1 与 WinCC 基本系统的差别

1.1 脚本执行

与 WinCC 基本系统不同，事件-触发脚本（例如 OnClick 等）与画面显示在同一个进程空间中运行。换句话说，在 Internet Explorer 的上下文中运行。

当在脚本中使用延时功能（例如 Sleep(3000)）时必须考虑上述问题。此时，这段时间的画面显示不能正常显示。

1.2 画面选择

在 Web Navigator 中，WinCC 画面是通过 Internet 异步下载的。
这就是为什么画面改变不是同步的，相对于基本系统而言是异步的。当使用脚本触发画面切换时必须考虑这个事实。

示例：

如果在脚本中触发了一个画面改变，切换到画面窗口，那么新画面中的对象不能在该脚本中进行访问。因为在那时画面还没有被载入。延时（例如 Sleep(2000)）也无助于此，因为脚本和 Internet Explorer 都在等待。

最好的解决方法是在后续的画面打开事件 OpenPicture 中执行这段脚本。

1.3 脚本编写

预定义：

下面是 Web Navigator 在脚本中特殊的预定义。

RUN_ON_WEBNAVIGATOR

该定义下面的脚本仅在 Web Navigator 中执行。因此，能够在 Web Navigator 中编写与基本系统具有不同行为的脚本。

例如：

```
void  OnOpenPicture  (char*  lpszPictureName,  char*
lpszObjectName, char*  lpszPropertyName)
{
#ifdef  RUN_ON_WEBNAVIGATOR
//  在这里编写仅在 Web Navigator 中执行的代码
}
#else
//  在这里编写仅在 WinCC 基本系统中执行的代码
#endif
}
```

1.4 访问其它画面中的对象

如果想从当前画面中通过脚本访问其它画面中的对象（例如，父画面中的对象或者画面窗口中的对象），那么仅能使用相对寻址方式，这与基本系统不同。

所有访问画面中对象的 C 脚本函数都以画面名称作为第一个参数。

示例： SetPropChar ("Start picture", "Text1", "Text", "Hallo here is a text");

当访问的对象不在同一个画面中时，在 Web Navigator 中具有特殊的画面名称，它指定了画面相对当前画面的位置。

在基本画面中定位一个对象

如果名称唯一，只需简单指定画面名称。

```
SetPropChar ("Start picture", "Text1", "Text", "Hallo here is a text");
```

否则，独立的画面名称，使用空字符串或者斜杠（lpszPictureName = "/"）。

示例：

```
SetPropChar ("", "Text1", "Text", "Hallo here is a text");

or

SetPropChar ("/", "Text1", "Text", "Hallo here is a text");
```

定位父画面中的对象

如果名称唯一，只需简单指定父画面的画面名称。

```
SetPropChar ("Picture1", "Text1", "Text", "Hallo here is a text");
```

否则，独立的画面名称，可以写成（lpszPictureName = "../"）。

示例：

```
SetPropChar ("../", "Text1", "Text", "Hallo here is a text");
```

画面窗口中的画面

如果欲访问画面窗口“Picture window 1”中的画面中的对象，那么可以相对当前画面进行定位。

```
SetPropChar ("./Picture window 1", "Text1", "Text",
"Hallo here is a text");
```

或者，如果 picture window 1 在父窗口中，

```
SetPropChar ("../Picture window 1", "Text1", "Text",
"Hallo here is a text");
```

或者，picture window 1 在根画面中，

```
SetPropChar ("/Picture window 1", "Text1", "Text", "Hallo
here is a text");
```

1.5 同步脚本函数

谨慎使用同步脚本函数，如 SetTagCharStateWait 等。它同步地写入一个变量并等待事件。因为这些调用同步地传到服务器，频繁调用这些脚本函数会对客户机性能产生很大的负面影响。对于 Internet 连接，这样的调用甚至要等上数秒。

操作画面中对象的脚本函数是安全的，因为到服务器的访问是单程的。

1.6 定义本地画面脚本变量

如果打算在脚本中使用任何画面特定的数据，应当避免在 Web Navigator 中使用数据管理器变量。在脚本中的读写，总会产生到服务器的往复操作，在 Internet 上，这种操作的性能比较差。

几种获取画面特定数据的方法。

组态不可见的图形对象

创建隐藏的图形对象，然后将这些隐藏对象的属性作为特定画面的变量。

另外，通过改变这些存储值，也利于执行动作。

示例：

创建一个隐藏的文本域。

可以方便地在 Text 属性中存储任何文本，随后可以通过任何脚本来读写这些文本。

在文本域的 Color 属性中可以存储颜色，可以通过脚本来读写它们。从而来干预画面的状态，但无需到服务器的往复操作。