayConfiguration()
'VBA765
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.UserValue1 = 0
.UserValue2 = 25
.UserValue3 = 50
.UserValue4 = 75
End With
End Sub
```

Siehe auch

UserValue4-Eigenschaft (Seite 2408)

UserValue3-Eigenschaft (Seite 2407)

UserValue2-Eigenschaft (Seite 2406)

GroupDisplay-Objekt (Seite 1947)

UserValue2-Eigenschaft

Beschreibung

Legt beim Objekt GroupDisplay einen beliebigen Wert fest oder gibt ihn zurück.

Den Wert können Sie z.B. von einem Skript auswerten lassen. In Runtime wird dieser Wert weder ausgewertet noch angezeigt.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel werden vier unterschiedliche Benutzerwerte zugewiesen:

```
Sub GroupDisplayConfiguration()
'VBA766
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.UserValue1 = 0
.UserValue2 = 25
```

```
.UserValue3 = 50
.UserValue4 = 75
End With
End Sub
```

Siehe auch

UserValue4-Eigenschaft (Seite 2408)

UserValue3-Eigenschaft (Seite 2407)

UserValue1-Eigenschaft (Seite 2405)

GroupDisplay-Objekt (Seite 1947)

UserValue3-Eigenschaft

Beschreibung

Legt beim Objekt GroupDisplay einen beliebigen Wert fest oder gibt ihn zurück.

Den Wert können Sie z.B. von einem Skript auswerten lassen. In Runtime wird dieser Wert weder ausgewertet noch angezeigt.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel werden vier unterschiedliche Benutzerwerte zugewiesen:

```
Sub GroupDisplayConfiguration()
'VBA767
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.UserValue1 = 0
.UserValue2 = 25
.UserValue3 = 50
.UserValue4 = 75
End With
End Sub
```

Siehe auch

UserValue4-Eigenschaft (Seite 2408)

UserValue2-Eigenschaft (Seite 2406)

UserValue1-Eigenschaft (Seite 2405)

GroupDisplay-Objekt (Seite 1947)

UserValue4-Eigenschaft

Beschreibung

Legt beim Objekt GroupDisplay einen beliebigen Wert fest oder gibt ihn zurück.

Den Wert können Sie z.B. von einem Skript auswerten lassen. In Runtime wird dieser Wert weder ausgewertet noch angezeigt.

Beispiel

Die Prozedur "GroupDisplayConfiguration()" greift auf die Eigenschaften der Sammelanzeige zu. In diesem Beispiel werden vier unterschiedliche Benutzerwerte zugewiesen:

```
Sub GroupDisplayConfiguration()
'VBA768
Dim objGroupDisplay As HMIGroupDisplay
Set objGroupDisplay = ActiveDocument.HMIObjects.AddHMIObject("GroupDisplay1",
"HMIGroupDisplay")
With objGroupDisplay
.UserValue1 = 0
.UserValue2 = 25
.UserValue3 = 50
.UserValue4 = 75
End With
End Sub
```

Siehe auch

UserValue3-Eigenschaft (Seite 2407)

UserValue2-Eigenschaft (Seite 2406)

UserValue1-Eigenschaft (Seite 2405)

GroupDisplay-Objekt (Seite 1947)

V

Value

Value-Eigenschaft

Beschreibung

Gibt den Wert einer Objekteigenschaft zurück oder legt ihn fest.

Beispiel

Verwenden Sie die Value-Eigenschaft, wenn Sie einen Wert über die Properties-Auflistung zurückgeben oder festlegen wollen. In diesem Beispiel wird über die Value-Eigenschaft auf die Eigenschaft eines ActiveX Controls zugegriffen:

```
Sub AddActiveXControl()
'VBA769
Dim objActiveXControl As HMIActiveXControl
Set objActiveXControl = ActiveDocument.HMIObjects.AddActiveXControl("WinCC_Gauge2",
"XGAUGE.XGaugeCtrl.1")
'
'Move ActiveX-Control:
objActiveXControl.Top = 40
objActiveXControl.Left = 60
'
'Modify individual properties:
objActiveXControl.Properties("BackColor").value = RGB(255, 0, 0)
End Sub
```

Siehe auch

Property-Objekt (Seite 2005)

VALUE_ACCESS_FAULT-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Variablenstatus "Zugriff auf Variable nicht erlaubt" eintritt, oder gibt ihn zurück.

Damit der Status ausgewertet werden kann, muss die VariableStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
```

```
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)

VariableStateChecked-Eigenschaft (Seite 2478)

VALUE_TIMEOUT-Eigenschaft (Seite 2454)

VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)

VALUE_SERVERDOWN-Eigenschaft (Seite 2451)

VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)

VALUE_MIN_RANGE-Eigenschaft (Seite 2448)

VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)

VALUE_MAX_RANGE-Eigenschaft (Seite 2445)

VALUE_INVALID_KEY-Eigenschaft (Seite 2439)

VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)

VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)

VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)

VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)

VariableStateValue-Objekt (Seite 2057)

VALUE_ADDRESS_ERROR-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Variablenstatus "Adressierungsfehler" eintritt, oder gibt ihn zurück.

Damit der Status ausgewertet werden kann, muss die VariableStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA771
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VariableStateChecked-Eigenschaft (Seite 2478)
VALUE_TIMEOUT-Eigenschaft (Seite 2454)
VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)
VALUE_SERVERDOWN-Eigenschaft (Seite 2451)
VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)
VALUE_MIN_RANGE-Eigenschaft (Seite 2448)
VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)
VALUE_MAX_RANGE-Eigenschaft (Seite 2445)
VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)
VALUE_INVALID_KEY-Eigenschaft (Seite 2439)
VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)
VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)
VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)
VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)
VariableStateValue-Objekt (Seite 2057)

VALUE_BAD_COMMLUV-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "bad, no communication (last usable value)" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA818
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
```

```
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_BAD_COMMNUV-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "bad, no communication (no usable value)" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_BAD_CONFERROR-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "bad, configuration error, value not accepted" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_BAD_DEVICE-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "bad, device failure" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_BAD_MISCSTATES-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "bad miscellanous states" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_BAD_NONSPECIFIC-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "bad, non-specific" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_BAD_NOTCONNECTED-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "bad, not connected" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_BAD_OUTOFSERV-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "bad, out of service" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_BAD_PROCRELNOM-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "bad, process related, no maintenance" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_BAD_PROCRELSUB-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "bad, process related, substitute value" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_CONVERSION_ERROR-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Variablenstatus "Wandlungsfehler" eintritt, oder gibt ihn zurück.

Damit der Status ausgewertet werden kann, muss die VariableStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA772
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VariableStateChecked-Eigenschaft (Seite 2478)
VALUE_TIMEOUT-Eigenschaft (Seite 2454)
VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)
VALUE_SERVERDOWN-Eigenschaft (Seite 2451)
VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)
VALUE_MIN_RANGE-Eigenschaft (Seite 2448)
VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)
VALUE_MAX_RANGE-Eigenschaft (Seite 2445)
VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)
VALUE_INVALID_KEY-Eigenschaft (Seite 2439)
VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)
VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)
VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)
VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)
VariableStateValue-Objekt (Seite 2057)

VALUE_HANDSHAKE_ERROR-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Variablenstatus "Protokollfehler" eintritt, oder gibt ihn zurück.

Damit der Status ausgewertet werden kann, muss die VariableStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA773
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
```

```
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VariableStateChecked-Eigenschaft (Seite 2478)
VALUE_TIMEOUT-Eigenschaft (Seite 2454)
VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)
VALUE_SERVERDOWN-Eigenschaft (Seite 2451)
VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)
VALUE_MIN_RANGE-Eigenschaft (Seite 2448)
VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)
VALUE_MAX_RANGE-Eigenschaft (Seite 2445)
VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)
VALUE_INVALID_KEY-Eigenschaft (Seite 2439)
VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)
VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)
VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)
VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)
VariableStateValue-Objekt (Seite 2057)

VALUE_HARDWARE_ERROR-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Variablenstatus "keine Netzwerkbaugruppe" eintritt, oder gibt ihn zurück.

Damit der Status ausgewertet werden kann, muss die VariableStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA774
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VALUE_MAX_RANGE-Eigenschaft (Seite 2445)

VariableStateChecked-Eigenschaft (Seite 2478)

VALUE_TIMEOUT-Eigenschaft (Seite 2454)

VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)

VALUE_SERVERDOWN-Eigenschaft (Seite 2451)

VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)

VALUE_MIN_RANGE-Eigenschaft (Seite 2448)

VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)

VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)

VALUE_INVALID_KEY-Eigenschaft (Seite 2439)

VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)

VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)

VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)

VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)

VariableStateValue-Objekt (Seite 2057)

VALUE_HIGHLIMITED-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "high limited" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
```

```
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_INVALID_KEY-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Variablenstatus "Variable nicht gefunden" eintritt, oder gibt ihn zurück.

Damit der Status ausgewertet werden kann, muss die VariableStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA775
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VariableStateChecked-Eigenschaft (Seite 2478)

VALUE_TIMEOUT-Eigenschaft (Seite 2454)

VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)

VALUE_SERVERDOWN-Eigenschaft (Seite 2451)

VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)

VALUE_MIN_RANGE-Eigenschaft (Seite 2448)

VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)

VALUE_MAX_RANGE-Eigenschaft (Seite 2445)

VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)

VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)

VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)

VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)

VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)

VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)

VariableStateValue-Objekt (Seite 2057)

VALUE_LOWLIMITED-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "low limited" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
```

```
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_MAX_LIMIT-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Variablenstatus "Obergrenze überschritten" eintritt, oder gibt ihn zurück.

Damit der Status ausgewertet werden kann, muss die VariableStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA776
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)

VariableStateChecked-Eigenschaft (Seite 2478)

VALUE_TIMEOUT-Eigenschaft (Seite 2454)

VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)

VALUE_SERVERDOWN-Eigenschaft (Seite 2451)

VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)

VALUE_MIN_RANGE-Eigenschaft (Seite 2448)

VALUE_MAX_RANGE-Eigenschaft (Seite 2445)

VALUE_INVALID_KEY-Eigenschaft (Seite 2439)

VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)

VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)

VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)

VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)

VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)

VariableStateValue-Objekt (Seite 2057)

VALUE_MAX_RANGE-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Variablenstatus "Formatgrenze überschritten" eintritt, oder gibt ihn zurück.

Damit der Status ausgewertet werden kann, muss die VariableStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA777
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
```

```
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VariableStateChecked-Eigenschaft (Seite 2478)

VALUE_TIMEOUT-Eigenschaft (Seite 2454)

VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)

VALUE_SERVERDOWN-Eigenschaft (Seite 2451)

VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)

VALUE_MIN_RANGE-Eigenschaft (Seite 2448)

VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)

VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)

VALUE_INVALID_KEY-Eigenschaft (Seite 2439)

VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)

VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)

VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)

VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)

VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)

VariableStateValue-Objekt (Seite 2057)

VALUE_MIN_LIMIT-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Variablenstatus "Untergrenze unterschritten" eintritt, oder gibt ihn zurück.

Damit der Status ausgewertet werden kann, muss die VariableStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA778
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VariableStateChecked-Eigenschaft (Seite 2478)
VALUE_TIMEOUT-Eigenschaft (Seite 2454)
VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)
VALUE_SERVERDOWN-Eigenschaft (Seite 2451)
VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)
VALUE_MIN_RANGE-Eigenschaft (Seite 2448)
VALUE_MAX_RANGE-Eigenschaft (Seite 2445)
VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)
VALUE_INVALID_KEY-Eigenschaft (Seite 2439)
VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)
VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)
VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)
VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)
VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)
VariableStateValue-Objekt (Seite 2057)

VALUE_MIN_RANGE-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Variablenstatus "Formatgrenze unterschritten" eintritt, oder gibt ihn zurück.

Damit der Status ausgewertet werden kann, muss die VariableStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA779
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
```

```
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VariableStateChecked-Eigenschaft (Seite 2478)

VALUE_TIMEOUT-Eigenschaft (Seite 2454)

VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)

VALUE_SERVERDOWN-Eigenschaft (Seite 2451)

VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)

VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)

VALUE_MAX_RANGE-Eigenschaft (Seite 2445)

VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)

VALUE_INVALID_KEY-Eigenschaft (Seite 2439)

VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)

VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)

VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)

VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)

VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)

VariableStateValue-Objekt (Seite 2057)

VALUE_NOT_ESTABLISHED-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Variablenstatus "keine Rückmeldung vom Kanal" eintritt, oder gibt ihn zurück.

Damit der Status ausgewertet werden kann, muss die VariableStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA780
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VariableStateChecked-Eigenschaft (Seite 2478)

VALUE_TIMEOUT-Eigenschaft (Seite 2454)

VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)

VALUE_SERVERDOWN-Eigenschaft (Seite 2451)

VALUE_MIN_RANGE-Eigenschaft (Seite 2448)

VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)

VALUE_MAX_RANGE-Eigenschaft (Seite 2445)

VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)

VALUE_INVALID_KEY-Eigenschaft (Seite 2439)

VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)

VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)

VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)

VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)

VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)

VariableStateValue-Objekt (Seite 2057)

VALUE_SERVERDOWN-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Variablenstatus "Server nicht verfügbar" eintritt, oder gibt ihn zurück.

Damit der Status ausgewertet werden kann, muss die VariableStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA781
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
```

```
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VariableStateChecked-Eigenschaft (Seite 2478)

VALUE_TIMEOUT-Eigenschaft (Seite 2454)

VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)

VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)

VALUE_MIN_RANGE-Eigenschaft (Seite 2448)

VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)

VALUE_MAX_RANGE-Eigenschaft (Seite 2445)

VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)

VALUE_INVALID_KEY-Eigenschaft (Seite 2439)

VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)

VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)

VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)

VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)

VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)

VariableStateValue-Objekt (Seite 2057)

VALUE_STARTUP_VALUE-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Variablenstatus "Startwert" eintritt, oder gibt ihn zurück.

Damit der Status ausgewertet werden kann, muss die VariableStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA782
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VariableStateChecked-Eigenschaft (Seite 2478)
VALUE_TIMEOUT-Eigenschaft (Seite 2454)
VALUE_SERVERDOWN-Eigenschaft (Seite 2451)
VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)
VALUE_MIN_RANGE-Eigenschaft (Seite 2448)
VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)
VALUE_MAX_RANGE-Eigenschaft (Seite 2445)
VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)
VALUE_INVALID_KEY-Eigenschaft (Seite 2439)
VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)
VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)
VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)
VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)
VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)
VariableStateValue-Objekt (Seite 2057)

VALUE_TIMEOUT-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Variablenstatus "keine Verbindung" eintritt, oder gibt ihn zurück.

Damit der Status ausgewertet werden kann, muss die VariableStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA783
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
```

```
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VariableStateChecked-Eigenschaft (Seite 2478)
VALUE_STARTUP_VALUE-Eigenschaft (Seite 2453)
VALUE_SERVERDOWN-Eigenschaft (Seite 2451)
VALUE_NOT_ESTABLISHED-Eigenschaft (Seite 2450)
VALUE_MIN_RANGE-Eigenschaft (Seite 2448)
VALUE_MIN_LIMIT-Eigenschaft (Seite 2447)
VALUE_MAX_RANGE-Eigenschaft (Seite 2445)
VALUE_MAX_LIMIT-Eigenschaft (Seite 2443)
VALUE_INVALID_KEY-Eigenschaft (Seite 2439)
VALUE_HARDWARE_ERROR-Eigenschaft (Seite 2436)
VALUE_HANDSHAKE_ERROR-Eigenschaft (Seite 2434)
VALUE_CONVERSION_ERROR-Eigenschaft (Seite 2432)
VALUE_ADDRESS_ERROR-Eigenschaft (Seite 2411)
VALUE_ACCESS_FAULT-Eigenschaft (Seite 2409)
VariableStateValue-Objekt (Seite 2057)

VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "uncertain, engineering unit range violation, high limit set" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
```

```
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_UNCERT_ENGVLOWLIM-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "uncertain, engineering unit range violation, low limit set" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_UNCERT_ENGVONLIM-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "uncertain, engineering unit range violation, on limits set" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_UNCERT_INITVAL-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "uncertain, initial value" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_UNCERT_LUV-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "uncertain, last usable value" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_UNCERT_MAINTDEM-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "uncertain, maintenance demanded" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_UNCERT_MISCSTATES-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "uncertain miscellanous states" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_UNCERT_NONSPECIFIC-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "uncertain, non-specific" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_UNCERT_PROCRELNOM-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "uncertain, process related, no maintenance" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_UNCERT_SIMVAL-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "uncertain, simulated value" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)
QualityCodeStateChecked-Eigenschaft (Seite 2342)
VALUE_UNCERT_SUBSTSET-Eigenschaft (Seite 2475)
VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)
VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)
VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)
VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)
VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)
VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)
VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)
VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)
VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)
VALUE_LOWLIMITED-Eigenschaft (Seite 2441)
VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)
VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)
VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)
VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)
VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)
VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)
VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)
VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)
VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)
VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)
QualityCodeStateValue-Objekt (Seite 2007)

VALUE_UNCERT_SUBSTSET-Eigenschaft

Beschreibung

Legt den Wert fest, welcher der dynamisierten Eigenschaft zugewiesen wird, wenn der Quality Code "uncertain, substitute set" eintritt, oder gibt ihn zurück.

Damit der Quality Code ausgewertet werden kann, muss die QualityCodeStateChecked-Eigenschaft den Wert TRUE haben.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Quality Code einer Variablen. Falls die Variable keinen Quality Code zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA770
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of qualitycodestate
.QualityCodeStateChecked = True
End With
With objDynDialog.QualityCodeStateValues(1)
'
'define a value for every state:
.VALUE_BAD_COMMLUV = 20
.VALUE_BAD_COMMNUV = 30
.VALUE_BAD_CONFERROR = 40
.VALUE_BAD_DEVICE = 60
.VALUE_BAD_MISCSTATES = 70
.VALUE_BAD_NONSPECIFIC = 80
.VALUE_BAD_NOTCONNECTED = 90
.VALUE_BAD_OUTOFSERV = 100
.VALUE_BAD_PROCRELNOM = 110
.VALUE_BAD_PROCRELSUB = 120
.VALUE_HIGHLIMITED = 130
.VALUE_LOWLIMITED = 140
.VALUE_UNCERT_ENGVHIGHLIM = 150
.VALUE_UNCERT_ENGVLOWLIM = 160
.VALUE_UNCERT_INITVAL = 170
.VALUE_UNCERT_LUV = 180
.VALUE_UNCERT_MAINTDEM = 190
.VALUE_UNCERT_MISCSTATES = 200
.VALUE_UNCERT_NONSPECIFIC = 210
.VALUE_UNCERT_PROCRELNOM = 220
.VALUE_UNCERT_SIMVAL = 230
.VALUE_UNCERT_SUBSTSET = 240
End With
End Sub
```

Siehe auch

VALUE_UNCERT_MISCSTATES-Eigenschaft (Seite 2467)

QualityCodeStateChecked-Eigenschaft (Seite 2342)

VALUE_UNCERT_SIMVAL-Eigenschaft (Seite 2473)

VALUE_UNCERT_PROCRELNOM-Eigenschaft (Seite 2471)

VALUE_UNCERT_NONSPECIFIC-Eigenschaft (Seite 2469)

VALUE_UNCERT_MAINTDEM-Eigenschaft (Seite 2465)

VALUE_UNCERT_LUV-Eigenschaft (Seite 2463)

VALUE_UNCERT_INITVAL-Eigenschaft (Seite 2461)

VALUE_UNCERT_ENGVONLIM-Eigenschaft (Seite 2459)

VALUE_UNCERT_ENGVLOWLIM-Eigenschaft (Seite 2457)

VALUE_UNCERT_ENGVHIGHLIM-Eigenschaft (Seite 2456)

VALUE_LOWLIMITED-Eigenschaft (Seite 2441)

VALUE_HIGHLIMITED-Eigenschaft (Seite 2437)

VALUE_BAD_PROCRELSUB-Eigenschaft (Seite 2430)

VALUE_BAD_PROCRELNOM-Eigenschaft (Seite 2428)

VALUE_BAD_OUTOFSERV-Eigenschaft (Seite 2426)

VALUE_BAD_NOTCONNECTED-Eigenschaft (Seite 2424)

VALUE_BAD_NONSPECIFIC-Eigenschaft (Seite 2422)

VALUE_BAD_MISCSTATES-Eigenschaft (Seite 2420)

VALUE_BAD_DEVICE-Eigenschaft (Seite 2418)

VALUE_BAD_CONFERROR-Eigenschaft (Seite 2416)

VALUE_BAD_COMMNUV-Eigenschaft (Seite 2414)

VALUE_BAD_COMMLUV-Eigenschaft (Seite 2412)

QualityCodeStateValue-Objekt (Seite 2007)

VariablesExist-Eigenschaft

Beschreibung

TRUE, wenn alle im Quellcode eines DynamicDialog-Objekts verwendeten Variablen definiert sind. Lese-Zugriff.

Mit dieser Eigenschaft können Sie überprüfen, ob alle Variablen, die Sie im Quellcode des Dynamik-Dialoges definiert haben, in WinCC angelegt sind.

Beispiel

--

Siehe auch

DynamicDialog-Objekt (Seite 1924)

VariableStateChecked-Eigenschaft

Beschreibung

TRUE, wenn der Status der angegebenen Variable im Dynamik-Dialog zur Dynamisierung verwendet wird. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert, ein Variablenname vergeben und drei analoge Wertebereiche angelegt:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA785
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

DynamicDialog-Objekt (Seite 1924)

VariableStateType-Eigenschaft

Beschreibung

Gibt die Art der Variablenüberwachung zurück, mit der eine Eigenschaft oder ein Ereignis dynamisiert wurde: keine Überwachung, Quality Code oder Variablenstatus. Lese-Zugriff.

Index	VariableStateType
0	hmiNoVariableState
1	hmiVariableQCState
2	hmiVariableState

Beispiel

Die Prozedur "GetVariableStateType()" liest die Art der Überwachung beim aktuellen Dokument aus. In diesem Beispiel wird die Art der Überwachung in einer Meldung ausgegeben:

```
Sub GetVariableStateType()
'VBA819
Dim objDyn As HMIDynamicDialog
Set objDyn =
ActiveDocument.Properties("Width").CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'TestVal'")
MsgBox objDyn.VariableStateType
objDyn.Delete
End Sub
```

Siehe auch

DynamicDialog-Objekt (Seite 1924)

VariableStateValues-Eigenschaft

Beschreibung

Gibt die VariableStateValues-Auflistung zurück. Verwenden Sie die VariableStateValues-Eigenschaft mit der Item-Eigenschaft, um einem Variablenstatus einen Wert zuzuordnen, der für die Dynamisierung verwendet wird.

Beispiel

Im folgenden Beispiel wird der Radius eines Kreises mit dem Dynamik-Dialog dynamisiert. Die Dynamisierung erfolgt über die Auswertung des Status einer Variablen. Falls die Variable keinen Status zurückgibt, wird ein Ersatzwert (ElseCase-Eigenschaft) definiert, ein Variablenname vergeben und drei analoge Wertebereiche angelegt:

```
Sub AddDynamicDialogToCircleRadiusTypeAnalog()
'VBA786
Dim objDynDialog As HMIDynamicDialog
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_A", "HMICircle")
Set objDynDialog = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeDynamicDialog,
"'NewDynamic1'")
With objDynDialog
.ResultType = hmiResultTypeAnalog
.AnalogResultInfos.ElseCase = 200
'
'Activate analysis of variablestate
.VariableStateChecked = True
End With
With objDynDialog.VariableStateValues(1)
'
'define a value for every state:
.VALUE_ACCESS_FAULT = 20
.VALUE_ADDRESS_ERROR = 30
.VALUE_CONVERSION_ERROR = 40
.VALUE_HANDSHAKE_ERROR = 60
.VALUE_HARDWARE_ERROR = 70
.VALUE_INVALID_KEY = 80
.VALUE_MAX_LIMIT = 90
.VALUE_MAX_RANGE = 100
.VALUE_MIN_LIMIT = 110
.VALUE_MIN_RANGE = 120
.VALUE_NOT_ESTABLISHED = 130
.VALUE_SERVERDOWN = 140
.VALUE_STARTUP_VALUE = 150
.VALUE_TIMEOUT = 160
End With
End Sub
```

Siehe auch

VariableStateValues-Objekt (Auflistung) (Seite 2058)

DynamicDialog-Objekt (Seite 1924)

VariableTriggers-Eigenschaft

Beschreibung

Gibt die VariableTriggers-Auflistung zurück. Verwenden Sie die VariableTriggers-Eigenschaft, um einer VB- oder C-Aktion einen Variablentrigger hinzuzufügen.

Beispiel

Im folgenden Beispiel wird der Kreisradius mit einem VB-Skript dynamisiert. Als Trigger wird ein Variablen-Trigger verwendet:

```
Sub DynamicWithVariableTrigger()
'VBA787
Dim objVBScript As HMIScriptInfo
Dim objVarTrigger As HMIVariableTrigger
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("Circle_VariableTrigger",
"HMICircle")
Set objVBScript = objCircle.Radius.CreateDynamic(hmiDynamicCreationTypeVBScript)
With objVBScript
'Triggername and cycletime are defined by add-methode
Set objVarTrigger = .Trigger.VariableTriggers.Add("VarTrigger", hmiVariableCycleType_10s)
.SourceCode = ""
End With
End Sub
```

Siehe auch

VariableTriggers-Objekt (Auflistung) (Seite 2061)

VarName-Eigenschaft

Beschreibung

Legt die Variable fest, deren Status im Dynamik-Dialog zur Dynamsierung verwendet werden soll oder gibt den Namen zurück.

Beispiel

In diesem Beispiel wird der Name der Triggervariablen ausgegeben, die zur Dynamisierung des Kreisradius verwendet wird:

```
Sub GetVarName()
'VBA788
Dim objVBScript As HMIScriptInfo
Dim objCircle As HMICircle
```

```
Set objCircle = ActiveDocument.HMIObjects.Item("Circle_VariableTrigger")
Set objVBScript = objCircle.Radius.Dynamic
With objVBScript
'Reading out of variablename
MsgBox "The radius is dynamicabled with: " & .Trigger.VariableTriggers.Item(1).VarName
End With
End Sub
```

Siehe auch

VariableStateValue-Objekt (Seite 2057)

VBAVersion-Eigenschaft

Beschreibung

Gibt die VBA-Versionsnummer zurück. Lese-Zugriff.

Beispiel

Im folgenden Beispiel wird die aktuelle VBA-Versionsnummer ausgegeben:

```
Sub ShowVBAVersion()
'VBA789
MsgBox Application.VBAVersion
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

VBE-Eigenschaft

Beschreibung

Gibt das VB-Extensibility-Objekt zurück. Lese-Zugriff

Beispiel

--

Siehe auch

Application-Objekt (Seite 1888)

Version-Eigenschaft

Beschreibung

Gibt die Versionsnummer der angegebenen Anwendung zurück. Lese-Zugriff.

Beispiel

Im folgenden Beispiel wird die Versionsnummer des Graphics Designer ausgegeben:

```
Sub ShowVersionOfGraphicsDesigner()
'VBA791
MsgBox Application.Version
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

Views-Eigenschaft

Beschreibung

Gibt die Views-Auflistung zurück. Verwenden Sie die Views-Auflistung, um z.B. eine neue Kopie eines Bildes zu erstellen.

Beispiel

Im folgenden Beispiel wird vom aktiven Bild eine Kopie erzeugt und aktiviert:

```
Sub AddView()
'VBA792
Dim objView As HMIView
Set objView = ActiveDocument.Views.Add
objView.Activate
End Sub
```

Siehe auch

Views-Objekt (Auflistung) (Seite 2064)

Visible-Eigenschaft

Beschreibung

TRUE, wenn das angegebene Objekt sichtbar sein soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Im folgenden Beispiel wird ein Kreis in das aktive Bild eingefügt. In Runtime soll dieser Kreis nicht sichtbar sein:

```
Sub HideCircleInRuntime()
'VBA793
Dim objCircle As HMICircle
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject ("myCircle", "HMICircle")
objCircle.Visible = False
End Sub
```

Siehe auch

ToolbarItem-Objekt (Seite 2043)

MenuItem-Objekt (Seite 1979)

HMIObject-Objekt (Seite 1955)

Document-Objekt (Seite 1920)

Toolbar-Objekt (Seite 2040)

Menu-Objekt (Seite 1976)

Application-Objekt (Seite 1888)

W - Z

WarningHigh-Eigenschaft

Beschreibung

Legt beim BarGraph-Objekt den oberen Grenzwert "Warning High" fest oder gibt ihn zurück.

Damit der Grenzwert überwacht wird, muss die Eigenschaft "CheckWarningHigh" auf "True" gesetzt sein.

Die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften ColorWarningHigh und TypeWarningHigh festgelegt.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des oberen Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "75" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA794
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis = absolute
.TypeWarningHigh = False
'Activate monitoring
.CheckWarningHigh = True
'Set barcolor = "red"
.ColorWarningHigh = RGB(255, 0, 0)
'Set upper limit = "75"
.WarningHigh = 75
End With
End Sub
```

Siehe auch

TypeWarningHigh-Eigenschaft (Seite 2399)

ColorWarningHigh-Eigenschaft (Seite 2147)

CheckWarningHigh-Eigenschaft (Seite 2134)

BarGraph-Objekt (Seite 1893)

WarningLow-Eigenschaft

Beschreibung

Legt beim BarGraph-Objekt den unteren Grenzwert "Warning Low" fest oder gibt ihn zurück.

Damit der Grenzwert überwacht wird, muss die Eigenschaft "CheckWarningLow" auf "True" gesetzt sein.

Die Darstellung beim Erreichen des Grenzwerts und die Art der Auswertung werden über die Eigenschaften ColorWarningLow und TypeWarningLow festgelegt.

Beispiel

Die Prozedur "BarGraphLimitConfiguration()" konfiguriert die Eigenschaften des unteren Grenzwertes für einen Alarm. In diesem Beispiel wird die Art der Auswertung auf "absolut" gesetzt. Der Alarm soll bei einem Wert von "12" ausgelöst werden.

```
Sub BarGraphLimitConfiguration()
'VBA795
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
'Set analysis = absolute
.TypeWarningLow = False
'Activate monitoring
.CheckWarningLow = True
'Set barcolor = "magenta"
.ColorWarningLow = RGB(255, 0, 255)
'Set lower limit = "12"
.WarningLow = 75
End With
End Sub
```

Siehe auch

TypeWarningLow-Eigenschaft (Seite 2400)

ColorWarningLow-Eigenschaft (Seite 2148)

CheckWarningLow-Eigenschaft (Seite 2135)

BarGraph-Objekt (Seite 1893)

Width-Eigenschaft

Beschreibung

Legt die Breite eines Objektes in Pixel fest oder gibt sie zurück.

Beispiel

Im folgenden Beispiel werden drei Objekte von unterschiedlicher Größe ins aktive Bild eingefügt. Danach werden alle Objekte ausgewählt und auf dieselbe Breite gesetzt:

```
Sub ApplySameWidthToSelectedObjects()
'VBA796
Dim objCircle As HMICircle
Dim objRectangle As HMIRectangle
Dim objEllipse As HMIEllipse
Set objCircle = ActiveDocument.HMIObjects.AddHMIObject("sCircle", "HMICircle")
Set objRectangle = ActiveDocument.HMIObjects.AddHMIObject("sRectangle", "HMIRectangle")
```

```
Set objEllipse = ActiveDocument.HMIObjects.AddHMIObject("sEllipse", "HMIEllipse")
With objCircle
.Top = 30
.Left = 0
.Width = 15
.Selected = True
End With
With objRectangle
.Top = 80
.Left = 42
.Width = 40
.Selected = True
End With
With objEllipse
.Top = 48
.Left = 162
.Width = 120
.BackColor = RGB(255, 0, 0)
.Selected = True
End With
MsgBox "Objects selected!"
ActiveDocument.Selection.SameWidth
End Sub
```

Siehe auch

HMIObject-Objekt (Seite 1955)

WinCCStyle-Eigenschaft

Beschreibung

Legt fest, in welchem Stil das Objekt dargestellt wird.

benutzerdefiniert	Stellt das Objekt entsprechend den eigenen Einstellungen dar.
global	Stellt das Objekt im global eingestellten Design dar.
Windows-Stil	Stelt das Objekt im Windows-Stil dar.

Beispiel

WindowBorder-Eigenschaft

Beschreibung

TRUE, wenn das Anwendungs- oder Bildfenster in Runtime mit Rahmen dargestellt werden soll. BOOLEAN Schreib-Lese-Zugriff.

Beispiel

Die Prozedur "ApplicationWindowConfig" greift auf die Eigenschaften des Applikationsfensters zu. In diesem Beispiel wird das Applikationsfenster konfiguriert:

```
Sub ApplicationWindowConfig()
'VBA797
Dim objAppWindow As HMIApplicationWindow
Set objAppWindow = ActiveDocument.HMIObjects.AddHMIObject("AppWindow",
"HMIApplicationWindow")
With objAppWindow
.Caption = True
.CloseButton = False
.Height = 200
.Left = 10
.MaximizeButton = True
.Moveable = False
.OnTop = True
.Sizeable = True
.Top = 20
.Visible = True
.Width = 250
.WindowBorder = True
End With
End Sub
```

Siehe auch

PictureWindow-Objekt (Seite 1992)

ApplicationWindow-Objekt (Seite 1891)

WindowMonitorNumber-Eigenschaft

Beschreibung

Legt fest, auf welchem Monitor das Bildfenster angezeigt wird. Voraussetzung ist, dass das System mehrere Monitore unterstützt. Das Attribut wirkt nur, wenn das Attribut "Unabhängiges Bildfenster" auf "ja" gesetzt ist.

1-n Die Nummer des Monitors im Betriebssystem, auf dem das Bildfenster angezeigt wird

Beispiel

WindowPositionMode-Eigenschaft

Beschreibung

Legt die Position und Skalierung des Bildfensters auf dem Bildschirm fest. Die Eigenschaft wirkt sich nur aus, wenn das Attribut "Unabhängiges Fenster" auf "ja" gesetzt ist.

Standard	Das Bildfenster wird in Originalgröße an der projektierten Stelle auf dem Bildschirm positioniert.
Zentrieren	Das Bildfenster wird in Originalgröße mittig auf dem Bildschirm positioniert.
Maximieren	Das Bildfenster wird auf die Größe des Bildschirms skaliert.

Beispiel

WindowsStyle-Eigenschaft

Beschreibung

Legt fest, ob das Objekt im Windows-Stil von WinCC Version 6.2 dargestellt wird. Es ist nur auswählbar, wenn als aktuelles Design "WinCC Classic" ausgewählt ist.

ja	Stellt das Objekt im Windows-Stil von WinCC Version 6.2 dar.
nein	Stellt das Objekt nicht im Windows-Stil von WinCC Version 6.2 dar.

Beispiel

WindowState-Eigenschaft

Beschreibung

Gibt den Status des Fensters zurück, das die angegebene Anwendung enthält. LESE-Zugriff.

WindowState	zugeordneter Wert
Maximized	0
Minimized	1
CustomSized	2

Beispiel

Im folgenden Beispiel wird der Fensterstatus des Graphics Designer ausgegeben:

```
Sub ShowWindowState()
'VBA798
Dim strState As String
Select Case Application.WindowState
Case 0
strState = "The application-window is maximized"
Case 1
strState = "The applicationwindow is minimized"
Case 2
strState = "The application-window has a userdefined size"
End Select
MsgBox strState
End Sub
```

Siehe auch

Application-Objekt (Seite 1888)

ZeroPoint-Eigenschaft

Beschreibung

Legt bei dem Objekt BarGraph die Lage des Nullpunktes des Balkens fest oder gibt sie zurück.

Geben Sie den Wert in % zur Gesamtbalkenhöhe an. Der Nullpunkt kann auch außerhalb des dargestellten Bereiches liegen.

Die Eigenschaft "ScalingType" muss auf "2" und "Scaling" auf "True" gesetzt sein.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel befindet sich der Nullpunkt auf der Hälfte der Balkenhöhe:

```
Sub BarGraphConfiguration()
'VBA799
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.Scaling = True
.ScalingType = 2
.ZeroPoint = 50
.ZeroPointValue = 0
End With
End Sub
```

Siehe auch

ZeroPointValue-Eigenschaft (Seite 2491)

ScalingType-Eigenschaft (Seite 2356)

Scaling-Eigenschaft (Seite 2355)

BarGraph-Objekt (Seite 1893)

ZeroPointValue-Eigenschaft

Beschreibung

Legt den absoluten Wert für den Nullpunkt fest oder gibt ihn zurück.

Beispiel

Die Prozedur "BarGraphConfiguration()" konfiguriert die Eigenschaften des BarGraph-Objektes. In diesem Beispiel wird der Absolutwert des Nullpunktes auf "0" gesetzt:

```
Sub BarGraphConfiguration()
'VBA800
Dim objBarGraph As HMIBarGraph
Set objBarGraph = ActiveDocument.HMIObjects.AddHMIObject("Bar1", "HMIBarGraph")
With objBarGraph
.Scaling = True
.ScalingType = 2
.ZeroPointValue = 0
End With
End Sub
```

Siehe auch

ZeroPoint-Eigenschaft (Seite 2490)

ScalingType-Eigenschaft (Seite 2356)

Scaling-Eigenschaft (Seite 2355)

BarGraph-Objekt (Seite 1893)

3DBarGraph-Objekt (Seite 1879)

Zoom-Eigenschaft

Beschreibung

Legt den Zoomfaktor fest oder gibt ihn zurück.

Beispiel

In diesem Beispiel wird vom aktiven Bild eine Kopie erzeugt und der Zoomfaktor auf 50% gesetzt:

```
Sub CreateViewFromActiveDocument()
'VBA801
Dim objView As HMIView
Set objView = ActiveDocument.Views.Add
objView.Zoom = 50
End Sub
```

Siehe auch

View-Objekt (Seite 2062)

PictureWindow-Objekt (Seite 1992)

3.5.2 VBA in weiteren WinCC-Editoren

3.5.2.1 VBA in weiteren WinCC-Editoren

Einleitung

Mit VBA haben Sie die Möglichkeit, auf andere WinCC-Editoren wie z.B. das Tag Logging zuzugreifen. Folgende Editoren können neben dem Graphics Designer mit VBA automatisiert werden:

- Variablenhaushalt
- Tag Logging
- Text Library
- Alarm Logging

Die Funktionen zum Zugriff auf die Editoren sind in der Klasse HMIGO enthalten.

Voraussetzung

Die Datei "HMIGenObjects.dll" ist referenziert. Dies geschieht während der WinCC-Installation automatisch.

Prinzip

Damit Sie mit VBA Zugriff auf die Klasse HMIGO haben, müssen Sie im VBA-Editor die "HMI GeneralObjects 1.0 Type Library" referenzieren ("Project" > "References"). Im Programmcode müssen Sie eine neue Instanz dieser Klasse erzeugen, z.B.:

```
'Dim HMIGOObject As New HMIGO
```

Erzeugen Sie mehrere unterschiedliche Objekte dieser Klasse, wenn Sie auf mehrere Objekte zur gleichen Zeit zugreifen. Sie benötigen im Tag Logging z.B. zwei Instanzen der Klasse HMIGO: die erste Instanz benötigen Sie zum Zugriff auf die Archivvariablen, die zweite Instanz zum Zugriff auf das Prozesswertarchiv selber.

Verwendung

Mit den von der Klasse HMIGO zur Verfügung gestellten Funktionen haben Sie Zugriff auf den Variablenhaushalt, das Tag Logging, die Text Library und das Alarm Logging. Damit Sie die Funktionen in VBA verwenden können, müssen Sie in WinCC ein Projekt geöffnet haben. Darüber hinaus haben Sie die Möglichkeit, auf die Eigenschaften der Klasse direkt zuzugreifen.

Damit können Sie direkt aus dem Programmcode beispielsweise mehrere Variablen erzeugen und deren Werte ändern, Texteinträge in der TextLibrary bearbeiten oder Meldungen anpassen.

Objektstatus abfragen

Die Klasse HMIGO verfügt über die Enumeration HMIGO_OBJECT_STATE, die den Zustand des angegebenen Objektes zurückliefert. Die Enumeration kann folgende Werte zurückliefern:

- OBJECT_EMPTY (2): Verbindung zum Objekt ist nicht vorhanden.
- OBJECT_OPENED (3): Verbindung zu einem Objekt besteht. Sie können seine Parameter ändern oder lesen.
- OBJECT_MODIFIED (4): Die Parameter eines Objektes wurden geändert. Ohne den Aufruf der entsprechenden Commit-Funktion gehen die Änderungen verloren.
- WINCC_CONNECTED (1): Das Objekt ist mit dem WinCC-Projekt verbunden. Diese Verbindung wird standardmäßg beim ersten Funktionsaufruf hergestellt. Um diese Verbindung aufzulösen, verwenden Sie z.B. die Anweisung HMIGO = nothing.

Fehlerbehandlung

Bei der Verwendung der Klasse HMIGO können Fehler auftreten. Verwenden Sie die Anweisung OnError, um auf diese Fehlermeldungen zu reagieren. Die Anweisung OnError muss vor dem Aufruf einer Funktion aus der HMIGeneralObjects-Klasse stehen:

```
Sub CreateTag()
'HMIGO_000
Dim hmiGOTag as New HMIGO
On Error GoTo ErrorHandlerHMIGO
hmiGOTag.CreateTag "NewTag", TAG_BINARY_TAG, "ExistingConnection", "DB1,DD0,QC",
"NewOrExistingGroupName"

'...
Exit Sub
ErrorHandlerHMIGO:
MsgBox ("Error: " & Err.Number & " " & Err.Description & " " & Err.Source)
Resume Next
End Sub
```

Dadurch wird ein vom Interface zurückgegebener Fehlertext ausgegeben.

Siehe auch

VBA im Alarm Logging (Seite 2547)

VBA in der Text Library (Seite 2534)

VBA im Tag Logging (Seite 2506)

VBA im Variablenhaushalt (Seite 2495)

3.5.2.2 VBA im Variablenhaushalt

VBA im Variablenhaushalt

Einleitung

Mit VBA können Sie Variablen direkt aus dem Programmcode erzeugen, modifizieren und löschen sowie deren Eigenschaften, Typen und Werte auslesen und ändern.

Prinzip

Wenn Sie die Instanz der Klasse HMIGO erzeugt haben, stehen Ihnen folgende Funktionen zum Zugriff auf den Variablenhaushalt zur Verfügung:

- CloseTag
- CommitTag
- CreateTag
- DeleteTag
- GetTag
- ListTag

Für die Parameterversorgung dieser Funktionen steht die Enumeration "HMIGO_TAG_TYPE" und "HMIGO_TAG_LIST_TYPE" zur Verfügung.

Hinweis

Die Variable darf während der Bearbeitung mit VBA nicht im Variablenhaushalt geöffnet sein oder geöffnet werden.
Wenn Sie den Datentyp einer Variablen ändern wollen, müssen Sie die Variable zuerst löschen und danach neu generieren. Die Parameter müssen Sie zuvor sichern, um diese Parameter nach der Erzeugung der Variablen zu übertragen.

Hinweis

Wenn Sie den Startwert einer binären Variable setzen, verwenden Sie die Werte "0" bzw. "1". Verwenden Sie nicht die Werte "False" bzw. "True". Diese Werte sind für die VBA-Programmierung in WinCC nicht mehr gültig und verursachen eine Fehlermeldung.

Ersetzen Sie in ihren bestehenden VBA-Codes die Werte "False" und "True" durch "0" und "1".

Zugriff auf die Objekteigenschaften

Auf die Parameter der oben genannten Funktionen können Sie auch direkt in VBA über folgende Objekteigenschaften zugreifen:

Objekteigenschaft	Beschreibung	Lesen/Schreiben
ObjectStateTag	Gibt den Objektstatus über die Enumeration HMIGO_OBJECT_STATE zurück. Weitere Informationen zu dieser Enumeration finden Sie in dieser Dokumentation unter "VBA in weiteren WinCC-Editoren".	ja/nein
TagName	Name der Variablen	ja/nein
TagGroupName	Name einer Gruppe, in der die Variable eingefügt wird. Wenn die Gruppe noch nicht existiert wird sie neu angelegt. Ohne Angabe eines Gruppennamens wird die Variable außerhalb aller Gruppen angelegt.	ja/nein
TagConnection	Name einer Verbindung, in der die Variable und/oder Gruppe neu anzulegen ist. Es muss sich um eine bereits existierende Verbindung handeln, sonst kann keine Variable angelegt werden. Wird der Name weggelassen, wird eine interne Variable angelegt.	ja/nein
TagMaximum	Setzt den neuen Wert der Obergrenze	ja/ja
TagMinimum	Setzt den neuen Wert der Untergrenze	ja/ja
TagStart	Setzt den neuen Startwert	ja/ja
TagS5S7Address	Adresse der S7- oder S5-Steuerung, mit der die Variable verbunden ist. Ohne Angabe einer Adresse wird ein leerer Eintrag übergeben.	ja/ja
TagType (Enum)	Datentyp der Variablen. Mögliche Typen sind: • TAG_BINARY_TAG (1) • TAG_SIGNED_8BIT_VALUE (2) • TAG_UNSIGNED_8BIT_VALUE (3) • TAG_SIGNED_16BIT_VALUE (4) • TAG_UNSIGNED_16BIT_VALUE (5) • TAG_SIGNED_32BIT_VALUE (6) • TAG_UNSIGNED_32BIT_VALUE (7) • TAG_FLOATINGPOINT_NUMBER_32BIT_IEEE_754 (8) • TAG_FLOATINGPOINT_NUMBER_64BIT_IEEE_754 (9) • TAG_TEXT_TAG_8BIT_CHARACTER_SET (10) • TAG_TEXT_TAG_16BIT_CHARACTER_SET (11) • TAG_RAW_DATA_TYPE (12) • TAG_STRUCT (14) • TAG_TEXT_REFERENCE (18)	ja/nein
TagUpdate (Enum)	Definiert, ob die Variable nur rechnerlokal oder projektweit aktualisiert wird. (nur für interne Variable) • TAG_COMPUTER_LOCAL (1) • TAG_PROJECT_WIDE (2)	ja/ja

Objekteigenschaft	Beschreibung	Lesen/Schreiben
LengthText	Länge einer Textvariable (0...255) "LengthText" ist auch für die Länge der Rohdatenvariablen verwendbar. Eine Prüfung auf Richtigkeit der Länge wird nicht vorgenommen. Beachten Sie die Vorgaben der Kommunikationskanäle.	ja/ja (nur für externe Variable vom Typ Text)
TagScaleValid	Legt eine lineare Skalierung fest.	ja/ja
TagScaleParam1	Setzt den Wert1 bei Wertebereich Prozess.	ja/ja
TagScaleParam2	Setzt den Wert2 bei Wertebereich Prozess.	ja/ja
TagScaleParam3	Setzt den Wert1 bei Wertebereich Variable.	ja/ja
TagScaleParam4	Setzt den Wert2 bei Wertebereich Variable.	ja/ja
TagStartvaluePersistence	Legt fest, ob eine interne Variable persistent gesetzt ist.	ja/ja
TagSubst	Ersatzwert (nur bei externen Variablen)	ja/ja
UseSubstValueOnCommonError	Setzt den Ersatzwert bei Verbindungsfehlern.	ja/ja
UseSubstValueOnMaxLimit	Setzt den Ersatzwert der Obergrenze.	ja/ja
UseSubstValueOnMinLimit	Setzt den Ersatzwert der Untergrenze.	ja/ja
UseSubstValueOnStartValue	Setzt den Ersatzwert des Startwerts.	ja/ja

Eine Beschreibung der Eigenschaften finden Sie unter der Parameterbeschreibung bei den entsprechenden Funktionen.

Hinweis

Der Punkt "Variablensynchronisation" im Eigenschaftsdialog von Variablen ist nicht mit VBA ansprechbar. Die Variablensynchronisation steht nur für interne Variablen zur Verfügung.

Bei externen Variablen ist der Punkt "Formatanpassung" nicht mit VBA ansprechbar.

Siehe auch

ListTag-Funktion (Seite 2504)

GetTag-Funktion (Seite 2503)

DeleteTag-Funktion (Seite 2502)

CreateTag-Funktion (Seite 2500)

CommitTag-Funktion (Seite 2499)

CloseTag-Funktion (Seite 2498)

VBA in weiteren WinCC-Editoren (Seite 2493)

CloseTag-Funktion

Beschreibung

Schließt die geöffnete Variable.

Hinweis

Geänderte Parameter werden nicht gespeichert.

Syntax

```
Ausdruck.CloseTag()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

--

Beispiel

```
Sub CloseTag()
' HMIGO_001
' procedure to close a variable
' tag need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strVariableName As String
  Set objHMIGO = New HMIGO
  strVariableName = "NewVariable"
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTag, vbOKOnly, "Status Variable"
 'open a tag
  objHMIGO.GetTag strVariableName
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateTag, vbOKOnly, "Status Variable"
 'close a tag
  objHMIGO.CloseTag
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTag, vbOKOnly, "Status Variable"

  Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListTag-Funktion (Seite 2504)

GetTag-Funktion (Seite 2503)

DeleteTag-Funktion (Seite 2502)

CreateTag-Funktion (Seite 2500)

CommitTag-Funktion (Seite 2499)

VBA im Variablenhaushalt (Seite 2495)

CommitTag-Funktion

Beschreibung

Schreibt die geänderten Parameter der geöffneten Variable nach WinCC.

Hinweis

Wenn Sie nach einem CommitTag-Aufruf weitere Parameter ändern, müssen Sie diese Änderungen mit einem erneuten Aufruf dieser Funktion nach WinCC schreiben.

Syntax

```
Ausdruck.CommitTag()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

--

Beispiel

```
Sub CommitTag()
' HMIGO_002
' procedure to change a property of a variable
' tag need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strVariableName As String
  Set objHMIGO = New HMIGO
  strVariableName = "NewVariable"
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTag, vbOKOnly, "Status Variable"
 'open a tag
```

```
  objHMIGO.GetTag strVariableName
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateTag, vbOKOnly, "Status Variable"
 'change a property
  objHMIGO.TagStart = 10
 'current status is "MODIFIED"
  MsgBox objHMIGO.ObjectStateTag, vbOKOnly, "Status Variable"
 'commit a tag
  objHMIGO.CommitTag
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateTag, vbOKOnly, "Status Variable"

  Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListTag-Funktion (Seite 2504)

GetTag-Funktion (Seite 2503)

DeleteTag-Funktion (Seite 2502)

CreateTag-Funktion (Seite 2500)

CloseTag-Funktion (Seite 2498)

VBA im Variablenhaushalt (Seite 2495)

CreateTag-Funktion

Beschreibung

Erzeugt eine neue Variable.

Syntax

```
Ausdruck.CreateTag(TagName,TagType,[Connection],[S7S5Address],
[GroupName])
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
TagName (String)	Name der Variablen, die zu erzeugen ist.
TagType (HMIGO_TAG_TYPE)	Datentyp der Variablen. Mögliche Typen sind: TAG_BINARY_TAG TAG_SIGNED_8BIT_VALUE TAG_UNSIGNED_8BIT_VALUE TAG_SIGNED_16BIT_VALUE TAG_UNSIGNED_16BIT_VALUE TAG_SIGNED_32BIT_VALUE TAG_UNSIGNED_32BIT_VALUE TAG_FLOATINGPOINT_NUMBER_32BIT_IEEE_754 TAG_FLOATINGPOINT_NUMBER_64BIT_IEEE_754 TAG_TEXT_TAG_8BIT_CHARACTER_SET TAG_TEXT_TAG_16BIT_CHARACTER_SET TAG_RAW_DATA_TYPE TAG_TEXT_REFERENCE
Connection (String, optional)	Name einer Verbindung, in der die Variable und/oder Gruppe neu anzulegen ist. Es muss sich um eine bereits existierende Verbindung handeln, sonst kann keine Variable angelegt werden. Wird der Name weggelassen, wird eine interne Variable und/oder Gruppe neu angelegt.
S7S5Address (String, optional)	Adresse der S7- oder S5-Steuerung, mit der die Variable verbunden ist. Ohne Angabe einer Adresse wird ein leerer Eintrag übergeben. Für die Konfiguration des Quality Codes muss der Parameter "S7S5Address" um den String ",QC" ergänzt werden, zum Beispiel: "DB1,DD0,QC". Wenn der Quality Code der Variablen nicht mehr überwacht werden soll, dann muss der String ",QC" gelöscht werden.
GroupName (String, optional)	Name einer Gruppe, in der die Variable eingefügt wird. Wenn die Gruppe noch nicht existiert wird sie neu angelegt. Ohne Angabe eines Gruppennamens wird die Variable außerhalb aller Gruppen angelegt.

Beispiel

```
Sub CreateTag()
' HMIGO_003
' procedure to create a variable
' tag must not be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strVariableName As String
  Set objHMIGO = New HMIGO
```

```
  strVariableName = "NewVariable"
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTag, vbOKOnly, "Status Variable"
 'create a tag
  objHMIGO.CreateTag strVariableName, TAG_SIGNED_32BIT_VALUE
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateTag, vbOKOnly, "Status Variable"
  Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListTag-Funktion (Seite 2504)
GetTag-Funktion (Seite 2503)
DeleteTag-Funktion (Seite 2502)
CommitTag-Funktion (Seite 2499)
CloseTag-Funktion (Seite 2498)
VBA im Variablenhaushalt (Seite 2495)

DeleteTag-Funktion

Beschreibung

Löscht die angegebene Variable.

Syntax

```
Ausdruck.DeleteTag(TagName)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
TagName (String)	Name der Variablen, die gelöscht werden soll.

Beispiel

```
Sub DeleteTag()
' HMIGO_004
' procedure to delete a variable
```

```
' tag need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strVariableName As String
  Set objHMIGO = New HMIGO
  strVariableName = "NewVariable"
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTag, vbOKOnly, "Status Variable"

 'delete a tag
  objHMIGO.DeleteTag strVariableName
  Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListTag-Funktion (Seite 2504)

GetTag-Funktion (Seite 2503)

CreateTag-Funktion (Seite 2500)

CommitTag-Funktion (Seite 2499)

CloseTag-Funktion (Seite 2498)

VBA im Variablenhaushalt (Seite 2495)

GetTag-Funktion

Beschreibung

Liest die Parameter der angegebenen Variable ein.

Die Parameter können Sie über die Objekteigenschaften ändern oder lesen. Eine Auflistung der verfügbaren Objekteigenschaften finden Sie in dieser Dokumentation unter "VBA im Variablenhaushalt".

Syntax

```
Ausdruck.GetTag(TagName)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
TagName (String)	Name der Variablen, deren Werte eingelesen werden sollen.

Beispiel

```
Sub GetTag()
' HMIGO_005
' procedure to open a variable
' tag need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strVariableName As String
  Set objHMIGO = New HMIGO
  strVariableName = "NewVariable"
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTag, vbOKOnly, "Status Variable"
 'open/ get a tag
  objHMIGO.GetTag strVariableName
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateTag, vbOKOnly, "Status Variable"
  Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListTag-Funktion (Seite 2504)
DeleteTag-Funktion (Seite 2502)
CreateTag-Funktion (Seite 2500)
CommitTag-Funktion (Seite 2499)
CloseTag-Funktion (Seite 2498)
VBA im Variablenhaushalt (Seite 2495)

ListTag-Funktion

Beschreibung

Die ListTag-Funktion gibt alternativ folgende Inhalte des Variablenhaushalts als Liste zurück:

- alle erstellten Kanal-Units
- alle erstellten Kanäle
- alle erstellten Verbindungen
- alle erstellten Variablengruppen
- alle erstellten Variablen

Syntax

```
Ausdruck.ListTag(ListType,pListArray,[Filter])
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ListType (HMIGO_TAG_LIST_TYPE)	Legt fest, welcher Inhalt als Liste zurückgegeben wird. Möglich ist: • TAG_CHANNEL_UNITS (0) alle erstellten Kanal-Units • TAG_CHANNELS (2) alle erstellten Kanäle • TAG_CONNECTIONS (3) alle erstellten Verbindungen • TAG_GROUPS (4) alle erstellten Variablengruppen • TAG_NAMES (5) alle erstellten Variablen
pListArray (Variant)	Liste mit dem angeforderten Inhalt.
Filter (String)	Optional können Filter gesetzt werden. Unterstützt werden auch die Wildcards "*" und "?".

Beispiel

Im folgenden Beispiel wird überprüft, ob die Liste mit den erstellten Verbindungen leer ist, da keine Verbindungen angelegt sind:

```
Sub ReadTagByConnection()
'HMIGO_027
'read content in data manager by connections
'no conections are implemented
  Dim objHMIGO As New HMIGO
  Dim arrContent As Variant
'read all connections
  objHMIGO.ListTag TAG_CONNECTIONS, arrContent
'check result
  If (UBound(arrContent) - LBound(arrContent) + 1) <= 0 Then
    MsgBox "no entries because no connections are implemented"
  End If
End Sub
```

Siehe auch

GetTag-Funktion (Seite 2503)

DeleteTag-Funktion (Seite 2502)

CreateTag-Funktion (Seite 2500)

CommitTag-Funktion (Seite 2499)

CloseTag-Funktion (Seite 2498)

VBA im Variablenhaushalt (Seite 2495)

3.5.2.3 VBA im Tag Logging

VBA im Tag Logging

Einleitung

Mit VBA können Sie Prozesswertarchive und Archivvariablen direkt aus dem Programmcode erzeugen, modifizieren und löschen.

Prinzip

Wenn Sie die Instanz der Klasse HMIGO erzeugt haben, stehen Ihnen folgende Funktionen zum Zugriff auf das TagLogging zur Verfügung:

- CloseTlgArchive
- CloseTlgTag
- CommitTlgArchive
- CommitTlgTag
- CreateTlgArchive
- CreateTlgTag
- DeleteTlgArchive
- DeleteTlgTag
- GetTlgArchive
- GetTlgTag

- ListTlgArchive
- ListTlgTag

Hinweis

Das Tag Logging darf während der Bearbeitung mit VBA nicht geöffnet sein oder geöffnet werden.
Bevor Sie die Trigger des Tag Logging in VBA verwenden wollen, müssen Sie bei einem neu erstellten WinCC-Projekt zuerst den Editor "Tag Logging" starten, ein Archiv anlegen und den Editor wieder schließen. Die Trigger werden vom Editor angelegt.

Direktzugriff auf die Objekteigenschaften

Auf die Parameter der oben genannten Funktionen können Sie auch direkt in VBA über folgende Objekteigenschaften zugreifen. In der Spalte "wird verwendet im" sehen Sie, ob Sie auf die Objekteigenschaft im Prozesswertarchiv (P) und/oder im Verdichtungsarchiv (V) zugreifen können.

Objekteigenschaft	Beschreibung	Lesen/ Schreiben	wird verwendet in
ObjectStateTlgArchive	Gibt den Objektstatus über die Enumeration HMIGO_OBJECT_STATE zurück. Weitere Informationen zu dieser Enumeration finden Sie in dieser Dokumentation unter "VBA in weiteren WinCC-Editoren".	ja/nein	P, V
TlgArchiveName	Name des Prozesswert- oder Verdichtungsarchivs.	ja/nein	P, V
TlgArchiveType	Legt fest, ob es sich um ein Prozesswertarchiv oder Verdichtungsarchiv handelt.	ja/nein	P, V
TlgArchiveAccessLevel Read	Die Berechtigungsstufe für Lesen.	ja/nein	P, V
TlgArchiveAccessLevel Write	Die Berechtigungsstufe für Schreiben.	ja/nein	P, V
TlgArchiveArchiveState	Legt fest, ob bei Systemstart die Archivierung gesperrt oder freigegeben ist. Die möglichen Werte der Enum HMIGO_TLG_ARCHIVE_STATE: • TLG_ARCHIVE_STATE_LOCKED (1) • TLG_ARCHIVE_STATE_ACTIVATED (0)	ja/ja	P, V
TlgArchiveBufferSize	Legt die Anzahl der Sätze für ein Umlaufarchiv fest.	ja/ja	P
TlgArchiveBufferType	Legt den Ablageort der Variablen fest. Die möglichen Typen der Enum HMIGO_TLG_ARCHIVE_BUFFER_TYPE: • TLG_ARCHIVE_BUFFER_TYPE_DISK (2) • TLG_ARCHIVE_BUFFER_TYPE_RAM (1)	ja/ja	P

Objekteigenschaft	Beschreibung	Lesen/ Schreiben	wird verwendet in
TlgArchiveCompressRange	Legt den Verdichtungszeitraum fest. Ist der Name des unter "Zeiten" im TagLogging-Editor definierten Timer, der größer gleich 1 Minute ist. Da das Format ein String ist, ist es sprachabhängig. Auch ermittelbar über die Funktion ListTlgArchive(TLG_ARCHIVE_TRIGGER_NAMES, arrTrigger)	ja/ja	V
TlgArchiveCompressType	Legt fest, mit welchem Algorithmus die Werte verdichtet werden. Die möglichen Typen der Enum HMIGO_TLG_ARCHIVE_COMPRESS_TYPE: • TLG_COMPRESS_TYPE_CALC (1) • TLG_COMPRESS_TYPE_CALC_COPY (2) • TLG_COMPRESS_TYPE_CALC_DEL (3) • TLG_COMPRESS_TYPE_CALC_COPY_DEL (4)	ja/ja	V
ObjectStateTlgTag	Gibt den Objektstatus über die Enumeration HMIGO_OBJECT_STATE zurück. Weitere Informationen zu dieser Enumeration finden Sie in dieser Dokumentation unter "VBA in weiteren WinCC-Editoren".	ja/nein	P, V
TlgTagArchiveName	Name der Archivvariable.	ja/nein	P, V
TlgTagName	Name des Archivs.	ja/nein	P, V
TlgTagType	Gibt den Variablentyp Die möglichen Typen der Enum HMIGO_TLG_TAG_TYPE: • TLG_TAG_TYP_ANALOG (65537) • TLG_TAG_TYP_BINARY (65538) • TLG_TAG_TYP_PROCESS (65544) • TLG_TAG_TYP_COMPRESS (65540)	ja/ja	P, V
TlgTagArchiving	Gibt die Erfassungsart an. Die möglichen Werte der Enum HMIGO_TLG_TAG_ARCHIVING: • TLG_TAG_ACYCLIC (8388609) • TLG_TAG_CYCLIC_CONTINUOUS (8388610) • TLG_TAG_CYCLIC_SELECTIVE (8388612) • TLG_TAG_ON_EVERY_CHANGE (8388616)	ja/ja	P

Objekteigenschaft	Beschreibung	Lesen/ Schreiben	wird verwendet in
TlgTagArchivingState	Legt fest, ob die Archivierung beim Systemstart freigegeben oder gesperrt ist. Die möglichen Werte der Enum HMIGO_TLG_TAG_ARCHIVING_STATE: • TLG_TAG_LOCKED (1) • TLG_TAG_ACTIVATED (0)	ja/ja	P, V
TlgTagOnError	Legt fest, ob bei einer Störung der zuletzt erfasste Wert oder der Ersatzwert gespeichert wird. Die möglichen Werte der Enum HMIGO_TLG_TAG_ON_ERROR: • TLG_TAG_LAST_VALUE (1) • TLG_TAG_SUBSTITUTE (2)	ja/ja	P
TlgTagTriggerType	Legt fest, wie die Archivierung bei Signalwechsel erfolgt. Die möglichen Werte der Enum HMIGO_TLG_TAG_TRIGGER_TYPE: • TLG_TAG_FROM_0_TO_1 (2) • TLG_TAG_FROM_1_TO_0 (3) • TLG_TAG_ALWAYS (4) • TLG_TAG_EVERY_CHANGE (1)	ja/ja	P
TlgTagOnChange	Legt fest, ob eine Archivierung bei Änderung erfolgen soll. Die möglichen Werte der Enum HMIGO_TLG_TAG_ON_CHANGE: • TLG_TAG_EVERY_VALUE (0) • TLG_TAG_RELATIVE_HYSTERESE (1) • TLG_TAG_ABSOLUTE_HYSTERESE (2)	ja/ja	P
TlgTagMethodType	Legt fest, nach welcher Methode der Wert vor der Archivierung bearbeitet wird. Die möglichen Werte der Enum HMIGO_TLG_TAG_METHOD_TYPE: • TLG_TAG_ACTUAL (1) • TLG_TAG_SUM (3) • TLG_TAG_MaxValue (5) • TLG_TAG_MinValue (4) • TLG_TAG_AVERAGE (2)	ja/ja	P, V
TlgTagTriggerScan	Name des Timers für den Erfassungszyklus.	ja/ja	P
TlgTagTriggerArchiving	Name des Timers für den Archivierungszyklus.	ja/ja	P
TlgTagStartTriggerFunction	Gibt den Namen einer Script Funktion an, über die auf ein Startereignis für den Beginn der Archivierung geprüft wird.	ja/ja	P
TlgTagStartTriggerModule	Gibt den Namen einer DLL an, aus der die Script Funktion für das Überprüfen eines Startereignisses aufgerufen wird.	ja/ja	P

Objekteigenschaft	Beschreibung	Lesen/ Schreiben	wird verwendet in
TlgTagStopTriggerFunction	Gibt den Namen einer Script Funktion an, über die auf ein Stoppereignis für den Beginn der Archivierung geprüft wird.	ja/ja	P
TlgTagTriggerFunction	Gibt den Namen einer Script Funktion für die dynamische Umschaltung von Erfassungs- und Archivierungszyklus an.	ja/ja	P
TlgTagNameCompressArchive	Enthält bei Verdichtungsarchiven den Namen des Quell-Archivs.	ja/ja	V
TlgTagNameCompressTag	Enthält bei Verdichtungsarchiven den Namen der Quell-Variable.	ja/ja	V
TlgTagNameRawValue	Enthält bei prozessgesteuerten Archiven den Namen der Rohdaten-Variable.	ja/ja	P
TlgTagConvertModule	Name der Normierungs-DLL, die zur Datenkonvertierung verwendet wird.	ja/ja	P
TlgTagNameProcTag	Name der Prozessvariablen, aus welcher der zu erfassende Wert übernommen wird.	ja/ja	P
TlgTagAliasName	Der Name, unter dem die Variable alternativ angesprochen werden kann (Alias).	ja/ja	P
TlgTagStartEvent	Name der Variable, über die der Start der Archivierung geprüft wird.	ja/ja	P
TlgTagStopEvent	Name der Variable, über die der Stopp der Archivierung geprüft wird.	ja/ja	P
TlgTagTriggerFactor	Enthält den Faktor für den Anzeigezyklus als Vielfaches vom Archivierungszyklus.	ja/ja	P
TlgTagUpperLimit	Wert für die Skalierung der oberen Grenze der Variable.	ja/ja	P
TlgTagLowerLimit	Wert für die Skalierung der unteren Grenze der Variable.	ja/ja	P
TlgTagHysterese	Wert für die Hysterese, über die geprüft wird, ob sich ein Wert verändert hat.	ja/ja	P

Siehe auch

ListTlgTag-Funktion (Seite 2533)
ListTlgArchive-Funktion (Seite 2531)
GetTlgArchive-Funktion (Seite 2528)
DeleteTlgTag-Funktion (Seite 2526)
DeleteTlgArchive-Funktion (Seite 2525)
CreateTlgTag-Funktion (Seite 2520)
CreateTlgArchive-Funktion (Seite 2517)
CommitTlgTag-Funktion (Seite 2516)
CommitTlgArchive-Funktion (Seite 2514)
CloseTlgTag-Funktion (Seite 2512)
CloseTlgArchive-Funktion (Seite 2511)
VBA in weiteren WinCC-Editoren (Seite 2493)

CloseTlgArchive-Funktion

Beschreibung

Schließt das geöffnete Prozesswert- oder Verdichtungsarchiv.

Hinweis

Geänderte Parameter werden nicht gespeichert.

Syntax

```
Ausdruck.CloseTlgArchive()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

--

Beispiel

```
Sub CloseTlgArchive()
' HMIGO_006
' procedure to close an archive
```

```
' the archive need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strArchiveName As String
  Set objHMIGO = New HMIGO
  strArchiveName = "NewArchive"
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTlgArchive, vbOKOnly, "Status Tlg Archive"
 'open archive
  objHMIGO.GetTlgArchive strArchiveName
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateTlgArchive, vbOKOnly, "Status Tlg Archive"
 'close archive
  objHMIGO.CloseTlgArchive
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTlgArchive, vbOKOnly, "Status Tlg Archive"

  Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListTlgTag-Funktion (Seite 2533)

ListTlgArchive-Funktion (Seite 2531)

GetTlgArchive-Funktion (Seite 2528)

DeleteTlgTag-Funktion (Seite 2526)

DeleteTlgArchive-Funktion (Seite 2525)

CreateTlgTag-Funktion (Seite 2520)

CreateTlgArchive-Funktion (Seite 2517)

CommitTlgTag-Funktion (Seite 2516)

CommitTlgArchive-Funktion (Seite 2514)

CloseTlgTag-Funktion (Seite 2512)

VBA im Tag Logging (Seite 2506)

CloseTlgTag-Funktion

Beschreibung

Schließt die geöffnete Archivvariable.

Hinweis

Geänderte Parameter werden nicht gespeichert.

Syntax

```
Ausdruck.CloseTlgTag()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

--

Beispiel

```
Sub CloseTlgTag()
' HMIGO_007
' procedure to close a tag logging tag
' the archive need to be created before
' the tag logging tag need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strArchiveName As String
  Dim strTlgTagName As String
  Set objHMIGO = New HMIGO
  strArchiveName = "NewArchive"
  strTlgTagName = "NewTag"

 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTlgTag, vbOKOnly, "Status Tlg Tag"
 'open/ get tag logging tag
  objHMIGO.GetTlgTag strArchiveName, strTlgTagName
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateTlgTag, vbOKOnly, "Status Tlg Archive"
 'close tag logging tag
  objHMIGO.CloseTlgTag
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTlgTag, vbOKOnly, "Status Tlg Archive"
  Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListTlgTag-Funktion (Seite 2533)
ListTlgArchive-Funktion (Seite 2531)
GetTlgArchive-Funktion (Seite 2528)
DeleteTlgTag-Funktion (Seite 2526)
DeleteTlgArchive-Funktion (Seite 2525)
CreateTlgTag-Funktion (Seite 2520)
CreateTlgArchive-Funktion (Seite 2517)
CommitTlgTag-Funktion (Seite 2516)
CommitTlgArchive-Funktion (Seite 2514)
CloseTlgArchive-Funktion (Seite 2511)
VBA im Tag Logging (Seite 2506)

CommitTlgArchive-Funktion

Beschreibung

Schreibt die geänderten Parameter des angegebenen Archivs nach WinCC.

Hinweis

Wenn Sie nach einem CommitTlgArchive-Aufruf weitere Parameter ändern, müssen Sie diese Änderungen mit einem erneuten Aufruf dieser Funktion nach WinCC schreiben.

Syntax

```
Ausdruck.CommitTlgArchive()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

--

Beispiel

```
Sub CommitTlgArchive()
' HMIGO_008
' procedure to change a property of an archive
```

```
' the archive need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strArchiveName As String
  Set objHMIGO = New HMIGO
  strArchiveName = "NewArchive"
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTlgArchive, vbOKOnly, "Status Tlg Archive"
 'open archive
  objHMIGO.GetTlgArchive strArchiveName
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateTlgArchive, vbOKOnly, "Status Tlg Archive"
 'change a property
  objHMIGO.TlgArchiveArchiveState = TLG_STATE_LOCKED
 'current status is "MODIFIED"
  MsgBox objHMIGO.ObjectStateTlgArchive, vbOKOnly, "Status Tlg Archive"
 'commit archive
  objHMIGO.CommitTlgArchive
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateTlgArchive, vbOKOnly, "Status Tlg Archive"

  Set objHMIGO = Nothing
End Sub
```

Siehe auch

DeleteTlgArchive-Funktion (Seite 2525)

ListTlgTag-Funktion (Seite 2533)

ListTlgArchive-Funktion (Seite 2531)

GetTlgArchive-Funktion (Seite 2528)

DeleteTlgTag-Funktion (Seite 2526)

CreateTlgTag-Funktion (Seite 2520)

CreateTlgArchive-Funktion (Seite 2517)

CommitTlgTag-Funktion (Seite 2516)

CloseTlgTag-Funktion (Seite 2512)

CloseTlgArchive-Funktion (Seite 2511)

VBA im Tag Logging (Seite 2506)

CommitTlgTag-Funktion

Beschreibung

Schreibt die geänderten Parameter der angegebenen Archivvariable nach WinCC.

Hinweis

Wenn Sie nach einem CommitTlgTag-Aufruf weitere Parameter ändern, müssen Sie diese Änderungen mit einem erneuten Aufruf dieser Funktion nach WinCC schreiben.

Syntax

```
Ausdruck.CommitTlgTag()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

--

Beispiel

```
Sub CommitTlgTag()
' HMIGO_009
' procedure to change a property of a tag logging tag
' the archive need to be created before
' the tag logging tag need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strArchiveName As String
  Dim strTlgTagName As String
  Set objHMIGO = New HMIGO
  strArchiveName = "NewArchive"
  strTlgTagName = "NewTag"

 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTlgTag, vbOKOnly, "Status Tlg Tag"
 'open/ get tag logging tag
  objHMIGO.GetTlgTag strArchiveName, strTlgTagName
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateTlgTag, vbOKOnly, "Status Tlg Archive"
 'change a property
  objHMIGO.TlgTagArchiving = TLG_TAG_ON_EVERY_CHANGE
 'current status is "MODIFIED"
  MsgBox objHMIGO.ObjectStateTlgTag, vbOKOnly, "Status Tlg Archive"
 'commit tag logging tag
  objHMIGO.CommitTlgTag
```

```
  'current status is "OPENED"
   MsgBox objHMIGO.ObjectStateTlgTag, vbOKOnly, "Status Tlg Archive"
   Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListTlgTag-Funktion (Seite 2533)
ListTlgArchive-Funktion (Seite 2531)
GetTlgArchive-Funktion (Seite 2528)
DeleteTlgTag-Funktion (Seite 2526)
DeleteTlgArchive-Funktion (Seite 2525)
CreateTlgTag-Funktion (Seite 2520)
CreateTlgArchive-Funktion (Seite 2517)
CommitTlgTag-Funktion (Seite 2516)
CommitTlgArchive-Funktion (Seite 2514)
CloseTlgTag-Funktion (Seite 2512)
CloseTlgArchive-Funktion (Seite 2511)
VBA im Tag Logging (Seite 2506)

CreateTlgArchive-Funktion

Beschreibung

Erstellt ein neues Prozesswert- oder Verdichtungsarchiv.

Syntax

```
Ausdruck.CreateTlgArchive(ArchiveName,ArchiveType)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ArchiveName (String)	Name des zu erstellenden Archivs.
ArchiveType (HMIGO_TLG_ARCHIVE_TYPE)	Typ des Archivs. Mögliche Typen sind: • TLG_PROCESSARCHIVE (131073)für ein Prozesswertarchiv • TLG_COMPRESSARCHIVE (131074)für ein Verdichtungsarchiv

Default-Werte beim Erstellen eines neuen Variablen-Archivs

Die folgende Tabelle zeigt die Default-Werte, die beim Erstellen eines neuen Prozesswert- oder Verdichtungsarchives eingetragen werden. Diese Werte können nachträglich geändert und mit der Funktion CommitTlgArchive geschrieben werden.

Eigenschaft	Default-Wert (Enum-Name => Wert)	Bemerkung
TlgArchiveAccessLevelRead	0	Ohne Berechtigungsstufe
TlgArchiveAccessLevelWrite	0	Ohne Berechtigungsstufe
TlgArchiveArchiveState	TLG_ARCHIVE_STATE_ACTIVATED (0)	Die Archivierung ist beim Start von Runtime gestartet.
TlgArchiveBufferSize	1000	Anzahl Datensätze
TlgArchiveBufferType	TLG_ARCHIVE_BUFFER_TYPE_DISK (2)	Die Werte werden in der Datenbank auf Festplatte abgelegt.
TlgArchiveCompressRange	"1 Tag". Dieser String ist für jede Sprache einzeln zu erstellen (z.B. englisch: "1 day")	Entspricht genau einem Tag. Nur relevant bei Verdichtungsvariablen. Besonderheit: der Anwender ist für Wert >= 1 Minute verantwortlich
TlgArchiveCompressType	TLG_COMPRESS_TYPE_CALC (1)	Verdichtungswerte nur berechnen. Nur relevant bei Verdichtungsvariablen.

Enum HMIGO_TLG_ARCHIVE_STATE

Parameter	Beschreibung
TLG_ARCHIVE_STATE_LOCKED (1)	Die Archivierung ist beim Systemstart gesperrt.
TLG_ARCHIVE_STATE_ACTIVATED (0)	Die Archivierung ist beim Start von Runtime gestartet.

Enum HMIGO_TLG_ARCHIVE_BUFFER_TYPE

Parameter	Beschreibung
TLG_ARCHIVE_BUFFER_TYPE_DISK (2)	Die Archivierung der Werte erfolgt auf Festplatte.
TLG_ARCHIVE_BUFFER_TYPE_RAM (1)	Die Archivierung der Werte erfolgt nur im Arbeitsspeicher.

Enum HMIGO_TLG_ARCHIVE_COMPRESS_TYPE

Parameter	Beschreibung
TLG_COMPRESS_TYPE_CALC (1)	Es werden nur die Verdichtungswerte berechnet.
TLG_COMPRESS_TYPE_CALC_COPY (2)	Die Verdichtungswerte werden berechnet und die Originalwerte kopiert.
TLG_COMPRESS_TYPE_CALC_DEL (3)	Die Verdichtungswerte werden berechnet und die Originalwerte anschließend gelöscht.
TLG_COMPRESS_TYPE_CALC_COPY_DEL (4)	Die Verdichtungswerte werden berechnet und die Originalwerte kopiert und anschließend gelöscht.

Beispiel

```
Sub CreateTlgArchive()
' HMIGO_010
' procedure to create an archive
' the archive must not be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strArchiveName As String
  Set objHMIGO = New HMIGO
  strArchiveName = "NewArchive"
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTlgArchive, vbOKOnly, "Status Tlg Archive"
 'create tag logging archive
  objHMIGO.CreateTlgArchive strArchiveName, TLG_PROCESSARCHIVE
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateTlgArchive, vbOKOnly, "Status Tlg Archive"
  Set objHMIGO = Nothing
End Sub
```

Siehe auch

GetTlgArchive-Funktion (Seite 2528)
ListTlgTag-Funktion (Seite 2533)
ListTlgArchive-Funktion (Seite 2531)
DeleteTlgTag-Funktion (Seite 2526)
DeleteTlgArchive-Funktion (Seite 2525)
CreateTlgTag-Funktion (Seite 2520)
CommitTlgTag-Funktion (Seite 2516)
CommitTlgArchive-Funktion (Seite 2514)
CloseTlgTag-Funktion (Seite 2512)
CloseTlgArchive-Funktion (Seite 2511)
VBA im Tag Logging (Seite 2506)

CreateTlgTag-Funktion

Beschreibung

Erstellt eine neue Archivvariable.

Syntax

```
Ausdruck.CreateTlgTag(ArchiveName,TagName,[TagType])
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ArchiveName (String)	Name eines existierenden Archivs, in das die Variable einzutragen ist.
TagName (String)	Name der zu erstellenden Variable.
TagType (HMIGO_TLG_TAG_TYPE, optional)	Legt den Typ der Variable fest. Wird kein Typ angegeben, dann wird der Default-Wert TLG_VAR_TYP_ANALOG eingetragen. Mögliche Typen sind: • TLG_VAR_TYP_ANALOG (65537) für eine analoge Variable. • TLG_VAT_TYP_BINARY (65538) für eine binäre Variable. • TLG_VAR_TYP_PROCESS (65544) für eine Prozess-Variable. • TLG_VAT_TYP_COMPRESS (65540) für eine Verdichtungs-Variable.

Default-Werte beim Erstellen einer neuen Archiv-Variable

Die folgende Tabelle zeigt die Default-Werte, die beim Erstellen einer neuen Archiv-Variable eingetragen werden. Diese Werte können nachträglich geändert und mit der Funktion CommitTlgTag geschrieben werden.

Eigenschaft	Default-Wert (Enum-Name => Wert)	Bemerkung
TlgTagType	TLG_VAR_TYP_ANALOG (65537)	Erfassung über eine analoge Datenmanager-Variable.
TlgTagArchiving	TLG_TAG_CYCLIC_CONTINUOUS (8388610)	Zyklisch kontinuierliche Erfassung.
TlgTagArchivingState	TLG_TAG_ACTIVATED (0)	Die Archivierung ist beim Start von Runtime gestartet.
TlgTagTriggerScan	1 Sekunde	Beachten Sie, dass "1 Sekunde" nur der Name des Triggers ist. Sie müssen selbst dafür sorgen, dass der Trigger existiert und tatsächlich den Zyklus 1s besitzt.
TlgTagTriggerArchiving	1 Sekunde	Beachten Sie, dass "1 Sekunde" nur der Name des Triggers ist. Sie müssen selbst dafür sorgen, dass der Trigger existiert und tatsächlich den Zyklus 1s besitzt.
TlgTagTriggerFactor	1	Anzeigezyklus und Archivierungs-Zyklus sind identisch.
TlgTagOnError	TLG_TAG_LAST_VALUE (1)	Es wird der letzte gültige Wert als Ersatzwert genommen.
TlgTagTriggerType	TLG_TAG_ALWAYS (4)	Es wird jeder Wert archiviert.

Eigenschaft	Default-Wert (Enum-Name => Wert)	Bemerkung
TlgTagMethodType	TLG_TAG_ACTUAL (1)	Keine Bearbeitung. Der Wert wird direkt übernommen.
TlgTagStartTriggerFunction	Keine Funktion angegeben	--
TlgTagStopTriggerFunction	Keine Funktion angegeben	--
TlgTagTriggerFunction	Keine Funktion angegeben	--
TlgTagUpperLimit	Kein Wert angegeben	--
TlgTagLowerLimit	Kein Wert angegeben	--
TlgTagNameCompressArchive	Kein Archivname angegeben	--
TlgTagNameCompressTag	Kein Variablenname angegeben	--
TlgTagNameRawValue	Keine Rohdaten-Variable angegeben	--
TlgTagStartTriggerModule	Kein DLL-Name angegeben	--
TlgTagNameProcTag	entspricht dem "TagName"	--
TlgTagOnChange	TLG_TAG_EVERY_VALUE (0)	Jeder Wert wird archiviert.
TlgTagHysterese	0	Es wird nicht über eine Hysterese geprüft.
TlgTagAliasName	Kein Wert angegeben	--
TlgTagStartEvent	Keine Variable angegeben	--
TlgTagStopEvent	Keine Variable angegeben	--

Liste der Enumeratoren für Tag Logging

Enum-Typen	Beschreibung
TLG_TAG_TYPE	Der übergebene Parameter gibt den Typ der Variable an. Die möglichen Typen stehen in der Tabelle Enum HMIGO_TLG_TAG_TYPE.
TLG_TAG_ARCHIVING	Der übergebene Parameter gibt die Erfassungsart an. Die möglichen Werte stehen in der Tabelle Enum HMIGO_TLG_TAG_ARCHIVING.
TLG_TAG_ARCHIVING_STATE	Der übergebene Parameter gibt an, ob die Archivierung beim Systemstart freigegeben oder gesperrt ist. Die möglichen Werte stehen in der Tabelle Enum HMIGO_TLG_TAG_ARCHIVING_STATE.
TLG_TAG_ON_ERROR	Der übergebene Parameter gibt an, was für ein Wert bei einer Störung gespeichert wird, der zuletzt erfasste oder der Ersatzwert. Die möglichen Werte stehen in der Tabelle Enum HMIGO_TLG_TAG_ON_ERROR.
TLG_TAG_TRIGGER_TYPE	Der übergebene Parameter gibt an, wie die Archivierung bei Signalwechsel erfolgt. Die möglichen Werte stehen in der Tabelle Enum HMIGO_TLG_TAG_TRIGGER_TYPE.

Enum-Typen	Beschreibung
TLG_TAG_METHOD_TYPE	Der übergebene Parameter gibt an, nach welcher Methode der Wert vor der Archivierung bearbeitet wird. Die möglichen Werte stehen in der Tabelle Enum HMIGO_TLG_TAG_METHOD_TYPE.
TLG_TAG_ON_CHANGE	Der übergebene Parameter gibt an, ob eine Archivierung bei Änderung erfolgen soll. Die möglichen Werte stehen in der Tabelle Enum HMIGO_TLG_TAG_ON_CHANGE.

Enum HMIGO_TLG_TAG_TYPE

Werte	Beschreibung
TLG_TAG_TYP_ANALOG (65537)	Analoge Variable
TLG_TAG_TYP_BINARY (65538)	Binäre Variable
TLG_TAG_TYP_PROCESS (65544)	Prozess-Variable
TLG_TAG_TYP_COMPRESS (65540)	Verdichtungsarchiv-Variable

Enum HMIGO_TLG_TAG_ARCHIVING

Werte	Beschreibung
TLG_TAG_ACYCLIC (8388609)	Azyklische Erfassung
TLG_TAG_CYCLIC_CONTINUOUS (8388610)	Zyklisch-kontinuierliche Erfassung
TLG_TAG_CYCLIC_SELECTIVE (8388612)	Zyklisch-selektive Erfassung
TLG_TAG_ON_EVERY_CHANGE (8388616)	Erfassung nur bei Änderung

Enum HMIGO_TLG_TAG_ARCHIVING_STATE

Werte	Beschreibung
TLG_TAG_LOCKED (1)	Erfassung bei Systemstart gesperrt.
TLG_TAG_ACTIVATED (0)	Erfassung bei Systemstart frei gegeben.

Enum HMIGO_TLG_TAG_ON_ERROR

Werte	Beschreibung
TLG_TAG_LAST_VALUE (1)	Der zuletzt gültig erfasste Wert wird verwendet.
TLG_TAG_SUBSTITUTE (2)	Es wird ein Ersatzwert eingetragen.

Enum HMIGO_TLG_TAG_TRIGGER_TYPE

Werte	Beschreibung
TLG_TAG_FROM_0_TO_1 (2)	Signalwechsel vom Wert 0 zu 1.
TLG_TAG_FROM_1_TO_0 (3)	Signalwechsel vom Wert 1 zu 0.
TLG_TAG_ALWAYS (4)	Immer archivieren.
TLG_TAG_EVERY_CHANGE (1)	Bei jedem Signalwechsel archivieren.

Enum HMIGO_TLG_TAG_METHOD_TYPE

Werte	Beschreibung
TLG_TAG_ACTUAL (1)	Der aktuelle Wert wird übernommen.
TLG_TAG_SUM (3)	Die Summe wird gebildet.
TLG_TAG_MaxValue (5)	Der größte Wert wird gespeichert.
TLG_TAG_MinValue (4)	Der kleinste Wert wird gespeichert.
TLG_TAG_AVERAGE (2)	Der Mittelwert wird gespeichert.

Enum HMIGO_TLG_TAG_ON_CHANGE

Werte	Beschreibung
TLG_TAG_EVERY_VALUE (0)	Der aktuelle Wert wird übernommen.
TLG_TAG_RELATIVE_HYSTERESE (1)	Es wird eine prozentual angegebene Hysterese für die Berechnung verwendet, ob der Wert zu archivieren ist.
TLG_TAG_ABSOLUTE_HYSTERESE (2)	Es wird eine absolut angegebene Hysterese für die Berechnung verwendet, ob der Wert zu archivieren ist.

Beispiel

```
Sub CreateTlgTag()
' HMIGO_011
' procedure to create a tag logging tag
' the archive need to be created before
' the tag logging tag must not be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strArchiveName As String
  Dim strTlgTagName As String
  Set objHMIGO = New HMIGO
  strArchiveName = "NewArchive"
  strTlgTagName = "NewTag"

 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTlgTag, vbOKOnly, "Status Tlg Tag"
 'create tag logging tag
```

```
  objHMIGO.CreateTlgTag strArchiveName, strTlgTagName, TLG_TAG_TYPE_ANALOG
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateTlgTag, vbOKOnly, "Status Tlg Archive"
  Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListTlgTag-Funktion (Seite 2533)
ListTlgArchive-Funktion (Seite 2531)
GetTlgArchive-Funktion (Seite 2528)
DeleteTlgTag-Funktion (Seite 2526)
DeleteTlgArchive-Funktion (Seite 2525)
CreateTlgArchive-Funktion (Seite 2517)
CommitTlgTag-Funktion (Seite 2516)
CloseTlgArchive-Funktion (Seite 2511)
CloseTlgTag-Funktion (Seite 2512)
VBA im Tag Logging (Seite 2506)

DeleteTlgArchive-Funktion

Beschreibung

Löscht das angegebene Archiv.

Syntax

```
Ausdruck.DeleteTlgArchive(ArchiveName)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ArchiveName (String)	Name des Archivs, das gelöscht werden soll. Im Archiv enthaltene Archivvariablen werden ebenfalls gelöscht.

Beispiel

```
Sub DeleteTlgArchive()
' HMIGO_012
' procedure to delete an archive
' the archive need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strArchiveName As String
  Set objHMIGO = New HMIGO
  strArchiveName = "NewArchive"
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTlgArchive, vbOKOnly, "Status Tlg Archive"

 'delete tag logging archive
  objHMIGO.DeleteTlgArchive strArchiveName
  Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListTlgTag-Funktion (Seite 2533)
ListTlgArchive-Funktion (Seite 2531)
GetTlgArchive-Funktion (Seite 2528)
DeleteTlgTag-Funktion (Seite 2526)
CreateTlgTag-Funktion (Seite 2520)
CreateTlgArchive-Funktion (Seite 2517)
CommitTlgTag-Funktion (Seite 2516)
CommitTlgArchive-Funktion (Seite 2514)
CloseTlgTag-Funktion (Seite 2512)
CloseTlgArchive-Funktion (Seite 2511)
VBA im Tag Logging (Seite 2506)

DeleteTlgTag-Funktion

Beschreibung

Löscht die angegebene Archivvariable.

Syntax

```
Ausdruck.DeleteTlgTag(ArchiveName,TagName)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ArchiveName (String)	Name des Archivs, in dem die zu löschende Archivvariable liegt.
TagName (String)	Name der zu löschende Archivvariable.

Beispiel

```
Sub DeleteTlgTag()
' HMIGO_013
' procedure to delete a tag logging tag
' the archive need to be created before
' the tag logging tag need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strArchiveName As String
  Dim strTlgTagName As String
  Set objHMIGO = New HMIGO
  strArchiveName = "NewArchive"
  strTlgTagName = "NewTag"

 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTlgTag, vbOKOnly, "Status Tlg Tag"
 'delete tag logging tag
  objHMIGO.DeleteTlgTag strArchiveName, strTlgTagName
  Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListTlgTag-Funktion (Seite 2533)

ListTlgArchive-Funktion (Seite 2531)

GetTlgArchive-Funktion (Seite 2528)

DeleteTlgArchive-Funktion (Seite 2525)

CreateTlgTag-Funktion (Seite 2520)

CreateTlgArchive-Funktion (Seite 2517)

CommitTlgTag-Funktion (Seite 2516)

CommitTlgArchive-Funktion (Seite 2514)

CloseTlgTag-Funktion (Seite 2512)

CloseTlgArchive-Funktion (Seite 2511)

VBA im Tag Logging (Seite 2506)

GetTlgArchive-Funktion

Beschreibung

Liest die Parameter des angegebenen Archivs ein.

Die Parameter können Sie über die Objekteigenschaften ändern oder lesen. Eine Auflistung der verfügbaren Objekteigenschaften finden Sie in dieser Dokumentation unter "VBA im TagLogging".

Syntax

```
Ausdruck.GetTlgArchive(ArchiveName)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ArchiveName (String)	Name des Archivs, dessen Werte eingelesen werden sollen.

Beispiel

```
Sub GetTlgArchive()
' HMIGO_014
' procedure to open an archive
```

```
' the archive need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strArchiveName As String
  Set objHMIGO = New HMIGO
  strArchiveName = "NewArchive"
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTlgArchive, vbOKOnly, "Status Tlg Archive"
 'open/ get tag logging archive
  objHMIGO.GetTlgArchive strArchiveName
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateTlgArchive, vbOKOnly, "Status Tlg Archive"
  Set objHMIGO = Nothing
End Sub
```

Siehe auch

CreateTlgTag-Funktion (Seite 2520)

ListTlgTag-Funktion (Seite 2533)

ListTlgArchive-Funktion (Seite 2531)

GetTlgArchive-Funktion (Seite 2528)

DeleteTlgTag-Funktion (Seite 2526)

DeleteTlgArchive-Funktion (Seite 2525)

CreateTlgArchive-Funktion (Seite 2517)

CommitTlgTag-Funktion (Seite 2516)

CommitTlgArchive-Funktion (Seite 2514)

CloseTlgTag-Funktion (Seite 2512)

CloseTlgArchive-Funktion (Seite 2511)

VBA im Tag Logging (Seite 2506)

GetTlgTag-Funktion

Beschreibung

Liest die Parameter der angegebenen Archivvariable ein.

Die Parameter können Sie über die Objekteigenschaften ändern oder lesen. Eine Auflistung der verfügbaren Objekteigenschaften finden Sie in dieser Dokumentation unter "VBA im TagLogging".

Syntax

```
Ausdruck.GetTlgTag(ArchiveName,TagName)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ArchiveName (String)	Name des Archivs, in dem die Archivvariable enthalten ist.
TagName	Name der Archivvariable, deren Parameter eingelesen werden sollen.

Beispiel

```
Sub GetTlgTag()
' HMIGO_015
' procedure to open a tag logging tag
' the archive need to be created before
' the tag logging need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim strArchiveName As String
  Dim strTlgTagName As String
  Set objHMIGO = New HMIGO
  strArchiveName = "NewArchive"
  strTlgTagName = "NewTag"

 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateTlgTag, vbOKOnly, "Status Tlg Tag"
 'open/ get tag logging tag
  objHMIGO.GetTlgTag strArchiveName, strTlgTagName
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateTlgTag, vbOKOnly, "Status Tlg Archive"
  Set objHMIGO = Nothing
End Sub
```

Siehe auch

CreateTlgTag-Funktion (Seite 2520)

ListTlgTag-Funktion (Seite 2533)

ListTlgArchive-Funktion (Seite 2531)

GetTlgArchive-Funktion (Seite 2528)

DeleteTlgTag-Funktion (Seite 2526)

DeleteTlgArchive-Funktion (Seite 2525)

CreateTlgArchive-Funktion (Seite 2517)

CommitTlgTag-Funktion (Seite 2516)

CommitTlgArchive-Funktion (Seite 2514)

CloseTlgTag-Funktion (Seite 2512)

CloseTlgArchive-Funktion (Seite 2511)

VBA im Tag Logging (Seite 2506)

ListTlgArchive-Funktion

Beschreibung

Die ListTlgArchive-Funktion gibt alternativ folgende Werte des Tag Logging in einer Liste zurück:

- alle vorhandenen Tag Logging Archive
- alle vorhandenen Zyklen / Timer

Syntax

```
Ausdruck.ListTlgArchive(ListType,pListArray,[Filter])
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ListType (HMIGO_TLG_ARCHIVE_LIST_TYPE)	Legt fest, welcher Inhalt in einer Liste zurückgegeben wird. Möglich ist: • TLG_ARCHIVE_NAMES (1) alle erstellten Tag Logging erstellten Archive • TLG_ARCHIVE_TRIGGER_NAMES (2) alle erstellten Zyklen / Timer
pListArray (Variant)	Liste mit dem angeforderten Inhalt.
Filter (String)	Optional können Filter gesetzt werden. Als Filter kann jeweils ein Triggernamen verwendet werden. Die Wildcards "*" und "?" werden unterstützt.

Beispiel

Im folgenden Beispiel wird überprüft, ob Archive projektiert sind:

```
Sub ReadTlgArchives()
'HMIGO_028
'read content in tag logging
'no archives are implemented
  Dim objHMIGO As New HMIGO
  Dim arrContent As Variant
'read all tlg archives
  objHMIGO.ListTlgArchive TLG_ARCHIVE_NAMES, arrContent
'check result
  If (UBound(arrContent) - LBound(arrContent) + 1) <= 0 Then
    MsgBox "no entries because no tag logging archives are implemented"
  End If
End Sub
```

Siehe auch

ListTlgTag-Funktion (Seite 2533)

GetTlgArchive-Funktion (Seite 2528)

DeleteTlgTag-Funktion (Seite 2526)

DeleteTlgArchive-Funktion (Seite 2525)

CreateTlgTag-Funktion (Seite 2520)

CreateTlgArchive-Funktion (Seite 2517)

CommitTlgTag-Funktion (Seite 2516)

CommitTlgArchive-Funktion (Seite 2514)

CloseTlgTag-Funktion (Seite 2512)

CloseTlgArchive-Funktion (Seite 2511)

VBA im Tag Logging (Seite 2506)

ListTlgTag-Funktion

Beschreibung

Die ListTlgTag-Funktion gibt alle erstellten Variablen eines Archivs oder aller Archive des Tag Logging in einer Liste zurück.

Syntax

```
Ausdruck.ListTlgTag(ListType,ListArray,[ArchiveName],[Filter])
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ListType (HMIGO_TLG_TAG_LIST_TYPE)	Legt fest, welcher Inhalt in einer Liste zurückgegeben wird. Möglich ist: TLG_TG_NAMES (1) alle in einem Archiv des Tag Logging erstellten Variablen
ListArray (Variant)	Liste mit dem angeforderten Inhalt.
ArchiveName (String)	Name des Archivs im Tag Logging (optional). Wenn der Name des Archivs nicht angegeben ist, werden die Variablen aller Archive zurückgegeben.
Filter (String)	Optional können Filter gesetzt werden. Die Wildcards "*" und "?" werden unterstützt.

Beispiel

Im folgenden Beispiel wird überprüft, ob Archivvariablen im Archiv "processarchive" projektiert sind:

```
Sub ReadTlgTag()
'HMIGO_029
'read content in tag logging
'no tags within archives are implemented
  Dim objHMIGO As New HMIGO
  Dim arrContent As Variant
  Dim strArchive as String
'set tlg archive name
  strArchive = "processarchive"
'read all tlg tags in specified archive
  objHMIGO.ListTlgTag TLG_TAG_NAMES, arrContent, strArchive
'check result
  If (UBound(arrContent) - LBound(arrContent) + 1) <= 0 Then
    MsgBox "no entries because no tag logging tags in specified archive are implemented"
  End If
End Sub
```

Siehe auch

ListTlgArchive-Funktion (Seite 2531)

GetTlgArchive-Funktion (Seite 2528)

DeleteTlgTag-Funktion (Seite 2526)

DeleteTlgArchive-Funktion (Seite 2525)

CreateTlgTag-Funktion (Seite 2520)

CreateTlgArchive-Funktion (Seite 2517)

CommitTlgTag-Funktion (Seite 2516)

CommitTlgArchive-Funktion (Seite 2514)

CloseTlgTag-Funktion (Seite 2512)

CloseTlgArchive-Funktion (Seite 2511)

VBA im Tag Logging (Seite 2506)

3.5.2.4 VBA in der Text Library

VBA in der Text Library

Einleitung

Mit VBA können Sie Texte der Text Library direkt aus dem Programmcode erzeugen, modifizieren und löschen sowie TextIDs und Texte auslesen.

Prinzip

Wenn Sie die Instanz der Klasse HMIGO erzeugt haben, stehen Ihnen folgende Funktionen zum Zugriff auf die TextLibrary zur Verfügung:

- CreateTextLanguage
- CreateText
- DeleteText
- DeleteTextLanguage
- GetText
- GetTextID
- ListText
- ModifyText

Für die Parameterversorgung dieser Funktionen steht die Enumeration "HMIGO_TEXT_CREATE_MODE" und "HMIGO_TEXT_LIST_TYPE" zur Verfügung.

Hinweis

Die TextLibrary darf während der Bearbeitung mit VBA nicht geöffnet sein oder geöffnet werden.

Siehe auch

ModifyText-Funktion (Seite 2546)

ListText-Funktion (Seite 2544)

GetTextID-Funktion (Seite 2542)

GetText-Funktion (Seite 2541)

DeleteTextLanguage-Funktion (Seite 2540)

DeleteText-Funktion (Seite 2538)

CreateText-Funktion (Seite 2537)

CreateTextLanguage-Funktion (Seite 2535)

VBA in weiteren WinCC-Editoren (Seite 2493)

CreateTextLanguage-Funktion

Beschreibung

Erstellt eine Sprache in der Text Library.

Syntax

```
Ausdruck.CreateTextLanguage(LanguageID)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
LanguageID (Long)	ID für die zu erstellende Sprache, z.B. 1031 für Deutsch, 1033 für Englisch etc. Eine Tabelle aller Sprachcodes finden Sie in der WinCC-Onlinehilfe unter dem Stichwort "Sprachkennungen".

Beispiel

```
Sub CreateTextLanguage()
' HMIGO_016
' procedure to create a language in text library
' language must not be created before
' LanguageID german = 1031
' LanguageID english(US) = 1033
' LanguageID spanish = 1034
' LanguageID french = 1040
' LanguageID farsi= 1065
' declarations
  Dim objHMIGO As HMIGO
  Dim lngLangugeNumber As Long
  Set objHMIGO = New HMIGO
  lngLangugeNumber = 1065     'farsi
 'create new language
  objHMIGO.CreateTextLanguage lngLangugeNumber
  Set objHMIGO = Nothing
End Sub
```

Siehe auch

ModifyText-Funktion (Seite 2546)

ListText-Funktion (Seite 2544)

GetTextID-Funktion (Seite 2542)

GetText-Funktion (Seite 2541)

DeleteTextLanguage-Funktion (Seite 2540)

DeleteText-Funktion (Seite 2538)

CreateText-Funktion (Seite 2537)

VBA in der Text Library (Seite 2534)

CreateText-Funktion

Beschreibung

Erstellt einen neuen Text für die angegebene Sprache. Texteinträge für andere Sprachen können mit ModifyText ergänzt werden.

Syntax

```
Ausdruck.CreateText(LanguageID,Text,CreateMode,TextID)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
LanguageID (Long)	ID der Sprache, für die der Text erstellt wird.
Text (String)	Neu zu erstellender Text.
CreateMode (HMIGO_TEXT_CREATE_MODE)	Modus der Texterstellung: • TEXT_ADD_REFCOUNT (0) erhöht lediglich den Referenz-Zähler, wenn bereits ein identischer Text vorliegt. • TEXT_CREATE_ALWAYS (1) legt immer eine neue Textzeile an und fügt den Text dort ein.
TextID (Long)	Gibt die TextID zurück, die der neu erstellte Text erhält bzw. die TextID, deren Referenz-Zähler erhöht wird. Diese ID wird benötigt zur Bearbeitung des Textes in anderen Funktionen.

Beispiel

```
Sub CreateText()
' HMIGO_017
' procedure to create a new text
' declarations
  Dim objHMIGO As HMIGO
  Dim lngLanguageID As Long
  Dim lngTextCreateMode As Long
  Dim lngTextID As Long         'return value of ".CreateText"
  Dim strText As String
  Set objHMIGO = New HMIGO
    strText = "new text"
  'LanguageID = english
  lngLanguageID = 1033
  '"TEXT_ADD_REFCOUNT" check if text exists, if not create new text
  lngTextCreateMode = 0
```

```
   '"TEXT_CREATE_ALWAYS" create always a new text (for messages)
'   lngTextCreateMode = 1

  'create new text
   objHMIGO.CreateText lngLanguageID, strText, lngTextCreateMode, lngTextID
  'show TextID of created text
   MsgBox "TextID: " & lngTextID, vbOKOnly, "Result CreateText"
   Set objHMIGO = Nothing
End Sub
```

Siehe auch

ModifyText-Funktion (Seite 2546)

ListText-Funktion (Seite 2544)

GetTextID-Funktion (Seite 2542)

GetText-Funktion (Seite 2541)

DeleteTextLanguage-Funktion (Seite 2540)

DeleteText-Funktion (Seite 2538)

CreateTextLanguage-Funktion (Seite 2535)

VBA in der Text Library (Seite 2534)

DeleteText-Funktion

Beschreibung

Löscht eine Textzeile. Es werden alle Sprachen für die entsprechende Textzeile sowie die Textzeile selbst gelöscht.

Syntax

```
Ausdruck.DeleteText(TextID)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
TextID (Long)	ID der Textzeile, die gelöscht werden soll.

Beispiel

```
Sub DeleteText()
' HMIGO_018
' procedure to delete a text
' text will be searched and deleted
' declarations
  Dim objHMIGO As HMIGO
  Dim lngLanguageID As Long
  Dim lngTextID As Long         'return value of GetTextID
  Dim strText As String
On Error GoTo ErrorHandler
  Set objHMIGO = New HMIGO
  strText = "new text"
  lngLanguageID = 1033

 'first: find text in text library and return TextID
  objHMIGO.GetTextID 1033, strText, lngTextID

 'if searched text exists: delete this text
  If Not lngTextID = -1 Then
    objHMIGO.DeleteText lngTextID
    MsgBox "Text : """ & strText & """ found in TextID: " & lngTextID & vbNewLine & _
           "TextID is deleted!", vbOKOnly, "Result DeleteText"
  Else
    MsgBox "Text : """ & strText & """ not found." & vbNewLine & _
           "No Text deleted!", vbOKOnly, "Result DeleteText"
  End If
  Set objHMIGO = Nothing
  Exit Sub
ErrorHandler:
 'if lngText = (-1),searched text does not exit
  If lngTextID = -1 Then
    'reset errorhandler
    Err.Clear
    Resume Next
  End If
  MsgBox "ErrNr. : " & Err.Number & vbNewLine & _
         "ErrDes.: " & Err.Description, vbOKOnly, "Error ocurred"
  'reset errorhandler
  Err.Clear
End Sub
```

Siehe auch

VBA in der Text Library (Seite 2534)

ModifyText-Funktion (Seite 2546)

ListText-Funktion (Seite 2544)

GetTextID-Funktion (Seite 2542)

GetText-Funktion (Seite 2541)

DeleteTextLanguage-Funktion (Seite 2540)

CreateText-Funktion (Seite 2537)

CreateTextLanguage-Funktion (Seite 2535)

DeleteTextLanguage-Funktion

Beschreibung

Ermöglicht das Löschen einer Sprache in der TextLibrary. Dabei werden alle Texte dieser Sprachen ebenfalls gelöscht.

Syntax

```
Ausdruck.DeleteTextLanguage(LanguageID)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
LanguageID (Long)	ID der Sprache, die gelöscht werden soll.

Beispiel

Im folgenden Beispiel wird überprüft, ob die Sprache '1036' existiert. Wenn ja wird diese Sprache gelöscht.

```
Sub DeleteLanguage()
'HMIGO_030
' delete an existing languages in TextLibrary
' language '1036'/spanish has to exist
  Dim objHMIGO As New HMIGO
  Dim arrContent As Variant
  Dim intLanguage As Long
  Dim lngPointer As Long
```

```
' get all existing languages
  objHMIGO.ListText TEXT_LANGUAGE_IDS, arrContent
' check requested list for language '1036'/ spanish and delete
  For lngPointer = LBound(arrContent) To UBound(arrContent)
    intLanguage = arrContent(lngPointer) + Val("&H400")
    If intLanguage = 1036 Then
     'delete language
      objHMIGO.DeleteTextLanguage intLanguage
    End If
  Next lngPointer
End Sub
```

Siehe auch

GetText-Funktion (Seite 2541)

ModifyText-Funktion (Seite 2546)

ListText-Funktion (Seite 2544)

GetTextID-Funktion (Seite 2542)

DeleteText-Funktion (Seite 2538)

CreateText-Funktion (Seite 2537)

CreateTextLanguage-Funktion (Seite 2535)

VBA in der Text Library (Seite 2534)

GetText-Funktion

Beschreibung

Gibt den Text für die gewählte TextID in der gewählten Sprache zurück.

Syntax

```
Ausdruck.GetText(LanguageID,TextID,Text)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGeneralObjects" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
LanguageID (Long)	ID der Sprache des zu lesenden Textes.
TextID (Long)	ID der Textzeile aus der Text gelesen werden soll.
Text (String)	Gibt den Text der gewählten Textzeile und Sprache zurück.

Beispiel

```
Sub GetText()
' HMIGO_019
' procedure to get a text
' text with TextID = '69' need to be created
' declarations
  Dim objHMIGO As HMIGO
  Dim lngLanguageID As Long
  Dim lngTextID As Long
  Dim strText As String         'return value of GetText
  Set objHMIGO = New HMIGO
  lngTextID = 69
  lngLanguageID = 1033

 'find text text library
  objHMIGO.GetText lngLanguageID, lngTextID, strText

 'show found text
  MsgBox "Read Text in TextID : " & lngTextID & " is """ & strText & """ !", _
                              vbOKOnly, "Result GetText"
  Set objHMIGO = Nothing
End Sub
```

Siehe auch

ModifyText-Funktion (Seite 2546)

ListText-Funktion (Seite 2544)

GetTextID-Funktion (Seite 2542)

DeleteTextLanguage-Funktion (Seite 2540)

DeleteText-Funktion (Seite 2538)

CreateText-Funktion (Seite 2537)

CreateTextLanguage-Funktion (Seite 2535)

VBA in der Text Library (Seite 2534)

GetTextID-Funktion

Beschreibung

Gibt die ID für den gesuchten Text in der gewählten Sprache zurück.

Gibt es mehrere Texte mit gleichem Inhalt, dann wird nur die Textzeile mit der niedrigsten ID zurückgeliefert. Ob es mehrere Textzeilen mit gleichem Text gibt, hängt vom CreateMode der CreateText-Funktion ab.

Syntax

```
Ausdruck.GetTextID(LanguageID,Text,TextID)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
LanguageID (Long)	ID der Sprache des gesuchten Textes.
Text (String)	Der gesuchte Text.
TextID (Long)	ID der Textzeile, in der der gesuchte Text gefunden wurde.

Beispiel

```
Sub GetTextID()
' HMIGO_020
' procedure to search a TextID
' text will be searched and a TextID will be returned
' declarations
  Dim objHMIGO As HMIGO
  Dim lngLanguageID As Long
  Dim lngTextID As Long         'return value of GetTextID
  Dim strText As String
On Error GoTo ErrorHandler
  Set objHMIGO = New HMIGO
  strText = "old text"
  lngLanguageID = 1033

 'first: find text in text library and return TextID
  objHMIGO.GetTextID 1033, strText, lngTextID

 'if searched text exists: delete this text
  If Not lngTextID = -1 Then
    MsgBox "Text : """ & strText & """ found in TextID: " & lngTextID, _
                  vbOKOnly, "Result GetTextID"
  Else
    MsgBox "Text : """ & strText & """ not found!", vbOKOnly, "Result GetTextID"
  End If
  Set objHMIGO = Nothing
  Exit Sub
ErrorHandler:
 'if lngText = (-1),searched text does not exit
  If lngTextID = -1 Then
    'reset errorhandler
    Err.Clear
    Resume Next
  End If
```

```
  MsgBox "ErrNr. : " & Err.Number & vbNewLine & _
         "ErrDes.: " & Err.Description, vbOKOnly, "Error ocurred"
  'reset errorhandler
  Err.Clear
End Sub
```

Siehe auch

ModifyText-Funktion (Seite 2546)

ListText-Funktion (Seite 2544)

GetText-Funktion (Seite 2541)

DeleteTextLanguage-Funktion (Seite 2540)

DeleteText-Funktion (Seite 2538)

CreateText-Funktion (Seite 2537)

CreateTextLanguage-Funktion (Seite 2535)

VBA in der Text Library (Seite 2534)

ListText-Funktion

Beschreibung

Die ListText-Funktion gibt alternativ folgende Inhalte der TextLibrary als Liste zurück:

- alle erstellten Sprachen
- alle ID der Texte
- den gesamten Text einer Sprache

Syntax

```
Ausdruck.ListText(ListType,pListArray,[LanguageID],[Filter])
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ListType (HMIGO_TEXT_LIST_TYPE)	Legt fest, welcher Inhalt als Liste zurückgegeben wird. Möglich ist: • TEXT_LANGUAGE_IDS (1) alle erstellten Sprachen. Das Ergebnis muss noch konvertiert werden, indem 400hex addiert wird. • TEXT_IDS (2) die ID aller Texte. • TEXT_TEXTS (3) alle Texte einer Sprache.
pListArray (Variant)	Liste mit dem angeforderten Inhalt.
LanguageID (Long)	ID der Sprache, dessen Text zurückgegeben wird.
Filter (String)	Optional können Filter verwendet werden. Die Wildcard "*" und "?" werden unterstützt.

Beispiel

Im folgenden Beispiel wird überprüft, ob die Liste mit dem Text einer Sprache leer ist, da die Sprache nicht existiert:

```
Sub ReadTextsByLanguage()
'HMIGO_031
'read content in textLibrary by language
  Dim objHMIGO As New HMIGO
  Dim arrContent As Variant
  Dim intLanguage As Integer
'set invalid language ID
  intLanguage = 1051 'language does not exist
'read all texts
  objHMIGO.ListText TEXT_TEXTS, arrContent, intLanguage
'check result
  If (UBound(arrContent) - LBound(arrContent) + 1) <= 0 Then
    MsgBox "no entries because wrong language selection"
  End If
End Sub
```

Siehe auch

ModifyText-Funktion (Seite 2546)

GetTextID-Funktion (Seite 2542)

GetText-Funktion (Seite 2541)

DeleteTextLanguage-Funktion (Seite 2540)

DeleteText-Funktion (Seite 2538)

CreateTextLanguage-Funktion (Seite 2535)

VBA in der Text Library (Seite 2534)

ModifyText-Funktion

Beschreibung

Ändert den Text für die gewählte Sprache mit der angegebenen ID.

Syntax

```
Ausdruck.ModifyText(LanguageID,TextID,Text)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
LanguageID (Long)	ID der Sprache des zu ändernden Textes.
TextID (Long)	ID der Textzeile des zu ändernden Textes.
Text (String)	Neuer Text, der eingefügt werden soll.

Beispiel

```
Sub ModifyText()
' HMIGO_021
' procedure to modify a text
' text will be searched and replaced
' declarations
  Dim objHMIGO As HMIGO
  Dim lngLanguageID As Long
  Dim lngTextID As Long         'return value of GetTextID
  Dim strOldText As String
  Dim strNewText As String
On Error GoTo ErrorHandler
  Set objHMIGO = New HMIGO
  strOldText = "old text"
  strNewText = "new text"
  lngLanguageID = 1033

 'first: find text in text library and return TextID
  objHMIGO.GetTextID 1033, strOldText, lngTextID

 'if searched text exists: replace this text
  If Not lngTextID = -1 Then
    objHMIGO.ModifyText lngLanguageID, lngTextID, strNewText
    MsgBox "Text : """ & strOldText & """ found in TextID: " & lngTextID & vbNewLine & _
           "Text replaced with : """ & strNewText & """ !", vbOKOnly, "Result DeleteText"
  Else
    MsgBox "Text : """ & strOldText & """ not found." & vbNewLine & _
```

```
          "No Replacements done!", vbOKOnly, "Result DeleteText"
  End If
  Set objHMIGO = Nothing
  Exit Sub
ErrorHandler:
 'if lngText = (-1),searched text does not exit
  If lngTextID = -1 Then
    'reset errorhandler
    Err.Clear
    Resume Next
  End If
  MsgBox "ErrNr. : " & Err.Number & vbNewLine & _
         "ErrDes.: " & Err.Description, vbOKOnly, "Error ocurred"
  'reset errorhandler
  Err.Clear
End Sub
```

Siehe auch

ListText-Funktion (Seite 2544)

GetTextID-Funktion (Seite 2542)

GetText-Funktion (Seite 2541)

DeleteTextLanguage-Funktion (Seite 2540)

DeleteText-Funktion (Seite 2538)

CreateText-Funktion (Seite 2537)

CreateTextLanguage-Funktion (Seite 2535)

VBA in der Text Library (Seite 2534)

3.5.2.5 VBA im Alarm Logging

VBA im Alarm Logging

Einleitung

Mit VBA können Sie Meldungen direkt aus dem Programmcode erzeugen, modifizieren und löschen.

Prinzip

Wenn Sie die Instanz der Klasse HMIGO erzeugt haben, stehen Ihnen folgende Funktionen zum Zugriff auf das Alarm Logging zur Verfügung:

- CloseSingleAlarm
- CommitSingleAlarm
- CreateSingleAlarm

- DeleteSingleAlarm
- GetSingleAlarm
- ListSingleAlarm

Für die Parameterversorgung dieser Funktionen steht die Enumeration "HMIGO_SINGLE_ALARM_CLASS_IDS" und "HMIGO_SINGLE_ALARM_LIST_TYPE" zur Verfügung.

Hinweis

Das Alarm Logging darf während der Bearbeitung mit VBA nicht geöffnet sein oder geöffnet werden.

Zugriff auf die Objekteigenschaften

Auf die Parameter der oben genannten Funktionen können Sie auch direkt in VBA über folgende Objekteigenschaften zugreifen:

Objekteigenschaft	Beschreibung	Lesen/ Schreiben
ObjectStateSingleAlarm	Gibt den Objektstatus über die Enumeration HMIGO_OBJECT_STATE zurück. Weitere Informationen zu dieser Enumeration finden Sie in dieser Dokumentation unter "VBA in weiteren WinCC-Editoren".	ja/nein
SingleAlarmMessageNumber	Nummer der Meldung	ja/nein
SingleAlarmAGNumber	AG-Nummer	ja/ja
SingleAlarmCPUNumber	CPU-Nummer des AGs.	ja/ja

Objekteigenschaft	Beschreibung	Lesen/ Schreiben
SingleAlarmClassID	Meldeklasse der Meldung. Die möglichen Werte der Enum SINGLE_ALARM_CLASS_IDS: • SINGLE_ALARM_ERROR (1) • SINGLE_ALARM_CLASS_2 (2) • SINGLE_ALARM_CLASS_3 (3) • SINGLE_ALARM_CLASS_4 (4) • SINGLE_ALARM_CLASS_5 (5) • SINGLE_ALARM_CLASS_6 (6) • SINGLE_ALARM_CLASS_7 (7) • SINGLE_ALARM_CLASS_8 (8) • SINGLE_ALARM_CLASS_9 (9) • SINGLE_ALARM_CLASS_10 (10) • SINGLE_ALARM_CLASS_11 (11) • SINGLE_ALARM_CLASS_12 (12) • SINGLE_ALARM_CLASS_13 (13) • SINGLE_ALARM_CLASS_14 (14) • SINGLE_ALARM_CLASS_15 (15) • SINGLE_ALARM_CLASS_16 (16) • SINGLE_ALARM_CLASS_SYSTEM_REQUIRE_ACKNOWLEDGEMENT (17) • SINGLE_ALARM_CLASS_SYSTEM_WITHOUT_ACKNOWLEDGEMENT (18)	ja/ja
SingleAlarmMessageTypeID	Typ ID der Meldung. Die erlaubten Werte sind von der Meldeklasse abhängig: • Klasse 1: Werte von 1 bis 16 • Klasse 2: Werte von 17 bis 32 • Klasse 3: Werte von 33 bis 48 • ... • Klasse 18: Werte von 263 bis 288	ja/ja
SingleAlarmTextXXID XX = 1...10	Für die Anwendertexte 1 bis 10 gibt es die Eigenschaften SingleAlarmText1ID bis SingleAlarmText10ID	ja/ja
SingleAlarmTagNameProcessValueXX XX = 1...10	Für die zu archivierenden Prozesswerte gibt es die Eigenschaften SingleAlarmTagNameProcessValue1 bis 10 Wenn Sie einen projektierten Prozesswert löschen möchten, dann müssen Sie diesen Parameter mit einer Variable vom Typ "Long" beschreiben, die den Wert "0" besitzt.[1)]	ja/ja
SingleAlarmTagName	Variablenname für Ereignis	ja/ja
SingleAlarmMessageBit	Bits für Bitmeldeverfahren	ja/ja
SingleAlarmQuitTag	Variablenname für Quittierstatus	ja/ja
SingleAlarmQuitBits	Bit für Bitmeldeverfahren	ja/ja
SingleAlarmStateTag	Variable für Statusabfrage	ja/ja
SingleAlarmStateBits	Bit für Statusvariable	ja/ja
SingleAlarmNormDLL	Name der Normierungs DLL	ja/ja
SingleAlarmQuitSingle	Quittierung der Meldungen, TRUE oder FALSE möglich	ja/ja

Objekteigenschaft	Beschreibung	Lesen/ Schreiben
SingleAlarmHornActivate	Aktivierung des Signalgebers, TRUE oder FALSE möglich	ja/ja
SingleAlarmArchiving	Archivierung der Meldung, TRUE oder FALSE möglich	ja/ja
SingleAlarmProtocol	Protokollierung der Meldung, TRUE oder FALSE möglich	ja/ja
SingleAlarmFlankInvert	Meldung bei fallender Flanke auslösen, TRUE oder FALSE möglich	ja/ja
SingleAlarmLockedOnStart	Meldung ist bei Hochlauf des Systems gesperrt, TRUE oder FALSE möglich	ja/ja
SingleAlarmGlobalAPFunction	Meldung an globale AP-Funktion weitergeben, TRUE oder FALSE möglich	ja/ja
SingleAlarmActionName	Name der Aktion	ja/ja
SingleAlarmActionParams	Parameter der Aktion	ja/ja
SingleAlarmInfoText	Infotext für Meldung	ja/ja
SingleAlarmGroup	Name der anwenderdefinierten Gruppenmeldung, der eine Meldung zugeordnet ist.	ja/ja

1)

```
Sub DeleteSingleAlarmTagNameProcessValue1()
     'HMIGO_033
     Dim objGO as HMIGO
     Dim var as Long
     var = 0
     Set objGO = new HMIGO
     'message 1 will be modified
       objGO.GetSingleAlarm 1
       objGO.SingleAlarmTagNameProcessValue1 = var
       objGO.CommitSingleAlarm
     Set objGO = nothing
   End Sub
```

Siehe auch

ListSingleAlarm-Funktion (Seite 2559)

GetSingleAlarm-Funktion (Seite 2558)

DeleteSingleAlarm-Funktion (Seite 2557)

CreateSingleAlarm-Funktion (Seite 2553)

CommitSingleAlarm-Funktion (Seite 2552)

CloseSingleAlarm-Funktion (Seite 2551)

VBA in weiteren WinCC-Editoren (Seite 2493)

CloseSingleAlarm-Funktion

Beschreibung

Schließt die geöffnete Meldung.

Hinweis

Geänderte Parameter werden nicht gespeichert. Wenn Sie die aktuellen Werte speichern wollen, müssen Sie wieder die Funktion CommitSingleAlarm() ausführen.

Syntax

```
Ausdruck.CloseSingleAlarm()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

--

Beispiel

```
Sub CloseSingleAlarm()
' HMIGO_22
' procedure to close a singlealarm
' message #100 need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim lngMsgNumber As Long
  Set objHMIGO = New HMIGO
  lngMsgNumber = 100
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateSingleAlarm, vbOKOnly, "Status SingleAlarm"
 'open a singlealarm
  objHMIGO.GetSingleAlarm lngMsgNumber
 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateSingleAlarm, vbOKOnly, "Status SingleAlarm"
 'close a singlealarm
  objHMIGO.CloseSingleAlarm
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateSingleAlarm, vbOKOnly, "Status SingleAlarm"

  Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListSingleAlarm-Funktion (Seite 2559)

GetSingleAlarm-Funktion (Seite 2558)

DeleteSingleAlarm-Funktion (Seite 2557)

CreateSingleAlarm-Funktion (Seite 2553)

CommitSingleAlarm-Funktion (Seite 2552)

VBA im Alarm Logging (Seite 2547)

CommitSingleAlarm-Funktion

Beschreibung

Schreibt die geänderten Parameter der geöffneten Meldung nach WinCC.

Hinweis

Wenn Sie nach einem CommitSingleAlarm-Aufruf weitere Eigenschaften ändern, müssen Sie diese Änderungen mit einem erneuten Aufruf der Funktion nach WinCC schreiben.

Syntax

```
Ausdruck.CommitSingleAlarm()
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

--

Beispiel

```
Sub CommitSingleAlarm()
' HMIGO_023
' procedure to change a property of a singlealarm
' message #100 need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim lngMsgNumber As Long
  Dim lngMsgBitNumber As Long
  Set objHMIGO = New HMIGO
  lngMsgNumber = 100
  lngMsgBitNumber = 10
 'current status is "EMPTY"
```

```
   MsgBox objHMIGO.ObjectStateSingleAlarm, vbOKOnly, "Status SingleAlarm"
  'open a singlealarm
   objHMIGO.GetSingleAlarm lngMsgNumber
  'current status is "OPENED" for changes
   MsgBox objHMIGO.ObjectStateSingleAlarm, vbOKOnly, "Status SingleAlarm"
  'change a property
   objHMIGO.SingleAlarmMessageBit = lngMsgBitNumber
  'current status is "MODIFIED" for changes
   MsgBox objHMIGO.ObjectStateSingleAlarm, vbOKOnly, "Status SingleAlarm"
  'commit a single alarm
   objHMIGO.CommitSingleAlarm
  'current status is "OPENED"
   MsgBox objHMIGO.ObjectStateSingleAlarm, vbOKOnly, "Status SingleAlarm"

   Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListSingleAlarm-Funktion (Seite 2559)

GetSingleAlarm-Funktion (Seite 2558)

DeleteSingleAlarm-Funktion (Seite 2557)

CreateSingleAlarm-Funktion (Seite 2553)

CloseSingleAlarm-Funktion (Seite 2551)

VBA im Alarm Logging (Seite 2547)

CreateSingleAlarm-Funktion

Beschreibung

Erstellt eine neue Meldung.

Syntax

```
Ausdruck.CreateSingleAlarm(MessageNumber,ClassID,MessageTypeID,Text1
ID,MessageTagName,MessageBit)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
MessageNumber (Long)	Nummer der Meldung. • Wird hier eine nicht verwendete Meldungsnummer angegeben, so wird diese übernommen. • Wird der Wert "0" eingetragen, wird die Meldungsnummer vom System frei vergeben. In diesem Fall wird hier die Meldungsnummer zurück gegeben.
ClassID (HMIGO_SINGLE_ALARM_CLASS_IDS)	Meldeklasse. Die möglichen Werte stehen in der Tabelle Enum HMIGO_SINGLE_ALARM_CLASS_IDS.
MessageTypeID (Integer)	Die erlaubten Werte sind von der Meldeklasse abhängig: • Klasse 1: Werte von 1 bis 16 • Klasse 2: Werte von 17 bis 32 • Klasse 3: Werte von 33 bis 48 • ... • Klasse 18: Werte von 263 bis 288
Text1ID (Long)	ID für den ersten Anwendertext. Über die Funktion ModifySingleAlarm können weitere neun Anwendertexte definiert werden (1-10).
MessageTagName (String)	Variablenname für das Ereignis.
MessageBit (Integer)	Bit bei Bitmeldeverfahren (0...31)

Default-Werte beim Erstellen einer neuen Meldung

Die folgende Tabelle zeigt die Default-Werte, die beim Erstellen einer neuen Meldung eingetragen werden. Diese Eigenschaften können verändert werden. Über die ModifySingleAlarm-Funktion werden die Änderungen gespeichert.

Parameter	Default-Wert (Enum-Name => Wert)	Bemerkung
SingleAlarmAGNumber	0	--
SingleAlarmCPUNumber	0	--
SingleAlarmTextXXID	Kein Text eingetragen	--
SingleAlarmTagNameProcessValueXX	Keine Variable eingetragen	--
SingleAlarmQuitTag	Keine Variable eingetragen	--
SingleAlarmQuitBits	0	Keine Bits gesetzt.
SingleAlarmStateTag	Keine Variable eingetragen	Entspricht genau einem Tag. Nur relevant bei Verdichtungsvariablen.
SingleAlarmStateBits	0	Keine Bits gesetzt.
SingleAlarmNormDLL	Kein Name eingetragen	--

Parameter	Default-Wert (Enum-Name => Wert)	Bemerkung
SingleAlarmQuitSingle	FALSE	Einzelquittierung, keine Sammelquittierung.
SingleAlarmHornActivate	FALSE	Keine Aktivierung der Hupe.
SingleAlarmArchiving	TRUE	Meldung wird archiviert.
SingleAlarmProtocol	TRUE	Meldung wird protokolliert.
SingleAlarmFlankInvert	FALSE	Nicht aktiviert.
SingleAlarmLockedOnStart	FALSE	Meldung ist nicht gesperrt.
SingleAlarmGlobalAPIFunction	FALSE	Meldung wird nicht weitergegeben.
SingleAlarmActionName	Keine Name eingetragen	--
SingleAlarmActionParams	Keine Parameter für Aktion eingetragen	--
SingleAlarmInfoText	Kein Text eingetragen	--
SingleAlarmGroup	Kein Text eingetragen	--

Enum HMIGO_SINGLE_ALARM_CLASS_IDS

Folgende Meldeklassen stehen zur Auswahl:

Werte	Beschreibung
SINGLE_ALARM_ERROR (1)	--
SINGLE_ALARM_CLASS_2 (2)	--
SINGLE_ALARM_CLASS_3 (3)	--
SINGLE_ALARM_CLASS_4 (4)	--
SINGLE_ALARM_CLASS_5 (5)	--
SINGLE_ALARM_CLASS_6 (6)	--
SINGLE_ALARM_CLASS_7 (7)	--
SINGLE_ALARM_CLASS_8 (8)	--
SINGLE_ALARM_CLASS_9 (9)	--
SINGLE_ALARM_CLASS_10 (10)	--
SINGLE_ALARM_CLASS_11 (11)	--
SINGLE_ALARM_CLASS_12 (12)	--
SINGLE_ALARM_CLASS_13 (13)	--
SINGLE_ALARM_CLASS_14 (14)	--
SINGLE_ALARM_CLASS_15 (15)	--
SINGLE_ALARM_CLASS_16 (16)	--
SINGLE_ALARM_CLASS_SYSTEM_REQUIRE_ACKNOWLEDGEMENT (17)	--
SINGLE_ALARM_CLASS_SYSTEM_WITHOUT_ACKNOWLEDGEMENT (18)	--

Beispiel

```
Sub CreateSingleAlarm()
' HMIGO_024
' procedure to create a SingleAlarm
' message must not be created before
' message Text ID need to be created before in text library
' declarations
  Dim objHMIGO As HMIGO
  Dim strMsgText As String      'message text
  Dim strMsgTagName As String   'message variable
  Dim lngMsgNumber As Long      'message number
  Dim lngMsgBitNumber As Long   'bit number within the message variable
  Dim lngMsgTypeID As Long      'message type
  Dim lngMsgClassID             'SINGLE_ALARM_ERROR
  Dim lngMsgTextID As Long      'message text ID from textlibrary
  Set objHMIGO = New HMIGO
  strMsgText = "NewText"
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateSingleAlarm, vbOKOnly, "Status SingleAlarm"
 'preset reqiured parameter
  lngMsgNumber = 50
  lngMsgClassID = 1
  lngMsgTypeID = 2
  lngMsgTextID = 69
  strMsgText = "new text message"
  strMsgTagName = "NewVariable"
  lngMsgBitNumber = 5

 'create a tag
  objHMIGO.CreateSingleAlarm lngMsgNumber,SINGLE_ALARM_ERROR, lngMsgTypeID,lngMsgTextID,
strMsgTagName, lngMsgBitNumber

 'current status is "OPENED"
  MsgBox objHMIGO.ObjectStateSingleAlarm, vbOKOnly, "Status SingleAlarm"
  Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListSingleAlarm-Funktion (Seite 2559)

GetSingleAlarm-Funktion (Seite 2558)

DeleteSingleAlarm-Funktion (Seite 2557)

CommitSingleAlarm-Funktion (Seite 2552)

CloseSingleAlarm-Funktion (Seite 2551)

VBA im Alarm Logging (Seite 2547)

DeleteSingleAlarm-Funktion

Beschreibung

Löcht die angegebene Meldung.

Syntax

```
Ausdruck.DeleteSingleAlarm(MessageNumber)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
MessageNumber (Long)	Nummer der zu löschenden Meldung.

Beispiel

```
Sub DeleteSingleAlarm()
' HMIGO_025
' procedure to delete a singlealarm
' message #100 need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim lngMsgNumber As Long

  Set objHMIGO = New HMIGO
  lngMsgNumber = 100
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateSingleAlarm, vbOKOnly, "Status SingleAlarm"

 'delete a singlealarm
  objHMIGO.DeleteSingleAlarm lngMsgNumber
  Set objHMIGO = Nothing
End Sub
```

Siehe auch

VBA im Alarm Logging (Seite 2547)

ListSingleAlarm-Funktion (Seite 2559)

GetSingleAlarm-Funktion (Seite 2558)

CreateSingleAlarm-Funktion (Seite 2553)

CommitSingleAlarm-Funktion (Seite 2552)

CloseSingleAlarm-Funktion (Seite 2551)

GetSingleAlarm-Funktion

Beschreibung

Liest die Parameter der eingegebenen Meldung ein.

Die Parameter können Sie über die Objekteigenschaften ändern oder lesen. Eine Auflistung der verfügbaren Objekteigenschaften finden Sie in dieser Dokumentation unter "VBA im Alarm Logging".

Syntax

```
Ausdruck.GetSingleAlarm(MessageNumber)
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
MessageNumber (Long)	Die Meldungsnummer der Meldung, die eingelesen werden soll.

Beispiel

```
Sub GetSingleAlarm()
' HMIGO_026
' procedure to open a singlealarm
' message #100 need to be created before
' declarations
  Dim objHMIGO As HMIGO
  Dim lngMsgNumber As Long
  Set objHMIGO = New HMIGO
  lngMsgNumber = 100
 'current status is "EMPTY"
  MsgBox objHMIGO.ObjectStateSingleAlarm, vbOKOnly, "Status SingleAlarm"
```

```
  'open/ get a tag
   objHMIGO.GetSingleAlarm lngMsgNumber
  'current status is "OPENED"
   MsgBox objHMIGO.ObjectStateSingleAlarm, vbOKOnly, "Status SingleAlarm"
   Set objHMIGO = Nothing
End Sub
```

Siehe auch

ListSingleAlarm-Funktion (Seite 2559)
DeleteSingleAlarm-Funktion (Seite 2557)
CreateSingleAlarm-Funktion (Seite 2553)
CommitSingleAlarm-Funktion (Seite 2552)
CloseSingleAlarm-Funktion (Seite 2551)
VBA im Alarm Logging (Seite 2547)

ListSingleAlarm-Funktion

Beschreibung

Die ListSingleAlarm-Funktion gibt folgende Inhalte des Alarm Logging in einer Liste zurück:

- alle erstellten Aktionen, die mit Meldungen verbunden sind
- alle erstellten ID der Meldeklassen
- alle erstellten Infotexte
- alle erstellten Meldungsnummern
- alle erstellten Typ ID der Meldungen
- alle erstellten Meldeklassen
- alle erstellten Gruppenmeldungen

Syntax

```
Ausdruck.ListSingleAlarm(ListType,pListArray,[Filter])
```

Ausdruck

Erforderlich. Ein Ausdruck, der ein Objekt vom Typ "HMIGO" zurückgibt.

Parameter

Parameter (Datentyp)	Beschreibung
ListType (HMIGO_SINGLE_ALARM_LIST_TYPE)	Legt fest, welcher Inhalt in einer Liste zurückgegeben wird. Möglich ist: • SINGLE_ALARM_ACTION_NAMES (1) alle erstellten Aktionen für Loop In Alarm, wenn der Parameter als String in der Konfiguration gesetzt ist • SINGLE_ALARM_CLASS_IDS (2) alle erstellten ID der Meldeklassen • SINGLE_ALARM_INFO_TEXTS (3) alle erstellten Infotexte • SINGLE_ALARM_MESSAGE_NUMBERS (4) alle erstellten Meldungsnummern • SINGLE_ALARM_MESSAGE_TYPE_IDS (5) alle erstellten Typ ID der Meldungen • SINGLE_ALARM_GROUP_MESSAGE_CLASSES (6) alle erstellten Meldeklassen • SINGLE_ALARM_GROUP_MESSAGE_USER_DEFINED (7) alle erstellten Gruppenmeldungen
pListArray (Variant)	Liste mit dem angeforderten Inhalt.
Filter (String)	Optional können Filter gesetzt werden. Die Wildcards "*" und "?" werden unterstützt.

Beispiel

Im folgenden Beispiel wird überprüft, ob Infotexte projektiert wurden:

```
Sub ReadSingleAlarm()
'HMIGO_032
'read content in alarm logging
'no info texts are implemented
  Dim objHMIGO As New HMIGO
  Dim arrContent As Variant
'read all info texts
  objHMIGO.ListSingleAlarm SINGLE_ALARM_INFO_TEXTS, arrContent
'check result
  If (UBound(arrContent) - LBound(arrContent) + 1) <= 0 Then
    MsgBox "no entries because no info texts are implemented"
  End If
End Sub
```

Siehe auch

CreateSingleAlarm-Funktion (Seite 2553)

CommitSingleAlarm-Funktion (Seite 2552)

CloseSingleAlarm-Funktion (Seite 2551)

VBA im Alarm Logging (Seite 2547)

Index

Ä

B

C

D

E

F

G

H

I

K

L

M

N

O

Ö

P

Q

R

S

T

U

V

W

X

Y

Z