SIEMENS

SIMATIC NET

# S7Beans / Applets for Advanced CPs

**Programming Tips**

**Supplement to the documentation of the Advanced CPs**
**CP 243−1 IT**
**CP 343−1 IT / CP 343−1 Advanced**
**CP 443−1 IT / CP 443−1 Advanced**




**Release 06/2008**
**C79000−G8976−C180−03**

## Classification of Safety−Related Notices

This manual contains notices which you should observe to ensure your own personal safety, as well as to protect the product and connected equipment. These notices are highlighted in the manual by a warning triangle and are marked as follows according to the level of danger:

**Danger**

indicates that death or severe personal injury **will** result if proper precautions are not taken.

**Warning**

indicates that death or severe personal injury **can** result if proper precautions are not taken.

**Caution**

with warning triangle indicates that minor personal injury can result if proper precautions are not taken.

**Caution**

without warning triangle indicates that damage to property can result if proper precautions are not taken.

**Notice**

indicates that an undesirable result or status can occur if the relevant notice is ignored.

**Note**

highlights important information on the product, using the product, or part of the documentation that is of particular importance and that will be of benefit to the user.

## Trademarks

SIMATIC® , SIMATIC HMI® and SIMATIC NET® are registered trademarks of SIEMENS AG.

Third parties using for their own purposes any other names in this document which refer to trademarks might infringe upon the rights of the trademark owners.

## Safety instructions regarding your product:

Before you use the product described here, read the safety instructions below thoroughly.

## Qualified personnel

Only **qualified personnel** should be allowed to install and work on this equipment. Qualified persons are defined as persons who are authorized to commission, to ground, and to tag circuits, equipment, and systems in accordance with established safety practices and standards.

## Correct usage of hardware products

Note the following

### Warning

This device and its components may only be used for the applications described in the catalog or the technical description, and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens.

This product can only function correctly and safely if it is transported, stored, set up, and installed correctly, and operated and maintained as recommended.

Before you use the supplied sample programs or programs you have written yourself, make certain that no injury to persons nor damage to equipment can result in your plant or process.

EU directive: Do not start up until you have established that the machine on which you intend to run this component complies with the directive 89/392/EEC.

## Correct usage of software products

Note the following

### Warning

This software may only be used for the applications described in the catalog or the technical description, and only in connection with software products, devices, or components from other manufacturers which have been approved or recommended by Siemens.

Before you use the supplied sample programs or programs you have written yourself, make certain that no injury to persons nor damage to equipment can result in your plant or process.

## Prior to startup

Before putting the product into operation, note the following warning:

### Caution

Prior to startup you must observe the instructions in the relevant documentation. For ordering data of the documentation please refer to the catalogs or contact your local SIEMENS representative.



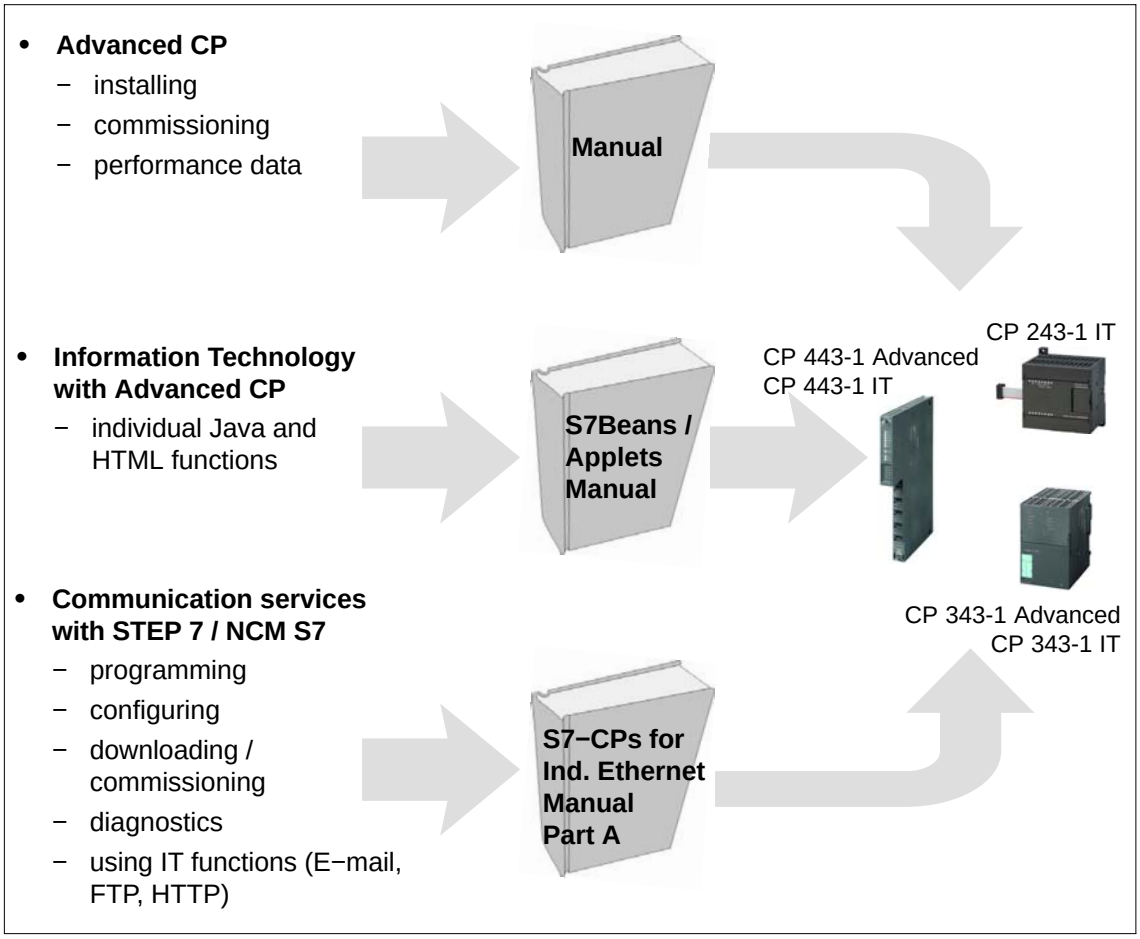
Siemens AG
Industry Automation
Industrial Communication
Postfach 4848, D-90327 Nürnberg

Subject to technical change.

# Preface / References

### Manuals on the topic of IT in SIMATIC

Information technology with Advanced CPs in SIMATIC is described in the following manuals:




You will also find these documents on the Manual Collection CD. This symbol is used at various points in the text to indicate that there is additional information and examples on the Manual Collection CD.

## Validity of these Programming Tips

This document is valid

- for the CP 443-1 Advanced for the SIMATIC S7-400

  and

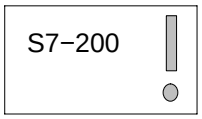
  for the CP 343-1 Advanced for the SIMATIC S7-300

  - with the required configuration software STEP 7 version 5.4 SP4 with NCM S7 for Industrial Ethernet
- for the CP 243−1 IT for the SIMATIC S7−200
  - with the required configuration software STEP 7 MicroWin version 3.2 SP1 or higher
- S7 BeansAPI version 2.5 or higher
- JBuilder version 3.0 or higher

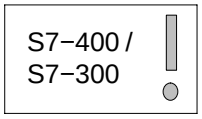
## Note

**Notice**

Please note that the S7−200 differs in some technical aspects from the S7−300/S7−400, where necessary in the description, these aspects are highlighted by the following symbols in the margin.




Text indicated by this symbol applies only to the S7−200.




Text indicated by this symbol applies only to the S7−300/S7−400 .

## References /.../

References to further documentation are specified with documentation numbers in slashes /.../. Based on these numbers, you can check the title of the documentation in the list of references at the end of the manual.

# Contents

The complete manual and the programming tips are in the Manual Collection. This symbol is used at various points in the text to indicate that there is additional information and examples in the Manual Collection.

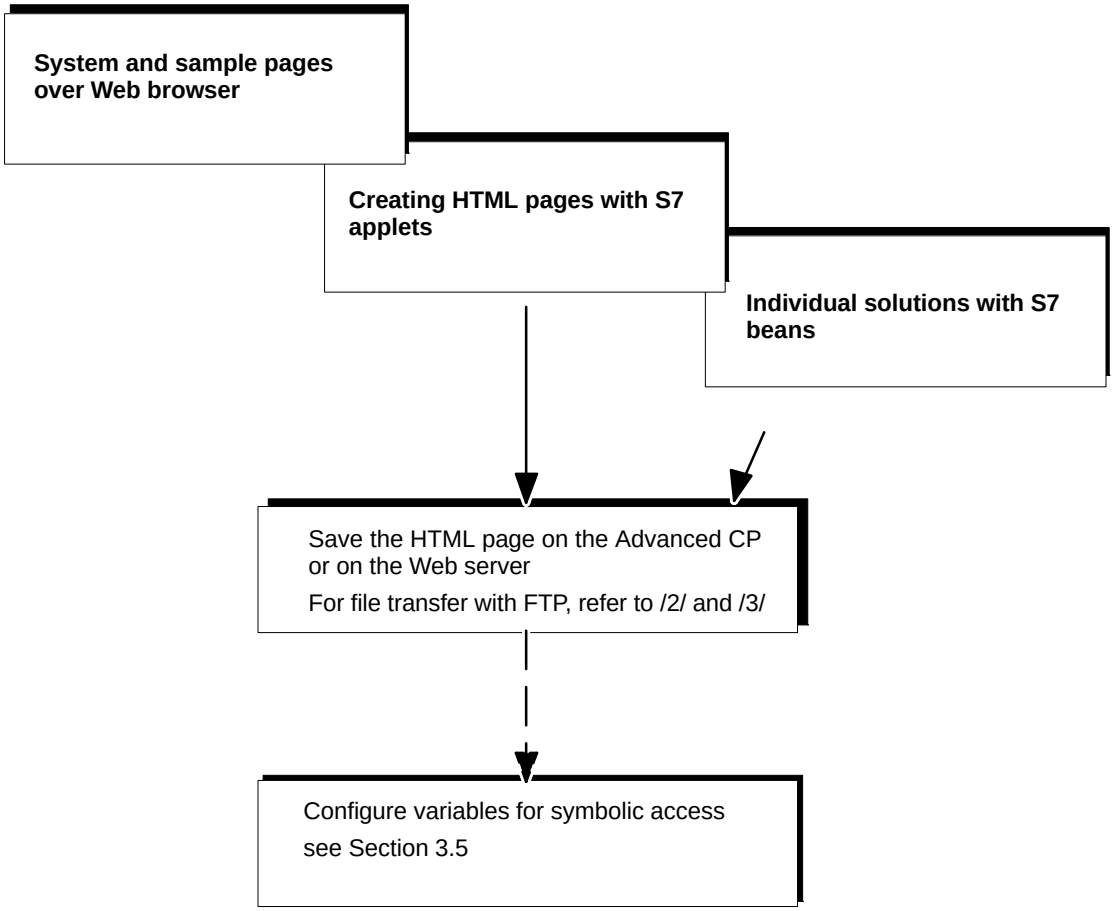
# 1 Creating HTML Pages − Overview

### Multilevel concept

The Advanced CP provides several levels to implement device and process data monitoring with HTML pages:

### How to...

...create or adapt your own individual HTML pages:




- System and sample pages over Web browser

  You want to use the options of the HTML process control predefined for the Advanced CP without extensive programming.

  The possibilities available to you are introduced in /2/.

- Creating HTML pages with S7 applets

  The Advanced CP provides you with predefined S7 applets with which you can create HTML pages and adapt them to your task.

The calls and call parameters are described in this manual on the S7 applets / beans.

- Individual solutions with S7 beans

  You want to use graphics options adapted to your application and create more complex applets.

  You not only want to display your process data in the plant pictures but also want to use the data, for example, for evaluation in a database.

  You can achieve this with the following options:

  − Create application-specific Java applets and Java applications and use predefined S7 beans, Java beans of other manufacturers and your own Java source text.

  − Create Java source code; use application-specific applets, Java beans and the supplied S7 beans.

# 2 Web Browser

## 2.1 Accessing the Advanced CP with a Web browser

### What is required?

To access the HTML pages on the Advanced CP or on the Web server you require a Web browser, for example Internet Explorer. The Web browser must meet the following requirements:

- JDK (Java Development Kit) 1.1.X is supported.

The Internet Explorer meets these requirements. Other Web browsers with the same range of functions can also be used.

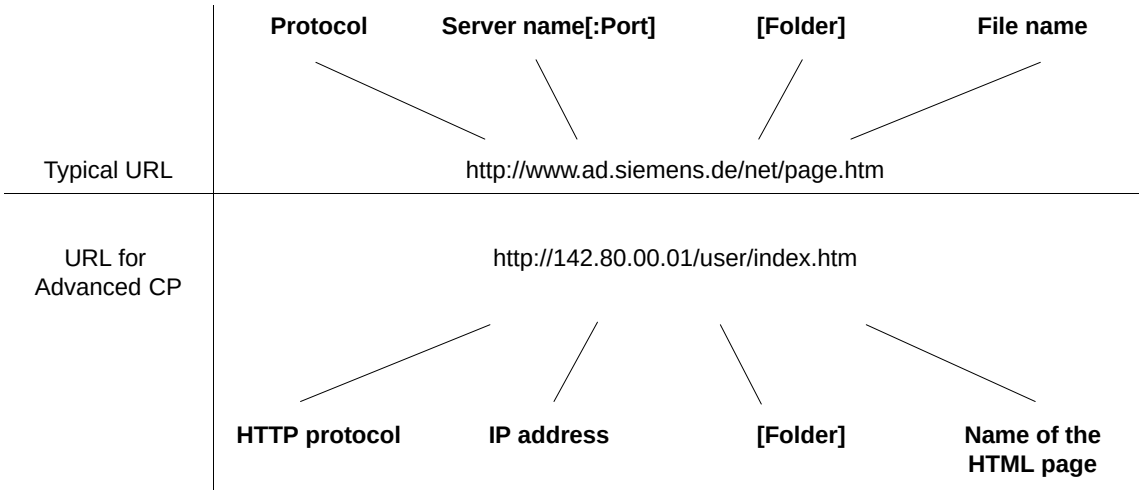
Other Web browsers may meet these requirements only with certain restrictions. You require a plug-in component corresponding to the Java reference implementation of a SUN Java Virtual Machine.

Note the other information including the versions of the products mentioned here in the manuals /1/ in the Manual Collection.

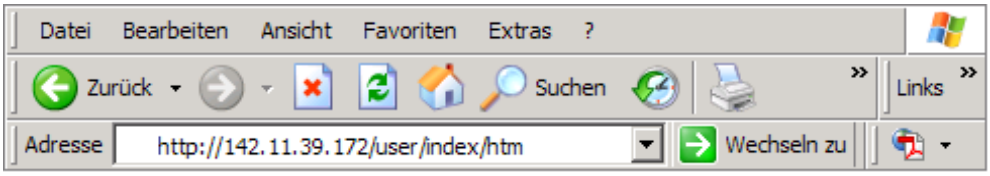
You will also find information and add-ons on the Internet.

## URL: Uniform Resource Locator

In the World Wide Web, addressing using URL has become standard. You can also access the Advanced CP with your Web browser using the URL. This URL can have almost any complexity but consists in principle of four essential parts. The following schematic illustrates the structure (typical URL) and shows the contents for calling Advanced CPs.




When accessing the Advanced CP using a Web browser, use the HTTP protocol to address the Web server on the Advanced CP:




You inform the CP of the IP address during configuration with STEP 7. If you have an attachment from Industrial Ethernet to your intranet or to the Internet, the CP can be contacted using the IP address in the intranet or Internet.

A detailed description of the structure of the IP address and the options of creating subnets or subnet masks is beyond the scope of this manual. You will find more detailed information in the STEP 7 online help and in the documentation listed in the references.

## 2.2 Settings in the Web browser

### General

Before you can access the Advanced CP using your Web browser, you must make or at least check certain settings. The settings are explained below based on the example of the Internet Explorer.

The settings shown here have been selected so that the execution of the S7 applets and S7 beans (JavaBeans) used on the Advanced CP is possible.

### Settings in the Internet Explorer

- Starting the Java Interpreter

  You will find the functions for the Java applications under the menu command “Tools” ▸ “Internet Options” ▸ “Advanced” tab, under the entry “Java (Sun) / Use JRE...”.

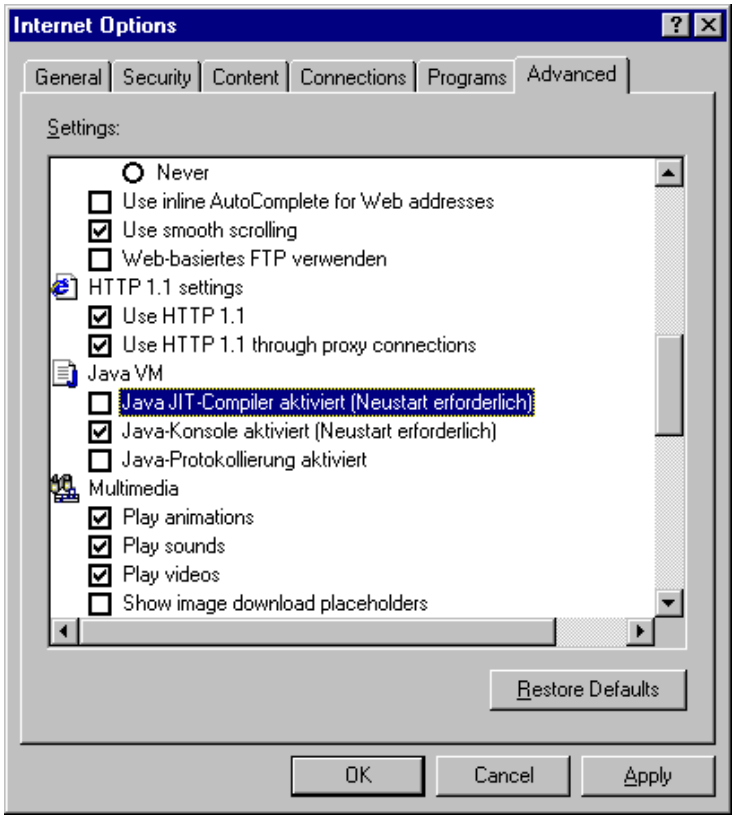



Figure 2-1

- Setting a proxy server

  For more information, check with your system administrator.

## Starting the Java Console

To track the execution of the Java applet and to get information about problems, you can start the Java Console.

In the Internet Explorer, use the “Use JRE...” option in the Internet Options dialog box in the “Advanced” tab as shown above. To call the Java Console, use the menu command “Tools” ▸ “Sun Java Console”.

**Note**

With the Internet Explorer, the Java console must first be activated before it can be used. To do this, select “Tools” ▸ “Internet Options” ▸ “Programs” tab ▸“Manage Add-ons” button ▸ and activate “Java plug-in <version>”.

# 3 HTML Pages on the Advanced CP

This chapter answers the following questions:

- How are HTML pages created that can access information on the S7 station?
- What are S7 applets and how are they used in HTML pages? What do I need to remember?
- How do I store my own HTML pages?
- How do I obtain a graphic representation of process information?
- How are HTML pages checked?

## 3.1 Creating HTML pages for the Advanced CP

### Uses

With your own HTML pages, the following possibilities are open to you:

- Process visualization in the Web browser adapted to your particular plant
- Process data represented numerically or graphically in the Web browser
- The ability to include the results of status queries in the display
- The querying and representation of process data in one HTML page on several S7 stations and on distributed systems.

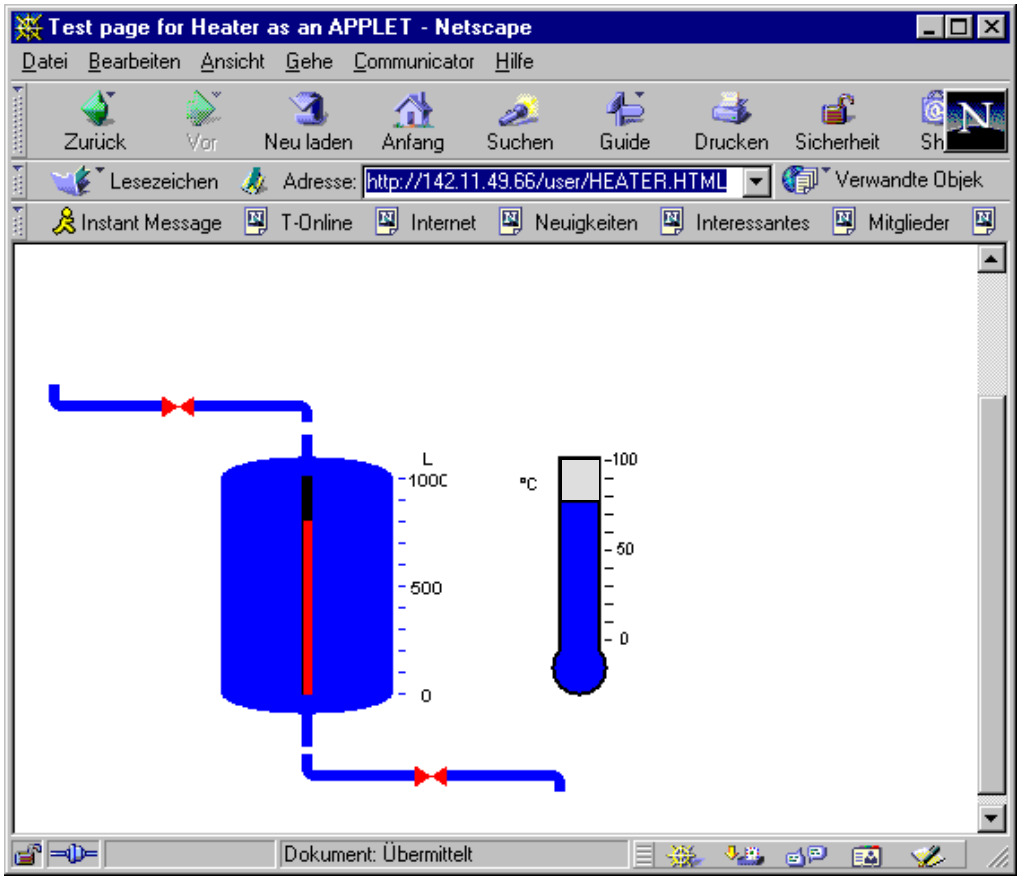



Figure 3-1

## HTML editor

To create your own HTML pages, you require an editor. It is easier to keep to the syntactic conventions for HTML pages if you use a suitable HTML editor. These normally allow the input of formatted text and the inclusion of graphics. The conversion to HTML syntax then takes place in the background. It is also normally possible to change the view and make your input directly in the HTML form.

**Note**

Depending on the settings, some HTML editors produce specific HTML code that may not be understood by all Web browsers. Make sure that the syntax of your HTML editor is compatible with your Web browser.

After completing the pages, you can transfer them using an FTP client or, in some cases, directly from the HTML editor to the Advanced CP.

## S7 applets are applets for SIMATIC S7

The Advanced CP provides several applets with which you can access the controller from the Web browser on your PC. You do not need to be familiar with Java to use these S7 applets. By following the instructions below, you will be able to integrate the calls in your HTML page simply and quickly.

## Extended access and display options − The Java Beans concept

The Java Beans concept allows you to create objects (Java components) and to link them simply to executable programs.

There is an S7 beans class library available for the Advanced CP (S7BeansAPI). The object classes contained in this library can be used for object-oriented access to a variety of information on the SIMATIC S7 and for graphic display of process variables.

The S7 beans class library provides an open interface allowing you to extend process data evaluation for example with databases, table calculation or management information systems.

### Organizing files − resources of the Advanced CP

The Advanced CP has memory available for storing your HTML pages. You will find information on this topic in the device manual of the Advanced CP /1/.

Please note the information in the readme.htm file on the Advanced CP.

**S7-400 / S7-300**

The simplest way to open the readme.htm file is by clicking the “Information” link on the home page of the Advanced CP.

This contains information about the meaning and purpose of the shipped files. You can then decide which files might be useful for your application. Using FTP functions (see /2/ ), you can organize the files on the Advanced CP to suit your requirements.

## 3.2 Designing HTML pages − a few basics

### A word of introduction

There is a considerable body of excellent literature dealing with the design of HTML pages. Check the recommended literature dealing with the topics of Web, HTML etc. in the appendix to this manual. This manual itself is deliberately restricted to explaining how you can include the functions supplied with the Advanced CP in your HTML application.

With the information here, you should be in a position to create and run HTML pages and use the S7 applets without needing a thorough study of HTML techniques.

Experienced authors of HTML pages will be able to use the information about assigning parameters for S7 applets in the following sections directly. For the less experienced user, we have included a certain amount of information on the topic. This information can then be extended by reading the literature mentioned above.

### Planning the structure

From the very beginning, you should be clear in your own mind about your document and page structure. HTML allows you to jump from topic to topic in the HTML pages displayed by the Web browser and, in the case of the Advanced CP, from controller to controller or plant to plant. Basically this means that the scope of the HTML pages depends on the links that you specify when you create the pages. On the other hand, the links themselves also depend on the storage of the HTML documents.

## Linking documents (HTML pages)

HTML pages are connected by links that are structured as follows:

- URLs with absolute address information

  Example:
  **<A HREF=”http://www.ad.siemens.de/net/index.htm”>ID_text</A>**

- URLs for simplified link information with relative addresses:
  - Use of relative URLs: Generally, these contain only the name of the folder and the file or even only the file name if the HTML page called is in the same folder as the calling HTML page.

    Example:
    **<A HREF=”processpic.htm”>ID_text</A>**

  - Use of server-related URLs: These addresses begin with ”/” and indicate that the HTML page is on the same server as the calling HTML page.

    Example:
    **<A HREF=”/pics/processpic.htm”>ID_text</A>**

## Creating HTML pages

To design pages that are both practical and attractive, HTML provides you with various description elements. The literature provides suitable information usually under the following topics:

- Displaying tables

  The table is an important tool in structuring information. In particular when designing HTML pages, tables are suitable for compensating certain weaknesses of HTML formats. Displaying HTML text in columns is, for example only possible if you use tables.

- Inserting graphics

  The use of pictures in HTML pages is a topic dealt with extensively in the literature on HTML. The image file formats you can use are GIF and JPG.

- HTML Forms (can only be used on the Advanced CP in conjunction with JavaScript)

  Forms are required when the user is intended to store information in the system in a standardized form. HTML provides various structure elements for interaction with the user. Form functions can also be useful for entering process data.

- Using templates

  The use of format templates allows generally valid formats to be specified and used in the HTML documents. This procedure is known from publishing systems such as MS Word or Word Perfect.

  It is adequate to simply say that there are various ways of linking format templates with HTML documents.

- Creating frames

  Using frames, you can divide HTML pages into several areas. This can significantly increase the clarity. You can, for example, use frames to make sure that the navigation menu remains visible while new areas are loaded.

  Note also the information on the topic ”The number of applet instances in an HTML page is limited” in Section 3.4.

- Embedding applets

  This is the main topic of this chapter, which describes the use of the special S7 applets.

- Using JavaScript

  With adequate experience, using Java Script can extend the function of HTML pages and the interaction with the user.

---

**Note**

The Internet itself is a treasure trove for HTML design. You can load HTML pages in your HTML editor at any time, save them and use them as a template for your own page design.

You can, for example, call up HTML pages you find particularly attractive in the HTML editor and then save them. If you do this, all the data of the pages, including the graphics are saved.

You should, however, remember that some HTML pages are protected by Copyright. You cannot, naturally, use these pages for your own page design.

---

## HTML editor − requirements

Chapter 2 lists the requirements that should be met by an HTML editor.

## S7 applets

With the S7 applets, you incorporate process data display and the input of process data into your HTML page. The S7 applets are described in the following section.

## 3.3 Creating and storing your own ”Home Page”

### Flexible use of the Advanced CP file system

The existing start page provides you with basic functions that are adequate for a large number of requirements.

The Advanced CP file system provides a flexible instrument for the presentation of functions and data adapted to your plant. By creating your own start page, you have the tool to extend the view to cover your entire plant or even further.

You can modify the existing start page or can replace it by your own home page.

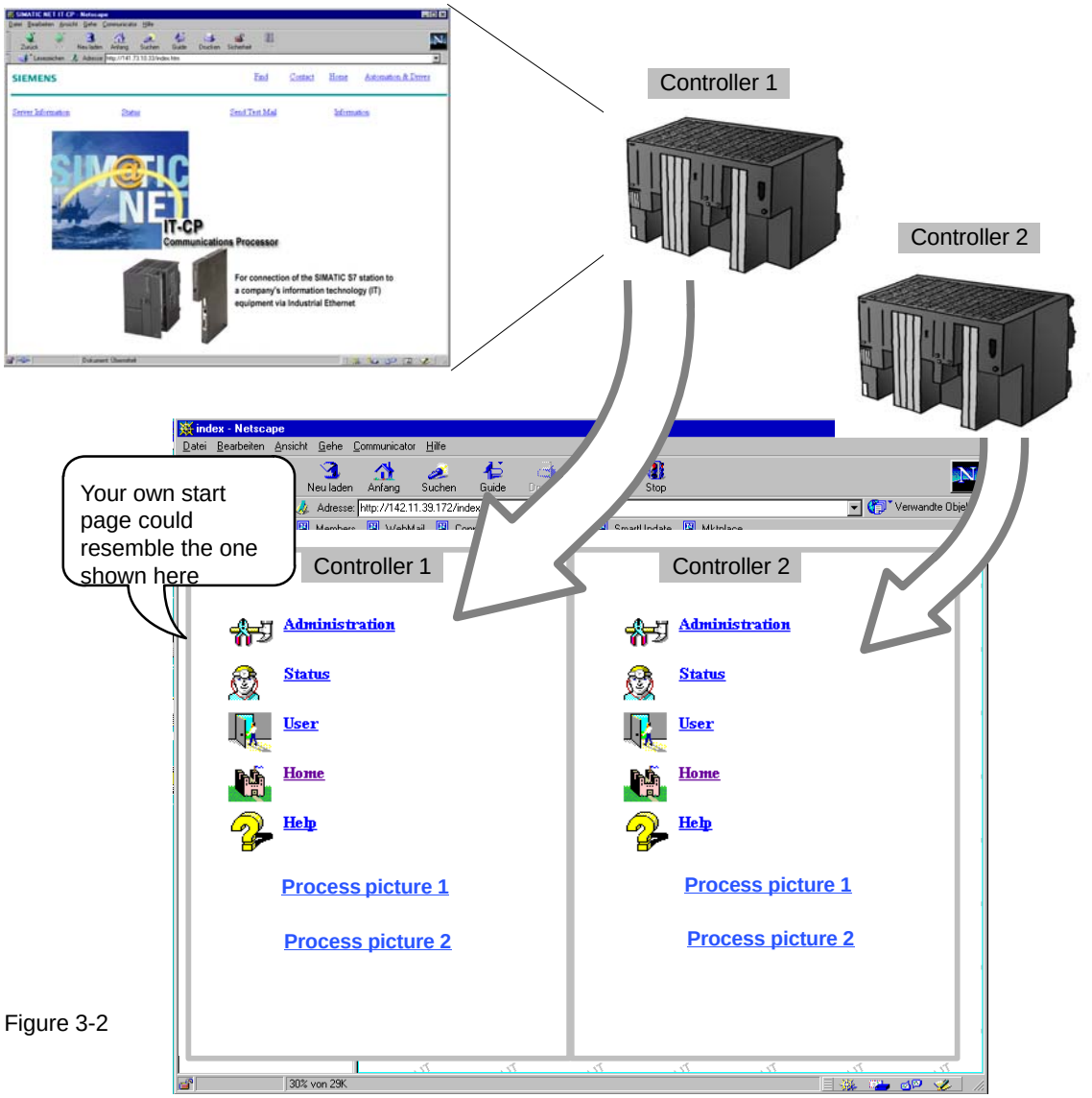



Figure 3-2

### What to do

If you want to start with the existing start page, load this in your HTML editor and add the additional instructions you require.

- The online option

  Load the HTML start page from your Advanced CP in your HTML editor and save it for further editing locally on your PC.

- The offline option

You will also find the HTML start page on the Manual Collection CD. You can then adapt your start page regardless of whether you have access to the Advanced CP and then download it to the Advanced CP.

### Points to remember

Refer to the information in the manual of the Advanced CP /1/ regarding the following points.

- The number of files that can be stored is limited by the size of the file system
- The number of characters in the URLs to be specified is limited.
- The length of the file names is limited.

### Including S7 applets

Flexible access to distributed HTML system pages is **one** aspect of designing the home page.

You have further opportunities for querying information if you include the S7 applets and S7 Beans in your HTML pages. This is explained in detail in later chapters.

### Examples

You will find sample applications on the SIMATIC NET Quick Start CD. You can order this on the Internet at the following address:

http://www.siemens.com/simatic-net/quickstart

### Downloading HTML pages

Use an FTP client and the file management functions as described in /2/ and /3/ to add to or replace the existing HTML pages.

## 3.4 S7 applets

### Meaning

S7 applets are special applets that allow read and write access to an S7 station via the Advanced CP.

The Web browser in which the applet was started is responsible for execution of the applets. This activates the applet and assigns a frame to it within the current HTML page according to the parameter settings.

**S7-400 / S7-300**

The following example illustrates the situation where all the supplied S7 standard applets are used within one HTML page. You can see that the S7 applets in this case are embedded in an HTML table.

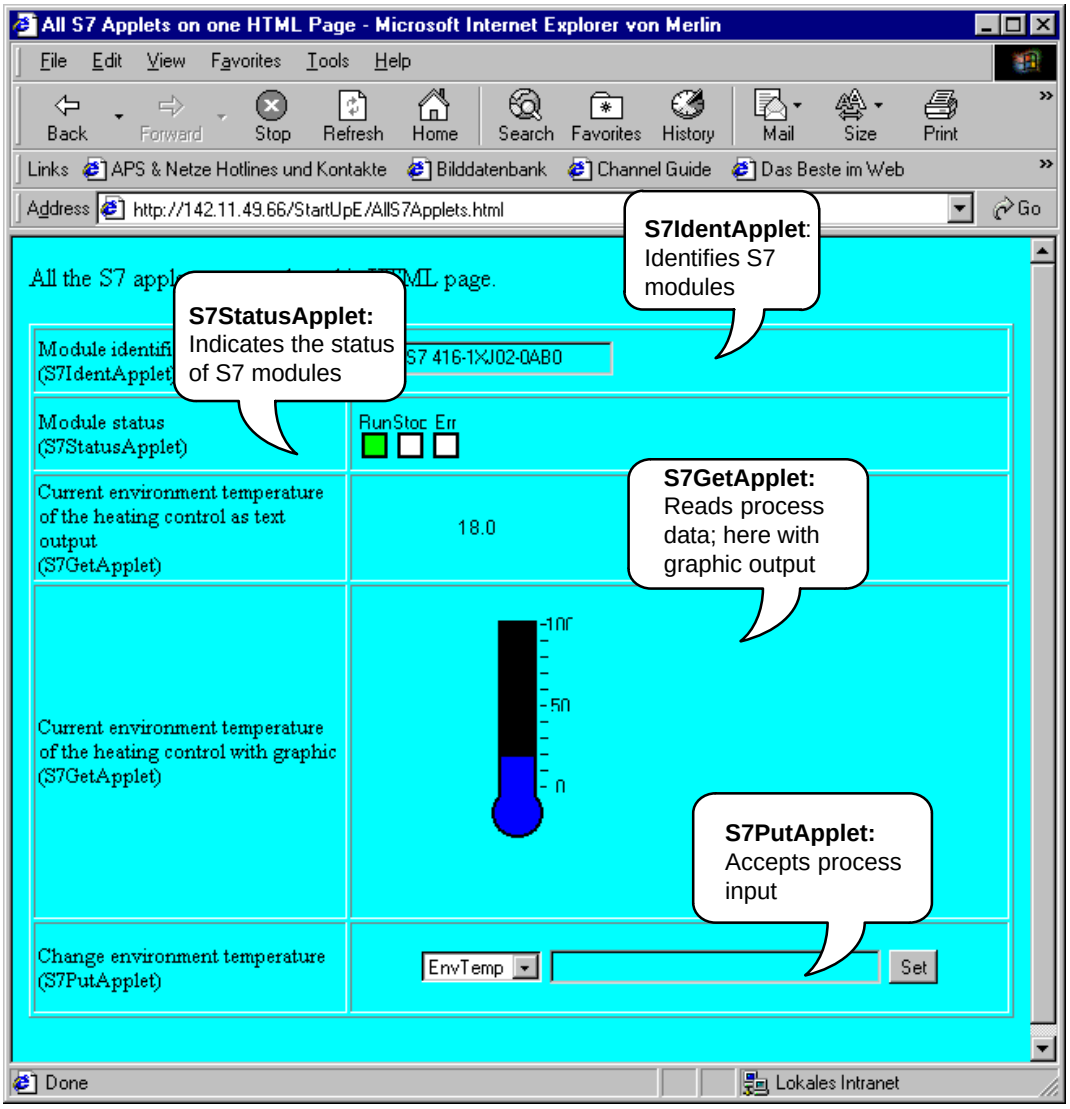



Figure 3-3

The table below contains further information. The chapters that follow describe in detail how to use and assign parameters for the S7 applets.

Table 3-1

| S7 Applet | Meaning |
|---|---|
| S7IdentApplet<br>(S7-300/400 only) | Identifies S7 modules based on the order number and version. |
| S7StatusApplet<br>(S7-300/400 only) | Displays the status of S7 modules<br>Example: Run/Stop |
| S7GetApplet | Reads process data cyclically, for example memory words or data in a data block.<br>The process data<br>• are addressed symbolically or in absolute form<br>• can be displayed graphically.<br>JavaBeans are used for graphic display. |
| S7PutApplet | Enters process data in the HTML pages and transfers/writes them to the controller, for example memory words or data in a data block.<br>The process data are addressed in symbolic or absolute form. |

## How the components interact

The S7 applets transfer special read or write jobs to the Advanced CP from the Web browser. When requested, the Advanced CP passes on the jobs to the relevant module or CPU.

The S7 applets can display messages about actions and errors in the Java Console (see Section 3.6). These messages provide you with information about the current status of program execution.

## The number of applet instances on an HTML page is limited

Remember that the number of applets that can be called in an HTML page is limited. The possible number depends on the Web browser you are using and your system environment (for example the operating system). You will find this maximum number in the documentation of your Web browser.

Increasing complexity with S7 beans:

If you require more complex displays, you should use the options provided by the S7 beans class library. If you create your own applets, you can access a larger number of process variables using S7 beans.  This will help you to avoid the restrictions relating to the number of applets.

## 3.4.1 Applet call and parameter assignment

### Calling S7 Applets in the HTML page

Like all Java programs, S7 applets have the file name extension ”class”. The applet call is embedded in the HTML page with a corresponding HTML tag (refer to the table below). The call to identify an S7 module in an S7 station is, for example, specified as follows in the applet tag:

**<Applet CODE=”de.siemens.simaticnet.itcp.applets.S7IdentApplet.class” ...>**

This assignment specifies the name or the address of the file with the applet. The notation used here indicates a relative address. All the applets are put together to make a jar file. If the CODEBASE attribute is also used in the call, the applet is located in the folder specified by CODEBASE.

**Example:**

The following call identifies an S7 module located in Rack 0 in Slot 3 of an S7 station. The information read is displayed on the HTML page in black script on a green background.

```
<APPLET CODE=”de.siemens.simaticnet.itcp.applets.S7IdentApplet.class”
CODEBASE=”/applets/”

ARCHIVE=”s7applets.jar, s7api.jar” NAME=”s7_MLFB” WIDTH=150
HEIGHT=35>

<PARAM name=”RACK” value=0>
<PARAM name=”SLOT” value=3>
<PARAM name=”BACKGROUNDCOLOR” value=”0x00FF00”>
 </APPLET>
```

You will find further examples of applications in the descriptions of the S7 applets themselves.

### General parameter assignment

Apart from the names of the S7 applets, you must specify certain general attributes and parameters. In addition to the general attributes and parameters required for each S7 applet, there are also function-dependent attributes and parameters. These are described with the S7 applets themselves.

The following tables show you the HTML tags, the attributes and parameters that are required with most S7 applets:

Table 3-2 General HTML tags when using applets

| HTML tag and attribute | Meaning |
|---|---|
| <APPLET...>...</APPLET> | This tag embeds an applet in an HTML document.<br>The applet is specified by attributes and parameters.<br>Example, see above. |
| <PARAM name="..." value="..."> | This tag identifies applet parameters. Each parameter is identified by a name and each parameter is assigned a value.<br>Example: <PARAM name="RACK" value=0> |

Table 3-3 General attributes of the S7 applets

| Attribute name | Type | Description |
|---|---|---|
| CODE | string | The attribute identifies the applet to be called.<br>The following must always be specified:<br>• Package path = constant<br>• Applet type = variable<br>Example:<br>CODE="de.siemens.simaticnet.itcp.applets.S7IdentApplet.class" |
| CODEBASE | string | This attribute identifies the path under which the applet file or the applet archive (see ARCHIVE) is stored.<br>Example: CODEBASE="/applets/" |
| ARCHIVE | string | This attribute names the applet archive in which the applet is contained.<br>Example: ARCHIVE="s7applets.jar, s7api.jar" |
| NAME | string | Unique applet name<br>Using this name, for example, outputs in the Java Console can be assigned to the applet.<br>Example: NAME="s7_MLFB" |
| WIDTH | int | Width of the display in the HTML page<br>This specifies the display width in the HTML page. The value selected must be high enough for the particular display.<br>Example: WIDTH=150 |
| HEIGHT | int | Height of the display in the HTML page.<br>This specifies the display width in the HTML page. The value selected must be high enough for the particular display.<br>Example: HEIGHT=35 |

Table 3-4 General parameters of the S7 applets

| Parameter name | Type | Description |
|---|---|---|
| BackGroundColor | string | Background color (24 bits RGB in hexadecimal format)<br>This parameter controls the color intensity for the RGB components.<br>For orientation, a selection of values is shown below:<br>white: 0xFFFFFF<br>black: 0x000000<br>red: 0xFF0000<br>green: 0x00FF00<br>blue: 0x0000FF<br>Note:<br>You can test the effects of the setting using the input tool (see Section 3.4.2). |

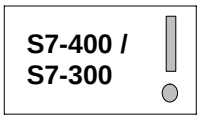
## 3.4.2 Parameter assignment tools

To support you when assigning parameters for the S7 applets, you can use various parameter assignment tools. These ensure that reliable, syntactically correct parameters are set for the S7 applets. The following tools are described:

- Input tool with the HTML editor
- Online parameter assignment for testing

The HTML pages available on the Manual Collection CD and on the Advanced CP in the Examples folder also provide a simple, efficient way to reuse existing call parameters by simply copying and pasting them.

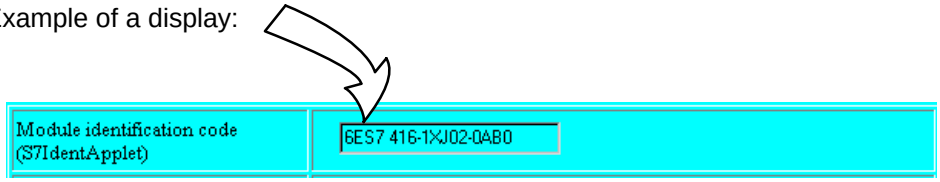
## 3.4.3 S7IdentApplet − Description




### Meaning

This is used to identify an S7 module in an S7 station. The applet reads the order number and the version of the specified S7 module.

Example of a display:




### Call tags

CODE="de.siemens.simaticnet.itcp.applets.S7IdentApplet.class"
CODEBASE="/applets/"
ARCHIVE="s7applets.jar, s7api.jar"

### Parameter assignment

In addition to the general parameters (see Section 3.4.1), the following parameters for the specific function must have values:

Table 3-5 Parameters specific to the applet

| Parameter name | Type | Description |
|---|---|---|
| Slot | byte | Slot number of the addressed module (1 to 18) |
| RACK | byte | Rack number of the addressed module (0 to 7) |

Table 3-6 Optional parameters specific to the applet

| | | |
|---|---|---|
| EDIT | bool | Online parameter assignment can be activated or deactivated.<br>Possible parameter setting<br>on = true<br>off = false<br>If the parameter in the applet call is not used, the default setting for online parameter assignment is **off (deactivated)!** |
| LANGUAGE | string | The language use for labels, messages, and diagnostic displays can be set permanently. Currently only German and English are supported. The two-character ISO-639 codes are used as parameters.<br><br>Possible settings<br>German = “de”<br>English = “en”<br><br>If the parameter is not used in the applet call, the language of the host system is used (or English if this language is not available). The language of the host system can be changed in Windows operating systems in the Control Panel – Regional Settings).<br>If an unsupported language is specified, the language of the host system is used (or English if this language is not available). |
| DEBUGLEVEL | int | Sets the level of detail for debug display in the Java console.<br><br>Possible settings<br>0 = no display<br>1 = display all messages<br>2 = display warning and error messages only<br>3 = display error messages only<br>4 = display only messages relating to fatal errors<br><br>if the parameter is not used in the applet call, level 3 is used; in other words, only error messages are displayed. |

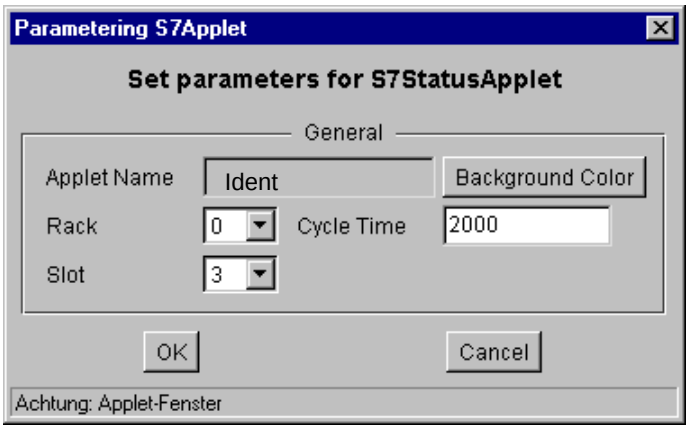
## Access rights

With the user name used for access, the following access right must be available (refer to the ”Edit User Entry” dialog in Section 3.4.2 ):

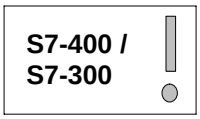
- “Query the order number of modules”

**Parameter assignment tools** (meaning and application see Section 3.4.2 )

**Online parameter assignment** is supported for test purposes. To use this option, double-click the output field to open the parameter assignment dialog.



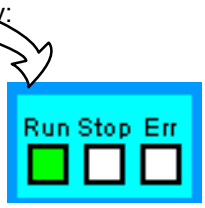
## 3.4.4 S7StatusApplet − Description




### Meaning

This queries the status information of the specified module.

Example of a display:




### Data output / display

The data output/display is graphic with additional text. The following table provides more information.

Table 3-7

| Display / color | Additional text | Meaning |
|---|---|---|
| green | Run | The user program is running. |
| yellow | Stop | The user program has been stopped. |
| Gray | Unknown | The connection to the Advanced CP is established; waiting for response. |
| red: | Error | There is a module error message. |
| Blue | Error | There is no connection to the addressed connection. |

### Access rights

With the user name used for access, the following access right must be available (refer to /2/, Section 3.3 “Configuration”, “User” tab):

- “query the status of modules”;

### Call tags

CODE=“de.siemens.simaticnet.itcp.applets.S7StatusApplet.class”
CODEBASE=“/applets/”
ARCHIVE=“s7applets.jar, s7api.jar”

### Parameter assignment

In addition to the general parameters (see Section 3.4.1), the following parameters for the specific function must have values:

Table 3-8 Parameters specific to the applet

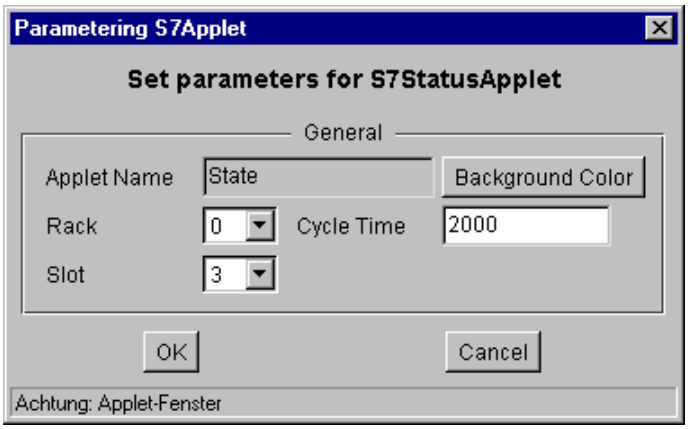
| Parameter name | Type | Description |
|---|---|---|
| Slot | byte | Slot number of the addressed module; (1−18) |
| RACK | byte | Rack number of the addressed module; (0−7) |
| CYCLETIME (cycle time) | int | Cycle time for the read job, specified in milliseconds. >5000 (recommended value) |

Table 3-9 Optional parameters specific to the applet

| | | |
|---|---|---|
| EDIT | bool | Online parameter assignment can be activated or deactivated.<br>Possible parameter setting<br>on = true<br>off = false<br>If the parameter in the applet call is not used, the default setting for online parameter assignment is **off (deactivated)!** |
| LANGUAGE | string | The language use for labels, messages, and diagnostic displays can be set permanently. Currently only German and English are supported. The two-character ISO-639 codes are used as parameters.<br><br>Possible settings<br>German = “de”<br>English = “en”<br><br>If the parameter is not used in the applet call, the language of the host system is used (or English if this language is not available). The language of the host system can be changed in Windows operating systems in the Control Panel – Regional Settings).<br>If an unsupported language is specified, the language of the host system is used (or English if this language is not available). |
| DEBUGLEVEL | int | Sets the level of detail for debug display in the Java console.<br><br>Possible settings<br>0 = no display<br>1 = display all messages<br>2 = display warning and error messages only<br>3 = display error messages only<br>4 = display only messages relating to fatal errors<br><br>if the parameter is not used in the applet call, level 3 is used; in other words, only error messages are displayed. |

**Parameter assignment tools** (meaning and application see Section 3.4.2 )

**Online parameter assignment** is supported for test purposes. To use this option, double-click the output field to open the parameter assignment dialog.



## 3.4.5 S7GetApplet − Description

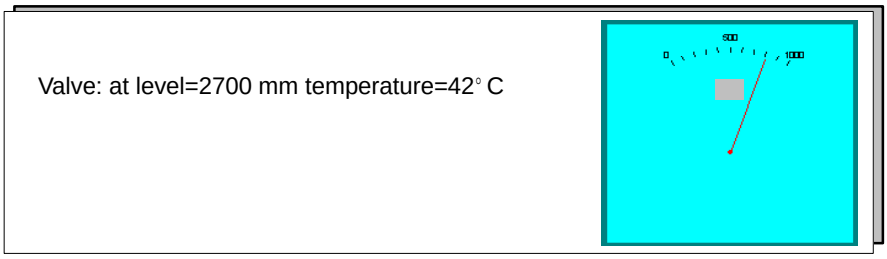
### Meaning

The applet reads data or data areas cyclically according to the parameter assignment from the S7 CPU. You can name the variables symbolically (not with the S7-200) or by specifying addresses.

Using a format string that is explained later, you specify the output form of the data.

The process values can be displayed either numerically or graphically.

Example of a display (numeric and graphic):




How to use graphic display and the JavaBeans is explained in Chapter 4.

### Initial situation

The variables can be identified using symbolic names or by their addresses.

Symbolic access (not with S7-200) requires suitably configured symbols on the Advanced CP. If symbolic access is used, access rights are checked according to the variable configuration (see Section 3.5).

### Access rights

**S7-400 / S7-300**

With the user name used for access, the following access right must be available (refer to /2/, Section 3.3 “Configuration”, “User” tab):

- ”Use the symbol table” (with symbolic access)
- ”Read variables using absolute addresses” (with absolute access)

### Call tags

CODE=“de.siemens.simaticnet.itcp.applets.S7GetApplet.class”
CODEBASE=“/applets/”
ARCHIVE=“s7applets.jar, s7api.jar”

## Parameter assignment

In addition to the general parameters (see Section 3.4.1), the following parameters for the specific function must have values:

Table 3-10 Parameters specific to the applet

| Parameter name | Type | Description |
|---|---|---|
| Slot | byte | Slot number of the addressed module (1 to 18)<br>(with S7-200 always R/S=0/0 ) |
| RACK | byte | Rack number of the addressed module (0 to 7)<br>(with S7-200 always R/S=0/0 ) |
| CYCLETIME | int | Cycle time for the read job, specified in milliseconds.<br>Range of values: >5000 (recommended) |
| FORMAT | string | The character string in the format parameter specifies how the read variable values will be displayed (example and explanation see below). |

Table 3-11 Optional applet-specific parameters − using JavaBeans

| Parameter name | Type | Description |
|---|---|---|
| DISPLAY | string | Identifies a JavaBean that can be used for graphic data display.<br>The currently possible values are as follows (see also Table 4-2, Page 58):<br>• CLTacho<br>• CLLevel<br>• CLThermo |
| MINVAL | int, float | MINVAL and MAXVAL specify the range for formatting graphically displayed variables. |
| MAXVAL | int, float | |
| DIMENSION | string | Here, you can specify a physical unit. |
| EDIT | bool | Online parameter assignment can be activated or deactivated.<br>Possible parameter setting<br>on = true<br>off = false<br>If the parameter in the applet call is not used, the default setting for online parameter assignment is **off (deactivated)!** |

Table 3-11 Optional applet-specific parameters − using JavaBeans, continued

| Parameter name | Type | Description |
|---|---|---|
| LANGUAGE | string | The language use for labels, messages, and diagnostic displays can be set permanently. Currently only German and English are supported. The two-character ISO-639 codes are used as parameters.<br><br>Possible settings<br>German = “de”<br>English = “en”<br><br>If the parameter is not used in the applet call, the language of the host system is used (or English if this language is not available). The language of the host system can be changed in Windows operating systems in the Control Panel – Regional Settings).<br>If an unsupported language is specified, the language of the host system is used (or English if this language is not available). |
| DEBUGLEVEL | int | Sets the level of detail for debug display in the Java console.<br><br>Possible settings<br>0 = no display<br>1 = display all messages<br>2 = display warning and error messages only<br>3 = display error messages only<br>4 = display only messages relating to fatal errors<br><br>if the parameter is not used in the applet call, level 3 is used; in other words, only error messages are displayed. |

Table 3-12 Parameters for symbolic variable addressing (**alternative** to absolute addressing − **only with S7-300/400**)

| Parameter name | Type | Description |
|---|---|---|
| SYMBOL | string | Name of the S7 variable symbol<br>The variable must be created with the STEP 7 symbol editor and configured for access via the Advanced CP (see Section 3.5). |

Table 3-13 Parameters for indirect variable addressing using the ANY pointer (**as an alternative** to SYMBOL)

| Parameter name | Type | Description |
|---|---|---|
| VARTYPE<br>(data type) | int | This variable type coding in the ANY pointer indicates the data type of the variable to be read; possible entries are as follows:<br>0x02 BYTE bytes (8 bits)<br>0x01 BOOL data type BOOL (1 bit)<br>0x03 CHAR characters (8 bits)<br>0x04 WORD words (16 bits)<br>0x05 INT integers (16 bits)<br>0x06 DWORD double words (32 bits)<br>0x07 DINT double integers (32 bits)<br>0x08 REAL floating-point numbers (32 bits)<br>0x09 DATE Date<br>0x0A TIME_OF_DAY (TOD) time of day<br>0x0B TIME time<br>0x13 STRING character string<br>This can be specified in decimal (for example 10) or hexadecimal (for example 0x0A).<br>Note:<br>The following data types are available and can be selected in the parameter dialog. Transferring these complex data types is, however, supported only by the S7 beans (see Chapter 4). Based on the S5 or S7 format description (see STEP 7 online help), these formats can then be decoded and processed further by the program.<br>0x0C S5TIME data type S5TIME<br>0x0E DATE_AND_TIME (DT) date and time (64 bits)<br>0x1C COUNTER counters<br>0x1D TIMER timers<br>**Note:**<br>The information here applies to the S7−300/400; for the S7−200 see /3/ |
| VARCNT<br>(repetition factor) | int | Number of variables to be read;<br>With this information, you can specify whether a variable or contiguous variable area is transferred. STEP 7 identifies arrays and structures as a number (here using the repetition factor) of data types.<br>Example: If 10 words are to be transferred, the value 10 must be specified for the repetition factor and the value 04 for the data type. |
| VARAREA<br>(memory area) | int | Area coding for identifying the memory area;<br>0x81 I memory area of inputs<br>0x82 Q memory area of outputs<br>0x83 M bit memory area<br>0x84 DB data block<br>This can be specified in decimal (for example 131) or hexadecimal (for example 0x83).<br>**Note:**<br>The information here applies to the S7−300/400; for the S7−200 see /3/ |
| VARSUBAREA<br>(subarea) | int | Subarea coding; for example, by specifying the DB number.<br>(with S7-200 always value=1) |

Table 3-13 Parameters for indirect variable addressing using the ANY pointer, (Suite)(**as an alternative** to SYMBOL), continued

| Parameter name | Type | Description |
|---|---|---|
| VAROFFSET (byte address) | int | Specifies a byte offset.<br>This information can be used to address the variable within the specified memory area (VARAREA), for example, the memory bit number. |
| VARBITOFFSET (bit address) | int | Specifies a bit offset.<br>This information can be used to address the bit of a variable of the type BOOL. |

## Notes on the address information VARTYPE...VAROFFSET

To read the data, the Advanced CP uses the S7 function SFB14 (GET). To address variables, the data type ANY pointer must therefore be supplied to transfer parameters to the SFB.

You will find further information on SFB14 (GET) in the online help of STEP 7, in the appendix to the contents in “Format of the ANY Parameter Type”. You will find detailed information on the ANY pointer in the STEP 7 online help.

## Range of values and using the FORMAT parameter

The following identifiers can be used in the FORMAT parameter:

Table 3-14 Meaning of the Format parameter

| Identifier | Number of relevant bytes | Representation |
|---|---|---|
| \ | 0 | \; Indicates that the following character is an ID as listed in this table.<br>To display ”\”<br>the following entry is necessary: ”\\”<br>Example of a variable of the type integer: \I |
| S | 1 | Bit string<br>Interprets the assigned byte as a string of bits to be represented individually. Example of output: 01101110 |
| O | 1 | Octal |
| H | 1 | Hexadecimal |
| B | 1 | Unsigned byte |
| C | 1 | Signed byte |
| D | 4 | Unsigned 32 |
| L | 4 | Signed 32 |

Table 3-14 Meaning of the Format parameter, continued

| Identifier | Number of relevant bytes | Representation |
|---|---|---|
| W | 2 | Unsigned 16 |
| I | 2 | Signed 16 |
| F | 4 | Floating point |
| C | 1 | Character |
| X (n, string1, string2) | 1 | Binary value (n= position 0 to 7, string1 = character string for value 1; string 2 = character string for value 0).<br>If you want to use several binary values within a byte for output, the identifier Y must be used alternately!<br>**Note the following!**<br>Between the opening bracket and the first comma, there must be **no** blank, otherwise the entry is not recognized. |
| Y (n, on, off) | 0 | Binary value (n= position 0 to 7, string1 = character string for value 1; string 2 = character string for value 0).<br>Notes on the function: In contrast to ID X, the position counter is not incremented. Y is used when several binary values within a byte are to be output. The ID X is used only for the output of the last binary value within a byte.<br>**Note the following!**<br>Between the opening bracket and the first comma, there must be **no** blank, otherwise the entry is not recognized. |
| G | 1 | Increments the position counter by 1 byte without representation:<br>This ID is required to skip bytes in the variable string. This is necessary when empty bytes must be taken into account due to the data structure (for example words and bytes are defined alternately). |

## Interpretation of the FORMAT string

The character string in the format parameter specifies how the read variable values will be displayed (example see below). It is assumed that variable values are read in the form of byte strings.

In the representation, the character string in the format string is interpreted starting from the left and assigned to the variable string. With each value assigned and output, a position counter is incremented according to the specified ”number of relevant bytes” in Table 3-14.

Output continues until all the format IDs have been processed. If it is not possible to assign all bytes, there is no further output. If more format IDs are specified than can be assigned, output continues until the format IDs have all been processed.

**Note**

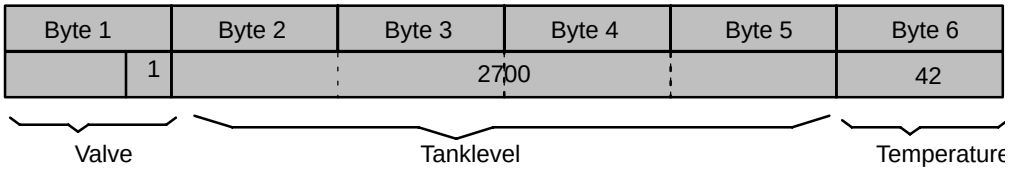
Make sure that the format string exactly matches the bytes in the variable string.

The following schematic illustrates the assignment and output based on an example of 3 variables.

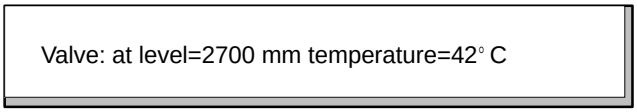
The variables are read as a contiguous string of 6 bytes from the S7-CPU. Following this, the format assignment assigns the various variable types and the variables are displayed.

**Format string**:valve: \X(0, open, closed)  tanklevel= \D mm temperature= \B °C

Variable string read (6 bytes):



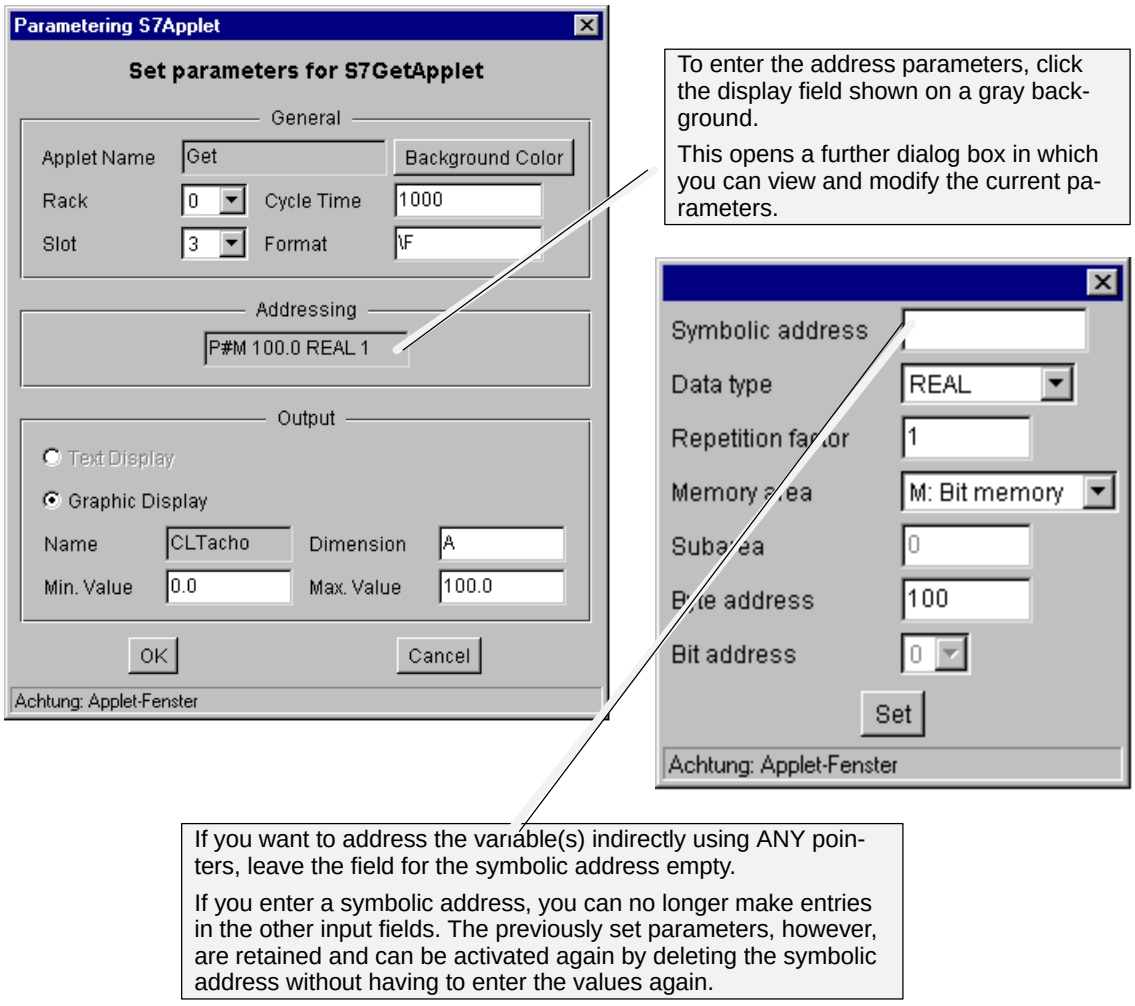

**Display on the HTML Page**




An **error message** (refer to output in the Java Console in Section 3.6) is only generated when there is an assignment conflict. Example: \D − a double word − is defined in the format string, the variable read, however, is only 2 bytes long.

**Parameter assignment tools** (meaning and application see Section 3.4.2 )

**Online parameter assignment** is supported for test purposes. To use this option, double-click the output field to open the parameter assignment dialog.



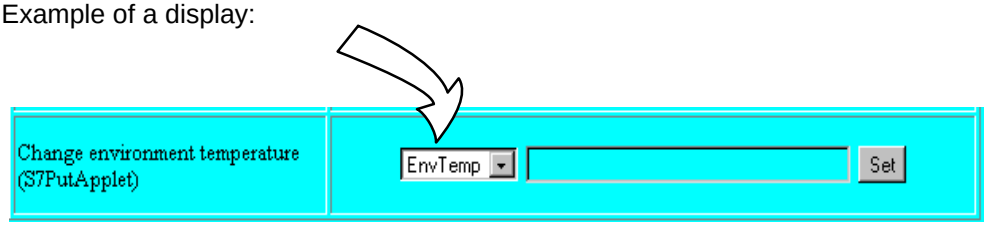
## 3.4.6 S7PutApplet − Description

### Meaning

This applet receives variable values entered by the user and transfers them to the S7-CPU.

According to the parameter settings in a format string, two fields are displayed in the Web browser in which you select the data area and enter the variable value. An additional button “Set” is used to trigger the write job.

Example of a display:




### Initial situation

The variables can be identified using symbolic names or by their addresses.

**S7-400 / S7-300**

Symbolic access is only possible if symbols were configured on the Advanced CP. If symbolic access is used, access rights are checked according to the variable configuration (see Section 3.5).

### Access rights

With the user name used for access, the following access right must be available (refer to /2/, Section 3.3 “Configuration”, “User” tab):

- ”Use the symbol table” (only with symbolic access)
- ”Write variables using absolute addresses” (only with absolute access)

### Call tags

CODE=“de.siemens.simaticnet.itcp.applets.S7PutApplet.class”
CODEBASE=“/applets/”
ARCHIVE=“s7applets.jar, s7api.jar”

### Parameter assignment

In addition to the general parameters (see Section 3.4.1), the following parameters for the specific function must have values:

Table 3-15 Parameters specific to the applet

| Parameter name | Type | Description |
|---|---|---|
| Slot | byte | Slot number of the addressed module (1 to 18)<br>(with S7-200 always R/S=0/0 ) |
| RACK | byte | Rack number of the addressed module (0 to 7)<br>(with S7-200 always R/S=0/0 ) |

Table 3-16 Optional parameters specific to the applet

| | | |
|---|---|---|
| EDIT | bool | Online parameter assignment can be activated or deactivated.<br>Possible parameter setting<br>on = true<br>off = false<br>If the parameter in the applet call is not used, the default setting for online parameter assignment is **off (deactivated)!** |
| LANGUAGE | string | The language use for labels, messages, and diagnostic displays can be set permanently. Currently only German and English are supported. The two-character ISO-639 codes are used as parameters.<br><br>Possible settings<br>German = “de”<br>English = “en”<br><br>If the parameter is not used in the applet call, the language of the host system is used (or English if this language is not available). The language of the host system can be changed in Windows operating systems in the Control Panel – Regional Settings).<br>If an unsupported language is specified, the language of the host system is used (or English if this language is not available). |
| DEBUGLEVEL | int | Sets the level of detail for debug display in the Java console.<br><br>Possible settings<br>0 = no display<br>1 = display all messages<br>2 = display warning and error messages only<br>3 = display error messages only<br>4 = display only messages relating to fatal errors<br><br>if the parameter is not used in the applet call, level 3 is used; in other words, only error messages are displayed. |

Table 3-17 Parameters for symbolic data addressing **(S7-300/400 only)**

| Parameter name | Type | Description |
|---|---|---|
| SYMBOLNUM | int | Number of variables that can be entered as symbols |
| SYMBOLn *) | string | Name of the S7 variable symbol. The name appears in the list box of the first text field.<br>The variable must be created with the STEP 7 symbol editor and configured for access via the Advanced CP (see Section 3.5). |
| SYMFORMATn *) | string | The character string in the format parameter specifies how the entered variable values will be interpreted.<br>The assignment of the entry to S7 variables is made according to the value n (example see below). |

*) Key: “n” indicates consecutive numbering of the variables addressed as symbols within a call;

Table 3-18 Parameters for addressing data using absolute addresses

| Parameter name | Type | Description |
|---|---|---|
| VARNUM | int | Number of variables to be written; |
| VARNAMEn *) | string | Variable name of a variable addressed indirectly. The name appears in the display in the list box of the first text field.<br>Maximum 256 characters |
| VARTYPEn *)<br>(data type) | int | Variable type coding in the ANY pointer;<br>The variable type coding in the ANY pointer indicates the data type of the variables to be written; the following entries are possible:<br>0x02 BYTE bytes (8 bits)<br>0x01 BOOL data type BOOL (1 bit)<br>0x03 CHAR characters (8 bits)<br>0x04 WORD words (16 bits)<br>0x05 INT integers (16 bits)<br>0x06 DWORD words (32 bits)<br>0x07 DINT double integers (32 bits)<br>0x08 REAL floating-point numbers (32 bits)<br>0x09 DATE date<br>0x0A TIME_OF_DAY (TOD) time of day<br>0x0B TIME time<br>0x13 STRING character string<br>Note:<br>The following data types are available and can be selected in the parameter dialog. Transferring these complex data types is, however, supported only by the S7 beans (see Chapter 4). Based on the S5 or S7 format description (see STEP 7 online help), these formats can then be decoded and processed further by the program.<br>0x0C S5TIME data type S5TIME<br>0x0E DATE_AND_TIME (DT) date and time (64 bits)<br>0x1C COUNTER counters<br>0x1D TIMER timers<br>**Note:**<br>The information here applies to the S7−300/400; for the S7−200 see /3/ |

Table 3-18 Parameters for addressing data using absolute addresses, continued

| Parameter name | Type | Description |
|---|---|---|
| VARAREAn *)<br>(memory area) | int | Area coding for identifying the memory area;<br>0x81 I memory area of inputs<br>0x82 Q memory area of outputs<br>0x83 M bit memory area<br>0x84 DB data block<br>This can be specified in decimal (for example 131) or hexadecimal (for example 0x83).<br>**Note:**<br>The information here applies to the S7−300/400; for the S7−200 see /3/ |
| VARSUBAREAn *)<br>(subarea) | int | Subarea coding; for example, by specifying the DB number. |
| VAROFFSETn *)<br>(byte address) | int | Specifies a byte offset.<br>This information can be used to address the variable within the specified memory area (VARAREA), for example, the memory bit number. |
| VARBITOFFSET<br>(bit address) | int | Specifies a bit offset.<br>This information can be used to address the bit of a variable of the type BOOL. |
| VARFORMATn *) | string | The character string in the format parameter specifies how the entered variable values will be interpreted.<br>Depending on what is specified, the information is assigned to the S7 variable in the input box. |

*) Legend: “n” stands for a continuous numbering of the indirectly (via ANY pointer) addressed variables within a call beginning with “1”.

## Range of values for the FORMAT parameter

The following identifiers can be used in the FORMAT parameter:

Table 3-19 Meaning of the Format parameter

| Identifier | Number of bytes used | The input string is interpreted as follows |
|---|---|---|
| S | 1 | Bit string |
| O | 1 | Octal |
| H | 1 | Hexadecimal |
| B | 1 | Unsigned byte |
| C | 1 | Signed byte |
| D | 4 | Unsigned 32 |
| L | 4 | Signed 32 |
| W | 2 | Unsigned 16 |

Table 3-19 Meaning of the Format parameter, continued

| Identifier | Number of bytes used | The input string is interpreted as follows |
|---|---|---|
| I | 2 | Signed 16 |
| C | n | Character string |
| F | 4 | Floating point 32 |

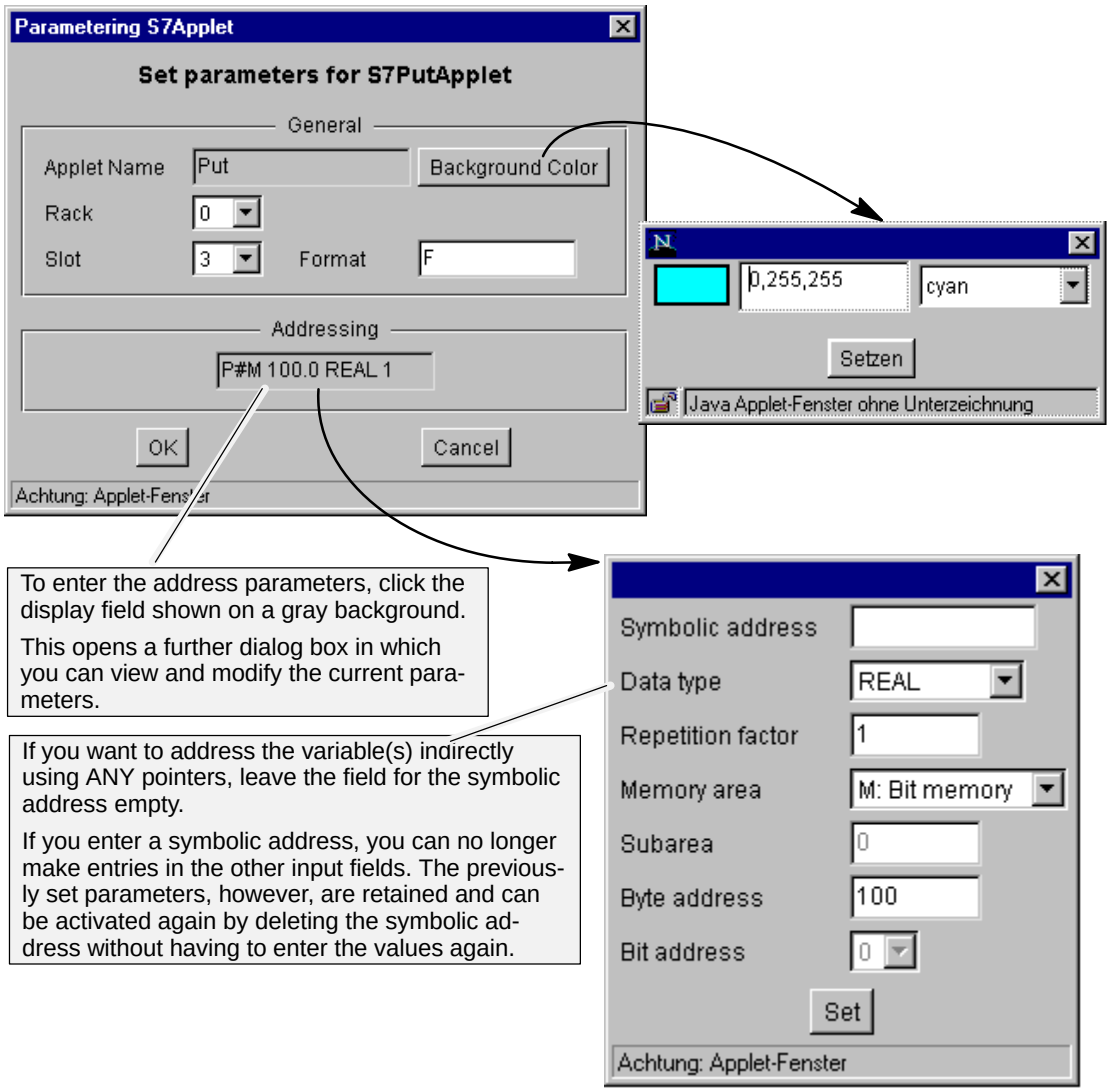
**Note**

In contrast to the S7GetApplet, the ”\” ID is not required with the S7PutApplet.

**Parameter assignment tools** (meaning and application see Section 3.4.2 )

- **Online parameter assignment** is supported for test purposes. The EDIT applet parameter must be set to true.

  Double-click on the display area of the PutApplet (**caution! not** in the input box for the variable value!) to open the following dialog box:




To enter the address parameters, click the display field shown on a gray background.

This opens a further dialog box in which you can view and modify the current parameters.

If you want to address the variable(s) indirectly using ANY pointers, leave the field for the symbolic address empty.

If you enter a symbolic address, you can no longer make entries in the other input fields. The previously set parameters, however, are retained and can be activated again by deleting the symbolic address without having to enter the values again.

## 3.5 Configuring variables for access using symbols

S7-400 / S7-300

### Symbolic variable access helps to avoid configuration errors

The S7 applets S7GetApplet and S7PutApplet allow convenient access to variables using symbolic names familiar from LAD/FBD/STL programming with the symbol table. This saves you time and headache trying to work with absolute addresses.

The following example shows how a name is assigned to data block DB100 in the symbol table.

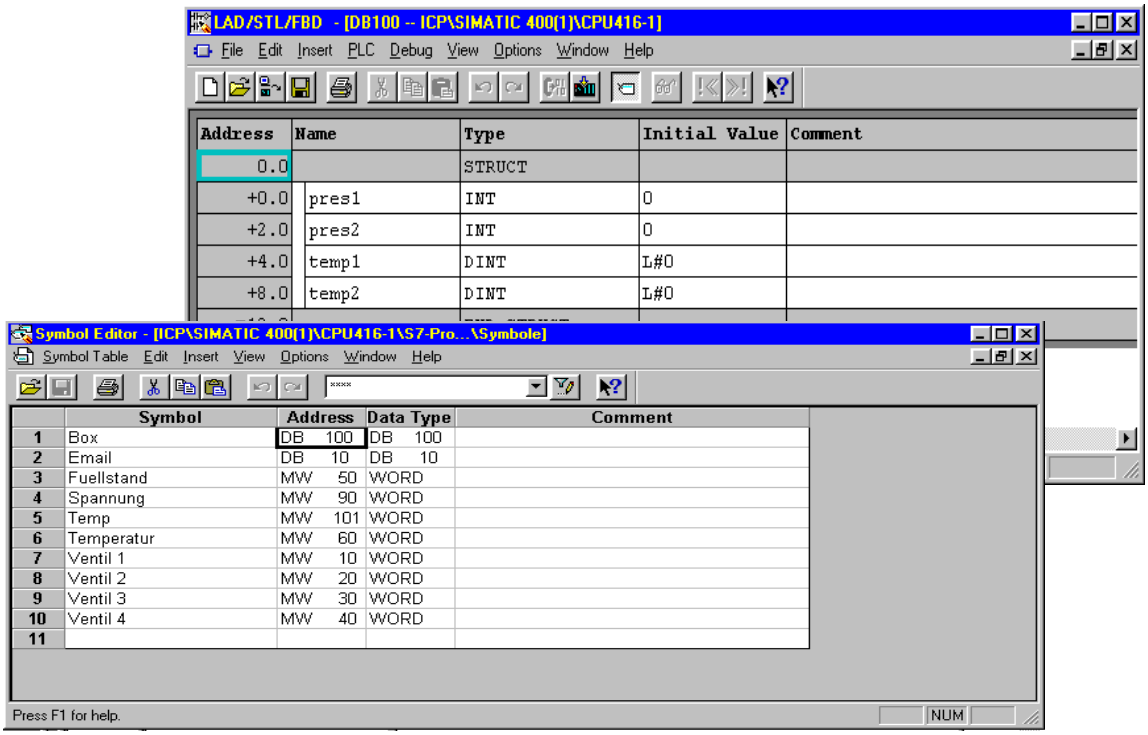



Figure 3-4

### Symbols for HTML pages

To access variables on the S7 CPU with Java applets using a Web browser, you must inform the Advanced CP of the names, addresses, and access rights of the variables. There are tabs for these settings in the Properties dialog of the Advanced CP.

The symbols that you specify here in the configuration must first be declared in the symbol table with the STEP 7 symbols editor. Following this, the assignments of the symbols to the variables have been established. With the configuration described here, you select the symbols you want to be able to access with the Web browser.

## Procedure

Open the properties dialog of your Advanced CP in STEP 7 HW Config.

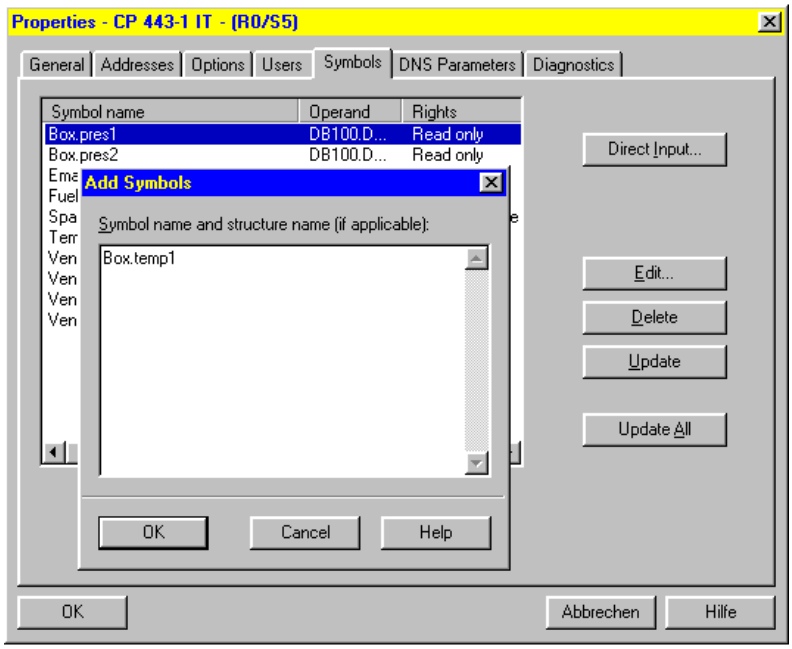



Figure 3-5

Enter the symbols of the variables or structure elements you want to be able to access with the Web browser. You can obtain detailed help on the dialogs at any time with the online help.

**Examples** (see also syntax rules for symbols in the online help)

- Simple variables:
  tank
  temperature
- Structure elements
  tank.pressure1
  tank.temperature

You must first enter the symbols in the symbol table using the symbols editor of STEP 7! Your entries will only be accepted if they match the entry in the symbol table.

## Assigning access rights

It is possible to assign access rights to the variables declared as symbols and these rights will be checked when the symbol is accessed. To set access rights, select the ”Edit...” button.

**Notice**

Access restrictions set here cannot be overridden by the EDIT applet parameter.

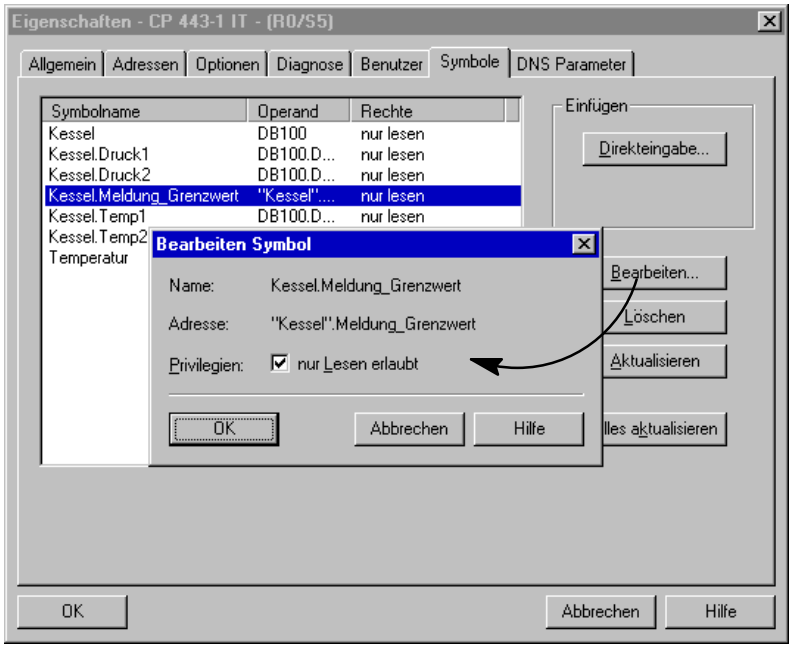



Figure 3-6

**Note**

If you use the symbolic names in the S7 beans or S7 applets, you will need to append the rack/slot number in parentheses.

Example: ”Temperature” variable needs to be specified in the S7Bean/S7Applet as ”Temperature(0/3)” if the variable is located in the CPU in rack/slot 0/3.

## Printing the list of variables

Along with the parameters of the Advanced CP module, you can print out a list of variables in HW Config.

## 3.6 Testing and using HTML pages

### Testing: Java Console

With the Web browser, you can follow and log the running of Java applets in a Java Console.

The S7 applets used in your HTML pages only display error messages in the Java Console. These messages provide you with information about unexpected reactions in the display of your HTML pages.

To display warnings or even all the available messages on the console, you can set the level of detail using the DEBUGLEVEL applet parameter (see Section 4.3):

Possible settings

0 = no display

1 = display all messages

2 = display warning and error messages only

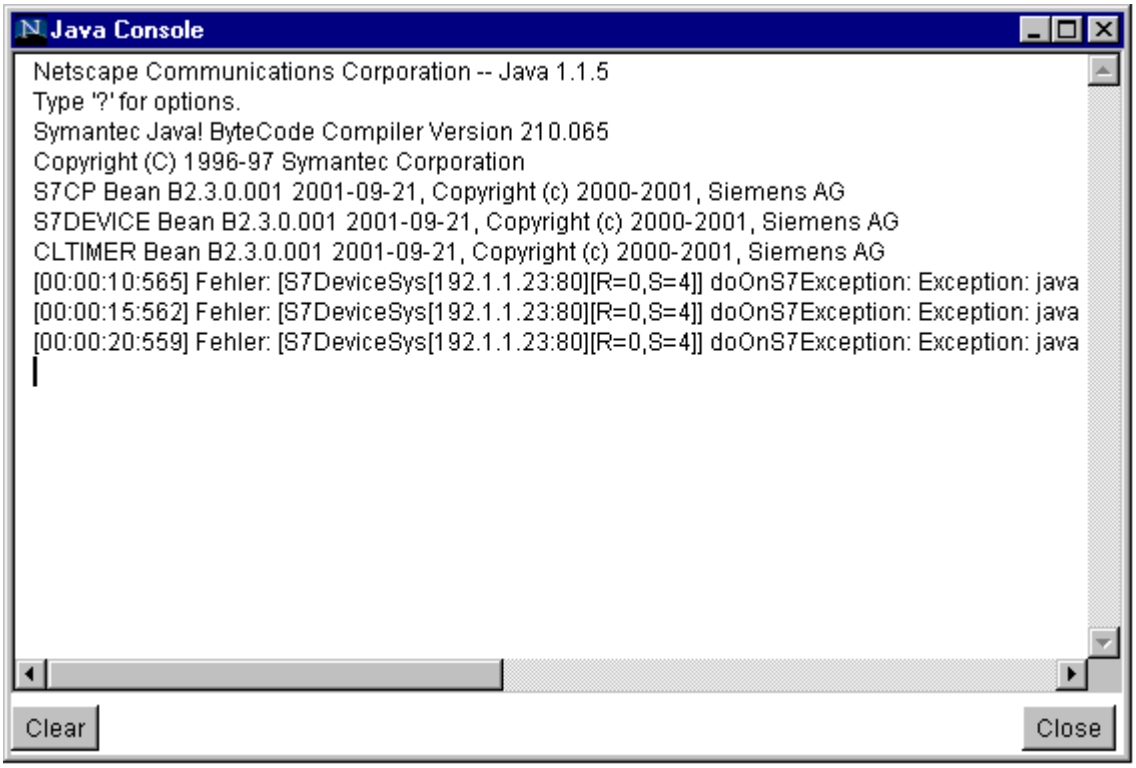
3 = display error messages only (default level)

4 = display only messages relating to fatal errors

if the parameter is not used in the applet call, level 3 is used; in other words, only error messages are displayed.

**Notice**

You should normally select Level 3 during normal operation! The other levels are intended only for test purposes.




### Loading: FTP for the transfer of HTML pages

Using FTP functions (see /2/ ), you can download the HTML pages to the Advanced CP and organize the files on the Advanced CP to suit your requirements.

## 3.7 JavaScript link to the S7 applets

You can also access the values of the S7 applets with a JavaScript thanks to special methods integrated in the S7 applets. These methods are listed below:

Table 3-20

| S7Applet | Method | Return / Transfer Type |
|---|---|---|
| S7GetApplet | getValue ( ) | java.lang.Object |
| S7PutApplet | setValue (param) | java.lang.Object |
| S7StatusApplet | getState ( ) *) | java.lang.String |
| S7IdentApplet | getIdent ( ) *) | java.lang.String |
| − All S7 Applets − | setRack (rack) | int |
| − All S7 Applets − | setSlot (slot) | int |

*) Methods only with the S7-300/400

## 3.8 Help on the S7 applets

This appendix explains how errors can occur when using the S7 applets.

If errors occur during operation, you will see messages displayed in the Java Console (see Section 2.2 and 3.6).

### All S7 applets

- The name of the applet class, the CODEBASE parameter, or the ARCHIVE parameter were not or incorrectly specified (check upper− /lowercase).
- The width and/or height of the applet was not specified or the value was too high or low.
- The syntax of the parameter tag <PARAM NAME=”...” VALUE=”...”> is wrong.
- You have forgotten a parameter or written it incorrectly.
- The BACKGROUND parameter for the background color of the applet is missing or is not in the valid range from 0x000000 to 0xFFFFFF.
- The user does not have rights to run the applet.
- The RACK and SLOT parameters for the rack number and the slot number do not match the rack and/or slot in which the module is actually located.

### Only S7StatusApplet and S7IdentApplet

- The addressed module is not a CPU / CP

### S7GETApplet and S7StatusApplet only

- No CYCLETIME parameter for the cycle time or is not an integer.

### S7GETApplet and S7PUTApplet only

- Symbol table not found
- A specified symbol was not found in the symbol table.
- The ANY pointer contains invalid values.
- The type of the specified parameter does not match the type expected (for example, integer, floating−point, string).

### S7GETApplet only

- The format string has a syntax error (for example, an unknown formatting character).
- The format string does not match the length of the data fetched from the CPU.
- The maximum and/or minimum value is not specified by the parameters MINVAL and/or MAXVAL if an S7 bean is used.
- The value of the MAXVAL parameter is less than or equal to the value in the MINVAL parameter.

### S7PUTApplet only

- The information in the SYMBOLNUM parameter and/or VARNUM does not match the actual symbol specified or the ANY pointer.
- A specified symbol has the read−only attribute in the symbol table

### S7StatusApplet only

- The addressed module is not capable of supplying its module status.
- If you want to display the status of more than one module and the cycle time selected is too short, you will often experience problems establishing the connection.

# 4 Individual Solutions with JavaBeans

In many situations, you will probably prefer to see a graphic visualization of the process values rather than a simple display of the numbers. In the plant pictures on operator control and monitoring systems, such display forms, for example level displays or temperature scales are normal.

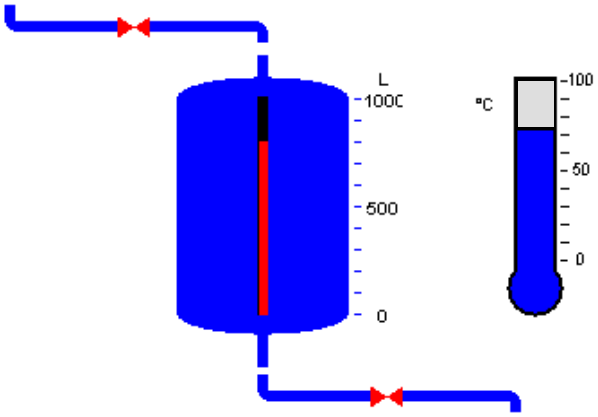



Figure 4-1

The JavaBeans concept in Java allows objects (Java components) to be created and linked simply to executable programs. JavaBeans also allow extremely flexible options for accessing process data on a SIMATIC S7 CPU.

As an Advanced CP user, you have an S7 beans class library (S7BeansAPI) available for Java programming. The object classes contained in this S7 beans class library can be used for object-oriented access to a variety of information on the SIMATIC S7 and can be used for graphic display of process variables.

## 4.1 JavaBeans concept and possible applications

### Standard application: You can use S7 beans along with the S7GetApplet

Simply by using the “S7GetApplet” of the Advanced CP, you can use a range of graphic elements from the S7 beans class library for display on your HTML page using the DISPLAY parameter (refer to the DISPLAY parameter in Section 3.4.5). The S7 beans to be used are marked in Table 4-2 in the following section.

### Extended applications: Configure S7 beans using “Builder Tools”

To be able to use the full range of S7 beans in the S7 bean class library, use builder tools in your Java programming environment.

With these builder tools you can create an object-oriented configuration for your application. In simplified form, this involves the following steps:

- Select the required S7 beans
- Link the S7 beans together to specify the data flow
- Specify the parameter settings for the S7 beans

Using this technique, you can configure and program complex process displays in your IDE with little effort.

For the experienced Java user, this provides almost unlimited options for further processing of the data acquired by the Advanced CP, for example, in databases or management information systems.

---

**Note**

When developing new applets and your own beans, check the run-time version in the file system and the development version in the Manual Collection.

---

## 4.2 The S7 Beans class library (S7BeansAPI)

### Application: Manual Collection

The S7 beans class library is in the Manual Collection. You can also download the current version from the Internet:

http://support.automation.siemens.com/WW/view/en/22964892

### The available S7 beans

The following tables provide you with an overview of the S7 beans supplied at the current time. Based on this table, you will get an impression of the configuration options provided by these S7 beans.

A distinction must be made between the following:

- S7 beans for devices

  These JavaBeans in the S7BeansAPI are provided for the modules and software objects that can be addressed the SIMATIC S7 rack. These provide the link in the program to the S7 beans for input and output (S7 beans for the client).

- S7 beans for the client

  These Java beans are available in the S7BeansAPI for graphic output of process data in process pictures on the client

- S7 utility beans for the client

  Utility beans are available for additional preparation of data. Since these are simply conversion functions, there is no direct graphic display by these beans.

  These S7 beans are in the JAR file s7util.jar.

Table 4-1 S7 beans for devices/ objects package = API (included in the JAR file s7api.jar)

| S7 bean | Function |
|---|---|
| S7CP | This bean represents the Advanced CP serving as the host. Any other Advanced CPs that exist must be addressed using S7Device.<br>This bean must be used with each applet for addressing and for saving the host address. |
| S7 Device | S7Device represents any intelligent S7 module such as a CPU, PROFIBUS CP, Ethernet CP, other Advanced CPs (however, under no circumstances the Advanced CP serving as host for the applets, to be addressed using the browser! ) |
| S7 Variable | This bean represents variables in the S7 CPU. |
| CLTimer | CLTimer is required for cyclic calling of methods of other beans.<br>Whenever you want to monitor the status of an S7 module or a process variable continuously (cyclically), you require this bean.<br>Note: CLTimer has **no** graphic representation |

Table 4-2 S7 beans for the client − package = GUI (included in the JAR file s7gui.jar)

| S7 Bean | Function | Can be used in S7GetApplet | Representation |
|---|---|---|---|
| CLTextIn | CLTextIn is a bean for entering text.<br>This text can be passed on to the S7 variable bean. | no | − Input field − |
| CLTextOut | CLTextOut is a bean for the textual output of values of process variables of the S7 variable bean | no |  |
| CLIdentOut<br>(S7-300/400 only) | CLIdentOut is a bean required for the textual display of an identification number of an Advanced CP or module using the S7 CP bean or s7 device bean.<br>Note:<br>Not all modules support this service. | no |  |
| CLStateLED<br>(S7-300/400 only) | CLStateLED is a bean for graphic representation of the status of an Advanced CP or a module.<br>This is represented by the color of the LED:<br>• green: RUN<br>• yellow: STOP<br>• red: Error message from the module<br>• blue: Connection error<br>• gray: Status unknown<br>Note:<br>Not all modules support this service. | no |  |
| CLState3LED<br>(S7-300/400 only) | CLState3LED is a bean for graphic representation of the status of an Advanced CP or a module.<br>The representation is in the form of three LEDs:<br>• green: RUN<br>• yellow: STOP<br>• red: Error message from the module<br>• blue: Connection error<br>• gray: Status unknown<br>Note:<br>Only intelligent modules support this service; I/O modules, for example, do not. | no |  |

Table 4-2 S7 beans for the client, continued − package = GUI (included in the JAR file s7gui.jar)

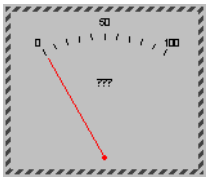
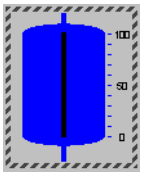
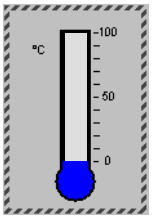
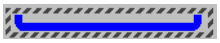
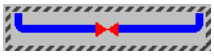
| S7 Bean | Function | Can be used in S7GetApplet | Representation |
|---|---|---|---|
| CLTacho | CLTacho is a bean for graphic representation of a pointer instrument.<br>The pointer represents the value of a process variable.<br>It is also possible to visualize a value reaching an upper or lower limit value with an LED display.<br>The size of the bean is scalable. | yes |  |
| CLLevel | CLLevel is a bean for graphic representation of the level of a process variable.<br>It is also possible to visualize a value reaching an upper or lower limit value with an LED display.<br>The size of the bean is scalable. | yes |  |
| CLThermo | CLThermo is a bean for graphic representation of a process variable as a temperature value.<br>It is also possible to visualize a value reaching an upper or lower limit value with an LED display.<br>The size of the bean is scalable. | yes |  |
| CLPipe | CLPipe is a bean for graphic display or a horizontal or vertical pipe.<br>The color of the pipe changes with a Boolean value.<br>The size of the bean is scalable. | no |  |
| CLValve | CLValve is a bean for graphic representation of a valve. The valve with its inlet can be displayed horizontally or vertically.<br>The valve is opened and closed by setting a Boolean value. | no |  |

Table 4-3 S7 Utility Beans Package = UTIL (included in the JAR file s7util.jar)

| S7 Utility Bean | Function |
|---|---|
| COUNTER | Supplies the counter reading (for example, C1) as a string in the format C#347. |
| TIMER | Supplies the value of timers (for example, T1) as a string in the format<br>S5T#1h3m2s. |
| DATE | Supplies the S7 type DATE as a string in the format D#2000−12−31. |
| TIME | Supplies the S7 type TIME as a string in the format T#9h6m6s. |
| DATEandTIME | Supplies the S7 type DATE_AND_TIME as a string in the format DT#00−12−31−12:31:47.487. |
| TIMEofDAY | Supplies the S7 type Time Of Day as a string in the format TOD#9:6:6.127. |
| S5TIME | Supplies the S7 type S5TIME as a string in the format S5T#1h3m2s. |
| ConvertNumberSystem | Supplies a decimal number optionally in hexadecimal, octal, or binary format (as a string).<br>Decimal numbers can also be converted to BCD numbers and vice versa. |

## 4.3 Linking S7 beans

Before your S7 beans can interact in your application, they must be connected together.

As a general introduction, we will first show you an overview of useful connections between the beans.

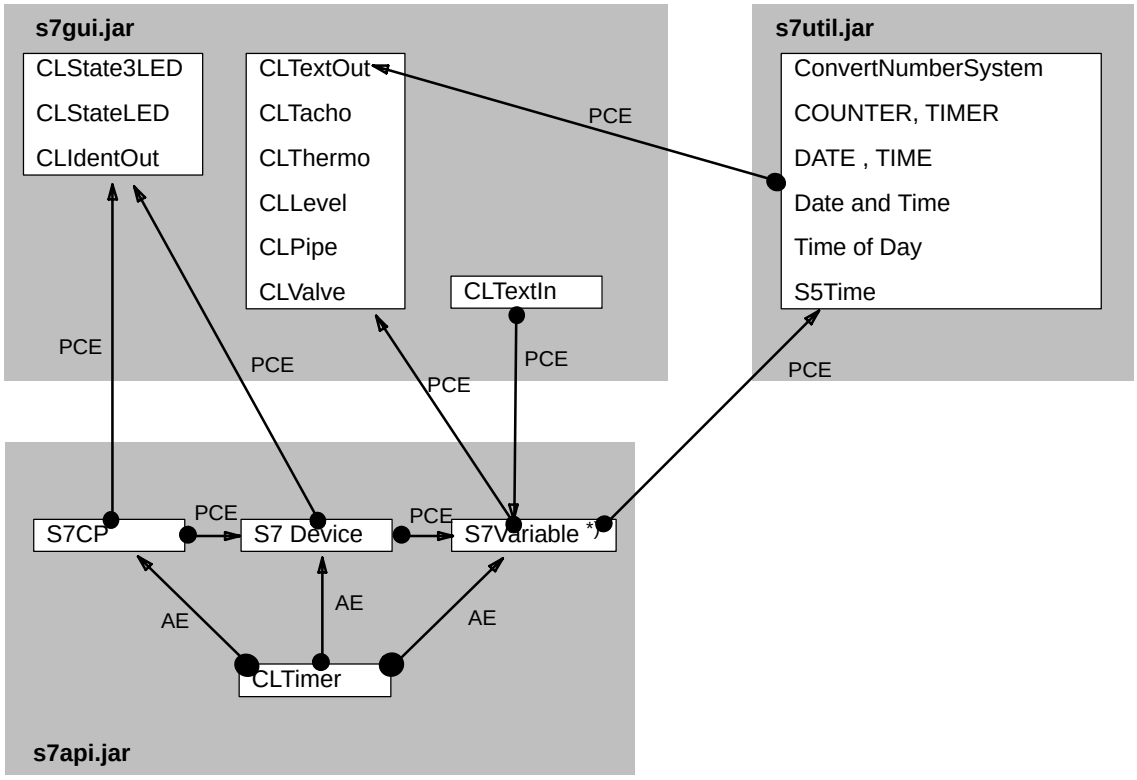
The following graphic shows which connections are possible; the abbreviations used are

AE: ActionEvent

PCE: PropertyChangeEvent

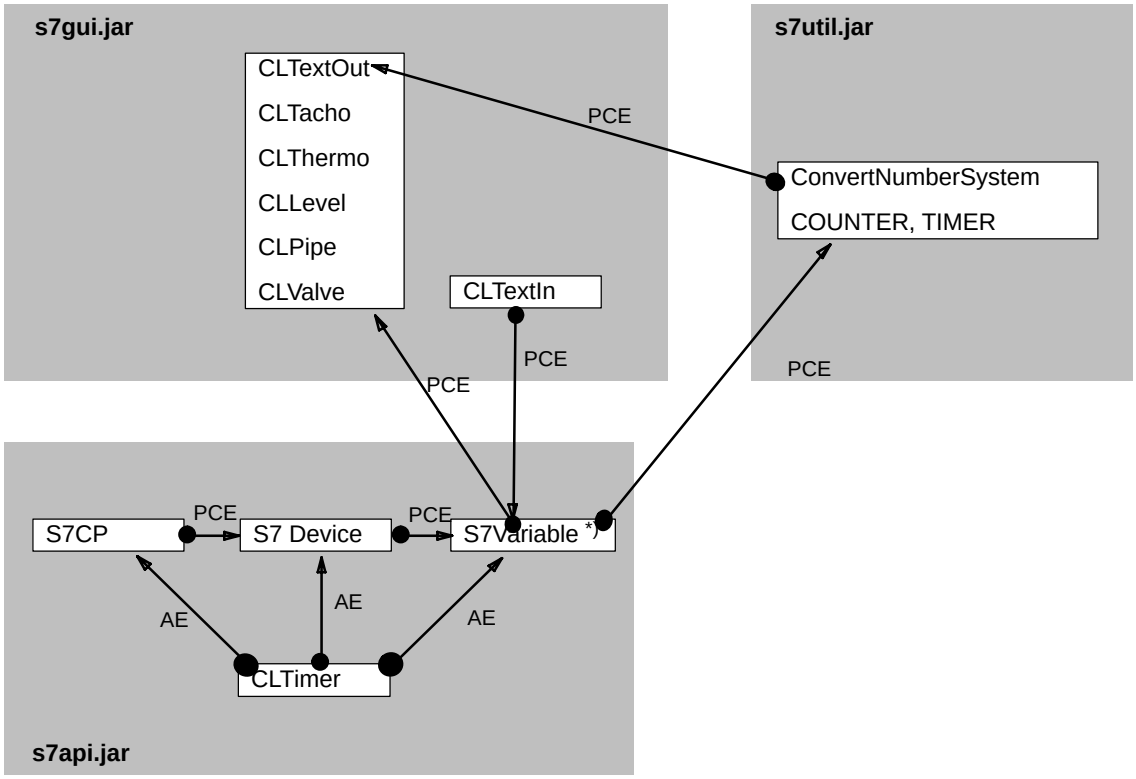
The producer of the event is shown at the end of the arrow with the dot and the tip of the arrow points to the listener.

### Linking S7 beans on the CP 343-1 IT and CP 443-1 IT




*) For more information on the variable types with S7-300 / S7-400, please refer to tables in Chapter

## Linking S7 Beans on the CP 243-1 IT




*) For more information on the variable types with S7-200, please refer to tables in Chapter

# 5 S7BeansAPI − Interface Description

This chapter describes the interfaces of the “invisible” S7API and S7Beans for devices. You will learn how to create instances of the beans (Constructor calls) and which access methods exist.

The S7 beans for input and output of values pass or receive their values using a PropertyChangeEvent (see Section 4.3: Connecting S7 Beans).

For more information, you should also read the JavaDoc for the S7BeansAPI supplied on the Manual Collection CD.

## 5.1 S7API methods

As of S7BeansAPI version V2.3, the **`de.siemens.simaticnet.itcp.api`** package contains the new class **S7Api**. This class contains a series of useful auxiliary functions that are introduced below.

### 5.1.1 Controlling the language used

With the **S7Api.setLocale(String newLanguage, String newCountry)** method, the language used for messages and labels can be set at run time. The **newLanguage** parameter is a language code defined according to ISO−639 and **newCountry** is the country code according to ISO−3166. The S7BeanApi ships with English and German resource files. These language files are located in the relevant JAR archives and have the extension *.properties. Based on these resource files, you can also create your own translations so that you can use languages other than English and German. Below there is an example of a **setLocale()** method call as you could use it in the **init()** method of one of your applets:

```
S7Api.setLocale(“de”, ”);
```

This call sets the language of the S7BeansAPI to neutral German, since no country code is specified.

---

**Note**

If no language is set, the S7BeansAPI uses the IDE language of the host if this exists just as in the development phase of the IDE.

---

### 5.1.2 Setting the level of detail for debug output

To avoid resource problems, as of S7BeansAPI version V2.3, the default setting is that debug messages in the Java console are displayed only for errors. With the new S7Api method **`setDebugLevel(int level)`**, you can specify the level of detail at run time. The following values can be passed for the **level** integer parameter:

0 = VOID_LEVEL: Display no messages

1 = LOG_LEVEL: All possible messages

2 = WARN_LEVEL: Display warnings and messages only

3 = ERR_LEVEL: Display error messages only (default)

4 = FATAL_LEVEL: Display fatal errors only

With the following call, for example, only warnings and error messages are displayed:

```
S7Api.setDebugLevel(2);
```

**Note**

The messages (warnings, errors etc.) are displayed with time stamp and object information. This makes it easier to analyze problems and the run time response.

### 5.1.3 Terminating the API mechanisms

To avoid resource problems when using the applets developed with the beans, all resources should be released when the relevant HTML page is exited and all threads stopped. Since the S7BeansAPI uses static resources and threads internally for communication with the Advanced CP, the **terminate()** method was created in the **S7Api** class to release these resources and to stop all threads.

Terminate() can normally be called after releasing all its own resources in the destroy() method of the applet. Since some browsers (for example, Netscape) do not call the destroy() method immediately when the HTML page is exited, but only when the applet is removed from the History cache of the browser, it is advisable to release the resources in the stop() method and to call terminate() from the S7Api class immediately. This does, however, require reinitializing all resources in the start() method of the applet because the applet, for example under Netscape, is also stopped and started again by changing the browser window size.

# 5.2 S7 beans

## 5.2.1 S7CP

### Description

The S7 bean S7CP is the component for connecting to the CP 343-1 IT / CP 443-1 IT via which you can access the PLC.

(Note: From one applet you can only ever access **the** Advanced CP from which the applet was loaded (Sandbox); if there are other Advanced CPs in the PLC, you can only access these using the S7Device S7 bean).

### Creating an instance − Constructor call:

To create an instance of the S7CP, start the Standard Constructor and then set the properties **host** and **modulName**.

Constructor Call:

As an alternative, you can also call the following Constructor and set the parameters directly. Please note, that the parameters rack and slot must both be set to 0 (zero) in this case.

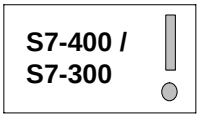
S7CP(String modulName, String host, int rack, int slot)

### Setting properties:

If you used the Standard Constructor, set the following properties:

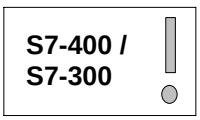
- setModulName(String modulName): Unique name for the instance.
- setHost(String host): IP address of the Advanced CP.

### Calling up the identification of the module (MLFB) (S7-300 / S7-400 only):




processIdent(): Initiates the MLFB number query. The method terminates immediately. As soon as the required information is available, it is sent to the listener using a PropertyChangeEvent (propertyName = identification).

### Calling up the status information of the module (S7-300 / S7-400 only):




processState(): Initiates the status information query. The method terminates immediately. As soon as the required information is available, it is sent to the listener using a PropertyChangeEvent (propertyName = state).

## 5.2.2 S7Device

### Description

The S7 bean S7Device represents an (intelligent) module, such as a CPU or a CP. An S7Device can also be a further Advanced CP via which you do not want to access the PLC from within the applet.

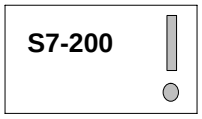
Each S7Device must be logged on with S7CP as PropertyChangeListener! You can log on several S7Devices at one S7CP.

### Creating an instance − Constructor call:

To create an instance of the S7Device, start the Standard Constructor and then set the properties *rack, slot* and *modulName.*

### Setting properties:

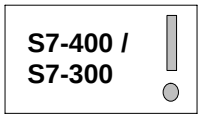
- setModulName(String modulName): Unique name for the instance.
- setRack(int rack): Number of the rack.
- setSlot(int slot): Number of the slot.




**Notice**

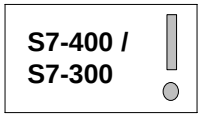
For an S7-200, please specify: rack =0; slot =0

### Calling up the Identification of the Module (MLFB) (S7-300 / S7-400 only):




processIdent(): Initiates the MLFB number query. The method terminates immediately. As soon as the required information is available, it is sent to the listener using a PropertyChangeEvent (propertyName = identification).

### Querying status information of the module:




processState(): Initiates the status information query. The method terminates immediately. As soon as the required information is available, it is sent to the listener using a PropertyChangeEvent (propertyName = state).

## 5.2.3 S7Variable

### Description

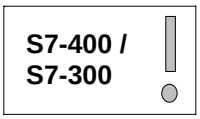
The S7 bean S7Variable represents a memory area on the CPU, that you either address using an ANY pointer or a symbolic address (the latter must be entered in the hardware configuration of the Advanced CP).

Each S7Variable must be logged on at the relevant S7Device as PropertyChangeListener. You can log on several S7Variables at one S7Device.

### Constructor call:

To create an instance of the S7Variable, start the Standard Constructor and then set the properties *symbolName* or *s7AnyPointer* and *variableName.*

### Setting properties:




- setSymbolName(String symbolName): Symbolic address.
- setS7Anypointer(S7Anypointer anypointer): ANY pointer address.
- setVariableName(String variableName): Unique name for this instance.

The symbolic addressing and the ANY pointer addressing are alternatives. If both are specified, the symbolic address has priority.

### Reading a memory area:

processGet(): Initiates the request to read a memory area. The method terminates immediately, the read job runs asynchronously. As soon as the requested information is available, this is sent to the listener using a PropertyChangeEvent (propertyName: the string entered in the *variableName* property).

getValue() object: Delivers the content of the memory area stored in the S7Variable. Remember, however, that this method does not necessarily supply the current content of the addressed memory area of the PLC! If no value was read previously with *processGet()*, no value will be returned here!

### Writing a memory areas:

setValue (newValue object): Initiates writing a value to the memory area of the CPU. The method terminates immediately, the write job is asynchronous.

### 5.2.4 CLTimer

#### Description

The CLTimer S7 bean creates an action event cyclically and can therefore be used as a trigger for the periodic reading of status information and memory areas.

Each S7 bean that executes an action periodically must be logged on at the CLTimer as an ActionListener. You must specify which method will be executed when the event arrives (for S7CP and S7Device, this is **processState()** a for S7Variable, this is **processGet()**). Several beans can be logged on with the CLTimer as ActionListeners. It is also possible to use several instances of CLTimer (for example, with different times).

#### Constructor call:

To create an instance of CLTimer, call the Standard Constructor and then, if necessary, the *delay property.*

#### Setting the property:

setDelay(int delay): Cycle time in milliseconds (default value is 5,000 ms).

#### Controlling CLTimer:

start(): Restarts the CLTimer after a stop() call. Following instantiation, the CLTimer starts automatically and does not need to be triggered with start().

stop(): Stops the CLTimer.

# A Property Change Events − Overview

The following table shows which S7Beans expect which Property Change Events (PCE) and the meaning or type that *NewValue* will have. A ' * ' in the “PCE name” column means that the name will not be checked; in other words, all incoming PCEs or PCEs that were not previously filtered out are interpreted in the same way.

Table 1-1

| Receiver bean | PCE name | Type / description of the expected PCE value (NewValue) |
|---|---|---|
| S7 Variable | inValue | Value to be set for the S7 from input bean<br>e.g. CLTextIn |
| | * | Value to be set for the S7 (produces warning in Java console) |
| CLPipe | * | Value to be displayed by bean (value > 0 = full) |
| CLHPipe | * | Value to be displayed by bean (value > 0 = full) |
| CLVPipe | * | Value to be displayed by bean (value > 0 = full) |
| CLLevel | * | Value to be displayed by bean (numeric value) |
| CLTacho | * | Value to be displayed by bean (numeric value) |
| CLThermo | * | Value to be displayed by bean (numeric value) |
| CLValve | * | Value to be displayed by bean (value > 0 = open) |
| CLTextOut | * | Value or text to be displayed by bean |
| ConvertNumberSystem | * | Value to be converted by bean (decimal number) |
| COUNTER | * | Value to be converted and then passed<br>The data type determines the direction:<br>byte[] : S7Variable −> COUNTER −> output bean<br>String: input bean −> COUNTER −> S7Variable |
| DATE *) | * | Value to be converted and then passed<br>The data type determines the direction:<br>Integer : S7Variable −> DATE −> output bean<br>String: input bean −> DATE −> S7Variable |
| DATEandTIME *) | * | Value to be converted and then passed<br>The data type determines the direction:<br>byte[]: S7Variable −> DATEandTIME −> output bean<br>String: input bean −> DATEandTIME −> S7Variable |

Table 1-1 , Fortsetzung

| Receiver bean | PCE name | Type / description of the expected PCE value (NewValue) |
|---|---|---|
| TIME *) | * | Value to be converted and then passed<br>The data type determines the direction:<br>Long: S7Variable −> TIME −> output bean<br>String: input bean −> TIME −> S7Variable |

*) these S7 types are not supported on an S7−200.

The following table shows which S7Beans which PCEs they send to their listeners themselves and the meaning or type that *NewValue* will have.

Table 1-2

| Sender | PCE name | Type / description of the sent PCE value (NewValue) |
|---|---|---|
| S7CP | S7CP | Object of the type S7CP |
| | slot | Slot as integer<br>(on S7−200 always value=0) |
| | rack | Rack as integer<br>(on S7−200 always value=0) |
| | host | Host name as string |
| | s7NetAddress | Object of the type S7NetAddress |
| | state | Status as integer |
| | identification | Identification as string |
| | moduleName | Module name as string |
| S7 Device | S7 Device | Object of the type S7Device |
| | slot | Slot as integer<br>(on S7−200 always value=0) |
| | rack | Rack as integer<br>(on S7−200 always value=0) |
| | host | Host name as string |
| | s7NetAddress | Object of the type S7NetAddress |
| | state | Status as integer |
| | identification | Identification as string |
| | moduleName | Module name as string |

Table 1-2 , continued

| Sender | PCE name | Type / description of the sent PCE value (NewValue) |
|---|---|---|
| S7 Variable | [Variable_name] | The new value of the variable after a “GET”.<br>Note: The variable name is assigned individually by the user during configuration of the S7Variable bean |
| CLTextIn | inValue | String newInValue |
| ConvertNumberSystem | ConvertNumberSystem | Converted string depending on setting, for example, as hex or octal string |
| COUNTER *) | COUNTER | Converted string |
| DATE *) | DATE | Converted string |
| DATEandTIME *) | DATEandTIME | Converted string |
| S5TIME *) | S5TIME | Converted string |
| TIME *) | TIME | Converted string |
| TIMEofDAY *) | TIMEofDAY | Converted string |
| TIMER | TIMER | Converted string |

*) these S7 types are not supported on an S7−200.

# B Comparison of S7 Types and Java Types

S7 types generally differ from the corresponding Java types in the range of values. To avoid problems here with signs etc., the corresponding types are as shown in the following table. If you use arrays (repetition factor (n) > 1), the elementary data types are used in Java. Such an array (for example, int[]) is considered as an object by Java which is not the case with other simple data types. This is particularly important for us for parameter passing, for example for setValue() of S7Variable.

Table 2-1

| S7 type | Java type | Java array |
|---|---|---|
| BOOL | Boolean | − |
| BYTE | Integer | int[] |
| CHAR | Character | char[] |
| WORD | Integer | int[] |
| INT | Integer | int[] |
| DWORD | Long | long[] |
| DINT | Long | long[] |
| REAL | Float | float[] |
| DATE *) | Integer | int[] |
| Time Of Day *) | Long | long[] |
| TIME *) | Long | long[] |
| S5TIME *) | byte[2] | byte[n*2] |
| Date And Time | byte[8] | byte[n*8] |
| STRING | String | − |
| COUNTER *) | byte[2] | byte[n*2] |
| TIMER | byte[2] | byte[n*2] |

*) these S7 types are not supported on an S7−200.

**Notice**

− An array of the type BOOL cannot be formed.

− An array of the type STRING cannot be formed.

# C References

## Locating Siemens literature

The order numbers for Siemens documentation can be found in the catalogs ”SIMATIC NET Industrial Communication, catalog IK PI” and ”SIMATIC Products for Totally Integrated Automation and Micro Automation, catalog ST 70”.

You can obtain these catalogs and any further information you require from your local SIEMENS office or national head office.

Some of the documents listed here are also in the Manual Collection supplied with every S7-CP.

Many SIMATIC NET manuals are available on the Internet pages of Siemens Customer Support for Automation:

http://support.automation.siemens.com/WW/view/de

Enter the ID of the relevant manual as a search key. The ID is shown below the literature name in brackets.

Manuals that are installed with the online documentation of the STEP 7 installation on your computer, can be selected from the Start menu (Start > SIMATIC > Documentation).

## Siemens literature


**/1/** SIMATIC NET
S7 CPs for Industrial Ethernet
Manual
Siemens AG
(manual for each CP in the SIMATIC NET Manual Collection, download addresses in the preface of /2/)

**/2/** SIMATIC NET
S7 CPs for Industrial Ethernet
Configuring and Commissioning
Configuration Manual
Teil A − General Application
Siemens AG
part of the online documentation in STEP 7 (▶ “NCM S7 manuals”)
(ID: 8777865)

**/3/** SIMATIC NET
Manual CP 243−1 IT − Communications Processor for Industrial Ethernet and Information Technology
Siemens AG
(ID: 18975343)

### Further recommended reading on the topics Internet/Web, HTML, Java

In the meantime, there are numerous books available with which you can learn Java; we recommend:

***/4/*** Java 2 in 21 Days
by Laura Lemay and Roger Cadenhead
SAMS (2003)

***/5/*** Java in a Nutshell
by David Flanagan
4th Edutuib for Java 1.4 (2002)
5th Edition for Java 5.0 (2005)

***/6/*** Handbuch der Java−Programmierung
by Guido Krügers
Addison−Wesley, Munich, 5th Edition (2007)