

Visualization with User-defined Web Pages

S7-Web2PLC

Application Description • November 2010

Applications & Tools

Answers for industry.

SIEMENS

Industry Automation and Drives Technologies Service & Support Portal

This entry originates from the Internet Service Portal of Siemens AG, Industry Automation and Drives Technologies. The following link takes you directly to the download page of this document.

<http://support.automation.siemens.com/WW/view/en/44212999>

If you have any questions regarding this document, please send us an e-mail to the following address:

online-support.automation@siemens.com

SIMATIC

Visualization with User-defined Web Pages

S7-Web2PLC

Automation Task

1

Automation Solution

2

**General Principles of Web
Pages and User-defined
Web Pages**

3

**Functional Mechanisms
of this Application**

4

**Configuration and
Settings**

5

Installation

6

Application Operation

7

Glossary

8

References

9

History

10

Warranty and Liability

Note

The application examples are not binding and do not claim to be complete regarding configuration, equipment and any eventuality. The application examples do not represent customer-specific solutions. They are only intended to provide support for typical applications. You are responsible for ensuring that the described products are used correctly. These application examples do not relieve you of the responsibility to use sound practices in application, installation, operation and maintenance. When using these application examples, you recognize that we cannot be made liable for any damage/claims beyond the liability clause described. We reserve the right to make changes to these application examples at any time without prior notice. If there are any deviations between the recommendations provided in this application example and other Siemens publications – e.g. Catalogs – the contents of the other documents have priority.

We do not accept any liability for information contained in this document.

Any claims against us – based on whatever legal reason – resulting from the use of the examples, information, programs, engineering and performance data, etc. described in this application example shall be excluded. Such an exclusion shall not apply in the case of mandatory liability, e.g. under the German Product Liability Act (“Produkthaftungsgesetz”), in case of intent, gross negligence, or injury of life, body or health, guarantee for the quality of a product, fraudulent concealment of a deficiency, or breach of a condition which goes to the root of the contract (“wesentliche Vertragspflichten”). However, claims arising from a breach of a condition which goes to the root of the contract shall be limited to the foreseeable damage which is intrinsic to the contract, unless caused by intent or gross negligence or based on mandatory liability for injury of life, body or health. The above provisions do not imply a change in the burden of proof to your detriment.

It is not permissible to transfer or copy these application examples or excerpts thereof without express authorization by Siemens Industry Sector.

Table of Contents

1	Automation Task	7
1.1	Overview	7
2	Automation Solution	9
2.1	Overview of the overall solution	9
2.2	Description of the content of this application	11
2.3	Hardware and software components used.....	14
2.4	Alternative solutions for user-defined web pages	16
3	General Principles of Web Pages and User-defined Web Pages.....	17
3.1	General principles of web pages.....	17
3.1.1	Principles of HTML	17
3.1.2	Using forms	18
3.1.3	Principles of JavaScript.....	18
3.1.4	Automatic refreshing of the web page.....	20
3.2	Principles of user-defined web pages	21
3.2.1	Creating user-defined web pages	21
3.2.2	Blocks required for user-defined web pages.....	23
3.3	Displaying variables from the CPU on the web page.....	24
3.3.1	Requirements	25
3.3.2	Procedure	25
3.4	Writing variables on the CPU with the help of the web page	26
3.4.1	Requirements	27
3.4.2	Procedure	27
3.5	Linking variables with texts in the HTML file	29
3.5.1	Requirements	29
3.5.2	Procedure	30
4	Functional Mechanisms of this Application	31
4.1	Functional principle of the S7 program	31
4.1.1	OB1	33
4.1.2	OB100	33
4.1.3	FC1 (AWP_Main)	34
4.1.4	FC2 (Sim_Tank)	35
4.2	Functional principle of the HTML file	39
4.2.1	AWP commands.....	39
4.2.2	Information on doctype and head of the HTML file	40
4.2.3	Displaying of images	42
4.2.4	Creating an invisible table with texts	43
4.2.5	Outputting CPU variables.....	44
4.2.6	Outputting texts via enumerations.....	44
4.2.7	Setting variables in the CPU with value and button	45
4.2.8	Setting variables in the CPU via a button only	45
5	Configuration and Settings	47
5.1	Configuration of the S7-300 station with the CPU 317-2 PN/DP	48
5.2	Creating the symbols in the symbol file.....	50
5.3	Exporting symbols, project settings and enumerations with S7-Web2PLC	51
5.4	Creating the HTML file	53
5.5	Generating project settings, enumerations and DB333 with S7-Web2PLC	53
5.6	Creating the S7 program	56
5.7	Calling the web page with a web browser.....	56
6	Installation	58
6.1	Hardware and software installation	58
6.2	Installation of the application example	59

Table of Contents

7	Application Operation	60
8	Glossary	63
9	References	65
9.1	Literature	65
9.2	Internet links	66
10	History	66

1 Automation Task

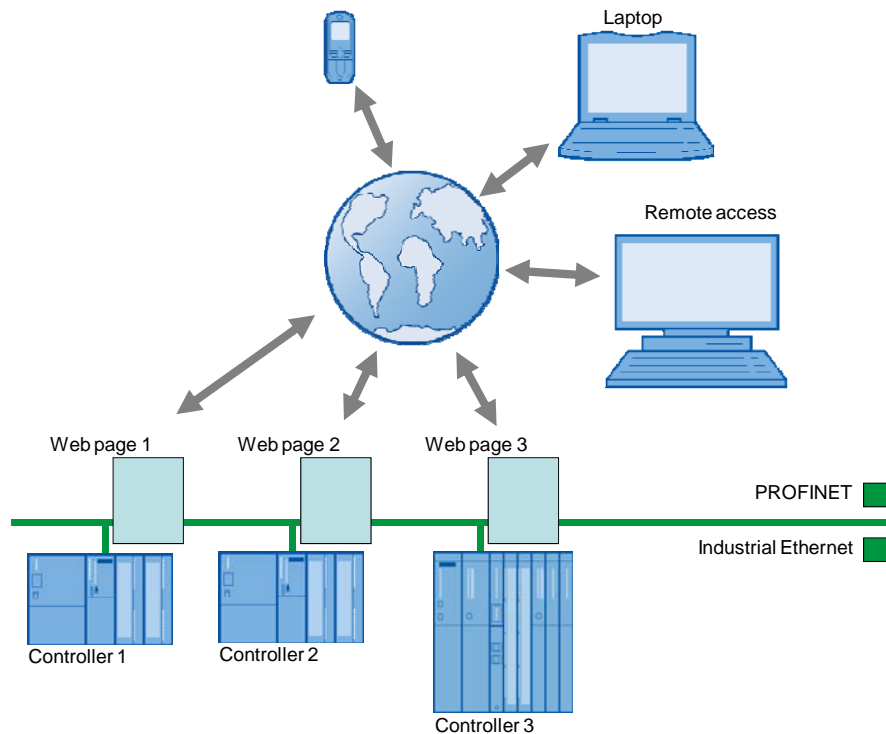
1.1 Overview

Overview of the automation task

Modern automation technology increasingly integrates internet technologies which – together with an integrated Ethernet-based communication – enable, for example, direct access to the system via the intranet. During the test and commissioning phase, the commissioning engineer wants to have flexible access to the CPU; individual data are to be visualized during operation for diagnostic purposes:

For access mechanisms via the internet or intranet it is reasonable to use already existing standards such as the http technology, standard web browsers and common "languages" such as HTML or JavaScript.

Figure 1-1



Description of the automation task

If you want to access a CPU via standard web mechanisms, the following requirements are to be met:

- They gain access to the CPU with standard hardware and standard mechanisms via Industrial Ethernet. They do not require any additional hardware and software.
- Access to the CPU is performed individually related to the system and also visualized, if required. Each CPU has its individual web page virtually "in the stomach".
- Also operating personnel without any automation knowledge is provided simple access to the CPU.

2 Automation Solution

2.1 Overview of the overall solution

Scheme

SIMATIC CPUs with PROFINET interface provide the opportunity to access variables of the CPU with the help of web pages provided by the system.

The CPU web server is accessed via a web browser: In addition to the standard mechanisms of the web page such as identification, diagnostic buffer, module status, messages, communication, topology, variable status and table, individual web pages can be designed and called for your particular application.

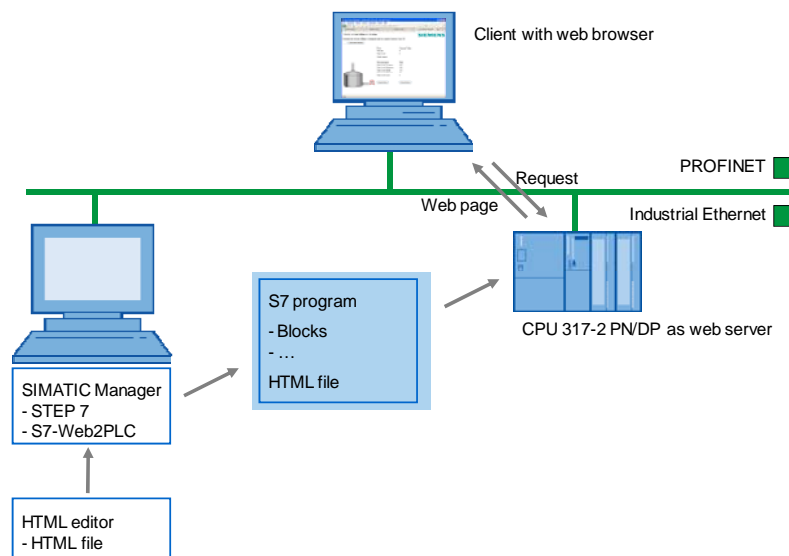
The web server with the web page is already integrated in the CPU.

To create your individual web page (user-defined web page), you can use any tools such as Microsoft Frontpage, Notepad, etc.. For designing your web page, you can use all options provided by HTML, CSS and JavaScript.

In addition, there is a special command syntax (AW P command) for directed communication with the CPU.

The following figure gives an overview of the implemented solution.

Figure 2-1



Procedure for creating user-defined web pages at a glance

1. With an HTML editor, you create the HTML file for the user-defined web page.
2. With STEP 7, you create the S7 program.
3. With the option package S7-Web2PLC which is integrated in STEP 7, the HTML file is stored in data blocks together with images, etc..
4. With the SIMATIC Manager, you transfer all data blocks to the CPU.
5. Via a web browser, you open the web page of the CPU. Access to the web server of the CPU is possible independently of the configuration computer; every output device with access to the PN interface of the CPU can display the web page.

Detailed explanations of the creation of a web page and programming in STEP 7 can be found from Chapter 5 Configuration and Settings on.

Configuration of the application

This application was implemented with a CPU 317-2 PN/DP. A PC is connected via the PROFINET interface. The PC serves for the creation of the S7 program and the HTML file as well as for displaying the web page in a web browser.

Shown are all steps necessary to create a web page and to subsequently call it via the CPU.

Topics not covered by this application

This application is an introduction to the use of S7-Web2PLC and user-defined web pages for beginners. Shown are simple methods for accessing the web page of a CPU with HTML and S7-Web2PLC.

This application does not include a complete description of HTML. To gain deeper knowledge of HTML and JavaScript, please refer to the literature and internet pages specified in Chapter 0 References.

Required knowledge

We assume that you are already familiar with SIMATIC S7 and STEP 7.

2.2 Description of the content of this application

Content of the example application

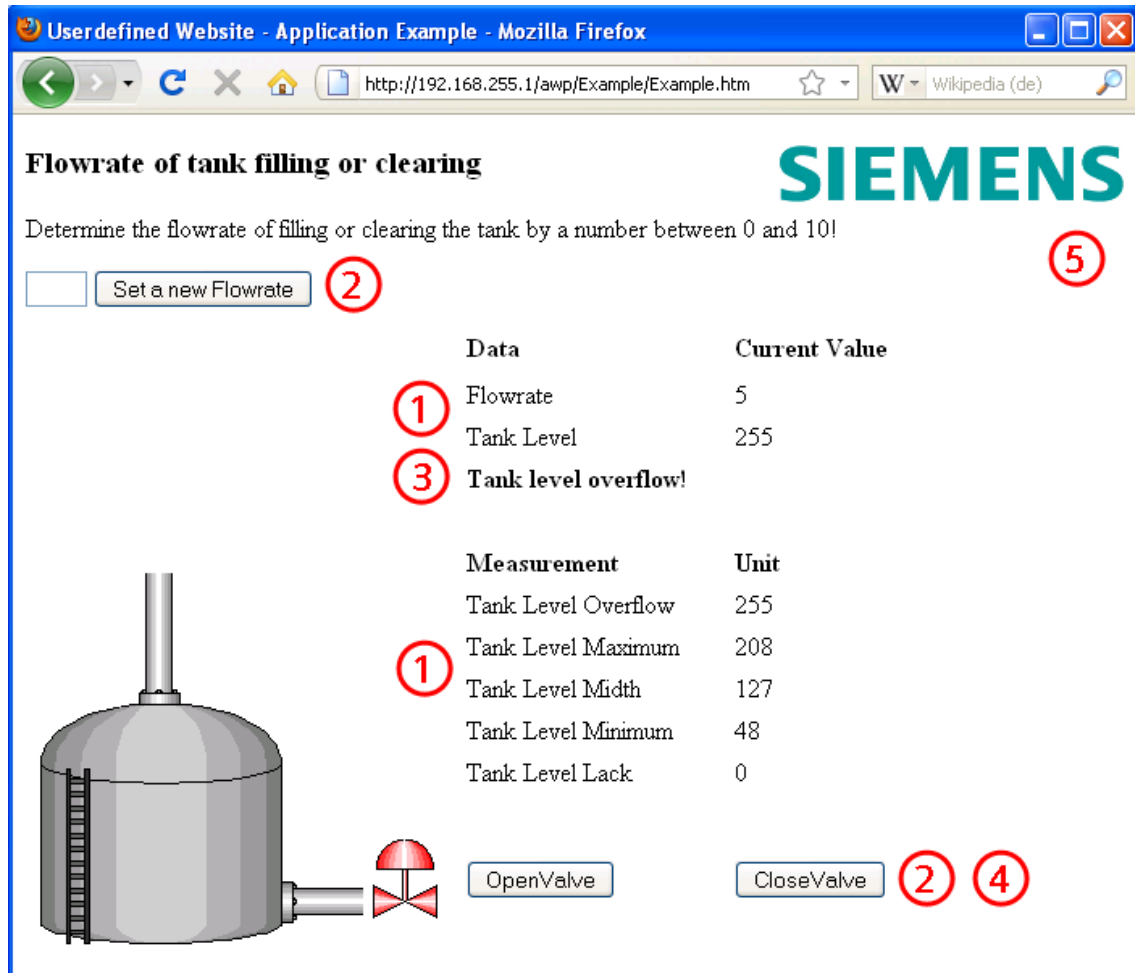
The example application provides the following deepening contents:

- Configuration of the web server for a CPU with PN interface
- Creation of a user-defined web page for the CPU with the following functions:
 - Displaying of CPU variables (1; see Figure 2-2)
 - Setting of CPU variables (2)
 - Displaying of texts which are linked with CPU variables (3)
 - Displaying of images which are linked with CPU variables (4)
 - Cyclic refreshing of the web page (5)
- Particularities in the S7 program creation
 - Providing variables for the web page
 - Further processing of variables from the web page in the S7 program

Overview and description of the user interface

The following figure shows the web page created in this application:

Figure 2-2



Description

The web page shows a tank with the tank filling level "Tank Level". The limit values of the tank filling level can be found under "Measurement".

Via the "OpenValve" button, the tank valve can be opened so that the liquid is cleared. With the "CloseValve" button, the tank valve is closed so that the tank is filled. Dependent on the pressed button, the valve position is indicated via the color.

The flow rate with which the tank is filled or cleared is defined via the "Set a new Flowrate" button. By default, a medium flow rate of 5 is set. The lower the value for "Flowrate", the faster the tank is cleared.

Via the indication, the status of the tank filling level is indicated in clear text.

Advantages of the solution with AWP

The creation of a user-defined web page is advantageous if no permanent HMI system is required, but diagnostic information and visualizations are needed occasionally. Since standard web technologies are used, no additional visualization hardware and software is required.

A solution with AWP is reasonable for **simple** applications and the web page can be designed individually according to your requirements.

2.3 Hardware and software components used

This application was created with the following components:

Hardware components

Note

For this application, you require the current firmware version of the CPU. Depending on the CPU type, the following entries contain links to the corresponding downloads:

- S7-300: <http://support.automation.siemens.com/WW/view/en/26290163>
- S7-400: <http://support.automation.siemens.com/WW/view/en/40945038>

Table 2-1

Component	Qty.	MLFB/order number	Note
CPU 317-2 PN/DP	1	6ES7317-2EK14-0AB0	Alternatively, every S7-300 (firmware version V3.2 or higher) or S7-400 CPU (firmware version V6.0 or higher) with PROFINET interface can be used.
SIMATIC S7-300 power supply PS307	1	6ES7307-1BA00-0AA0	Alternatively, every S7-300 or S7-400 power supply can be used.
PG/PC with Ethernet interface	1		
IE FC TP STANDARD CABLE	1	6XV1840-2AH10	Connecting cable IE; minimum order quantity 20 m
RJ45 plug-in connector	2	6GK1901-1BB10-2AA0	Easy to assemble

Standard software components

Table 2-2

Component	Qty.	MLFB/order number	Note
STEP 7 V5.5 with integrated S7-Web2PLC	1	6ES7810-4CC10-0YA5	S7-Web2PLC can be found on the STEP 7 CD under "Optional Components".
Software tool for creating HTML files, e.g. Frontpage, Notepad, ...	1		
Web browser, e.g. Internet Explorer, Mozilla Firefox, ...	1		

Example files and projects

The following list contains all files and projects used in this example.

Table 2-3

Component	Note
44212999_Web2PLC_CODE_v10.zip	The packed file contains the STEP 7 project with the related HTML file. The HTML file with the related files is stored in the project directory \Step7\S7Proj\Web\Web2PLC\Example\html.
44212999_Web2PLC_DOKU_v10_e.pdf	This document.

2.4 Alternative solutions for user-defined web pages

Depending on your application task, Siemens offers a wide range of solutions for data access to S7-CPU's with web technologies:

Java applets

With the help of IE CPs (CPs 243-1IT, 343-1 Advanced, 443-1 Advanced) and Java applets, you can easily create a simple visualization user interface. Process variables are integrated via S7 controls. The Java user interface is displayed in a standard web browser.

Java Beans

S7 Java Beans provide the opportunity to create individual Java applets for medium to large web applications.

Sm@rt Service with WinCC flexible

With WinCC flexible and the Sm@rtService option, it is possible to connect to a control panel via the internet directly from your service/maintenance workplace.

WinCC Web Navigator

The WinCC option package "WinCC Web Navigator" enables the creation of a solution for the topic "operation and monitoring" via the intranet/internet. Thus, you can implement a distribution of the operating and monitoring functions of your automation system via the intranet/internet with the standard means of WinCC. In that, common internet safety methods are supported by the "WinCC Web Navigator" and the supplied wizards provide quick support for your task.

Further links

Further information on alternative solutions can be found in Chapter 0 References.

3 General Principles of Web Pages and User-defined Web Pages

General definitions

In the context of web design, the term web page is used for a document in the World Wide Web, which can be called from a web server with a web browser by specifying a Uniform Resource Locator (URL). In this context, it is also spoken about an HTML page or an HTML document.

A user-defined web page is understood as a web page with an additional command syntax (AWP commands) which can be used to access an S7-CPU with PN interface.

3.1 General principles of web pages

If you already have basic knowledge of HTML, you may skip this Chapter and continue reading at Chapter 3.2 Principles of user-defined web pages.

3.1.1 Principles of HTML

HTML stands for "Hypertext Markup Language" and is a text-based markup language for structuring headers, texts, lists, tables or images. Among other things, HTML does without loops and variables and is therefore no programming language.

Structure

An HTML document consists of three areas:

- Document type declaration (doctype) at the beginning of the file, stating the document type definition (DTD) used, e.g. HTML 4.01 Transitional.
- HTML head for information which is not to be displayed in the display area of the web browser.
- HTML body for information which is displayed in the web browser.

HTML elements (tags)

The content of HTML files is written in "HTML elements" and is marked by tags. Almost all HTML elements are marked by an introductory and a concluding tag. The content in between is the "scope of application" of the corresponding element.

Example: Text paragraphs are marked by the <p> tag. The end of a tag is represented by an introductory "</".

```
<p>This is a text.</p>
```

Tags are cascadable and can be nested.

Typical tags

The following table gives an overview of the most important tags for structuring information, which are also used in this example application:

Table 3-1

Representation	Function	Example
<code><!-- ... --></code>	Comment	<code><!-- This is a comment! --></code>
<code><a> ... </code>	Link	
<code> ... </code>	Boldface	<code>This text is bold.</code>
<code><body> ...</body></code>	Content is displayed in the web browser	
<code><h1> ... </h1></code>	Text heading	
<code><head> ... </head></code>	Head of an HTML file	
<code><html> ... </html></code>	Fundamental web page tag	
<code></code>	Integration of an image	
<code><p> ... </p></code>	Text paragraph	
<code><style> ... </style></code>	Definition domain for stylesheet formattings	
<code><table> ... </table></code>	Table Creates a table in combination with <code><tr></code> and <code><td></code>	
<code><td> ... </td></code>	Table column	
<code><th> ... </th></code>	Table head	
<code><tr> ... </tr></code>	Table row	

3.1.2 Using forms

Forms are used for being able to perform interactions with the user.

For example, the user can fill in input fields in a form and then send the form by clicking a button. The content of the form is thus sent to the web server.

With the "POST" method, the content of the form is transferred from the web browser to the web server with a special POST request.

3.1.3 Principles of JavaScript

JavaScript is an own programming language and was created for the purpose of optimizing HTML pages. JavaScript is a client-side programming language, which means that the JavaScript programs are executed in the web browser and interpreted by the web browser during runtime.

JavaScript is supposed to supplement HTML, not to replace it.

JavaScript is fundamentally different from the programming language Java. The similarity of the name is due to the intention to make a connection to the then very popular programming language for marketing reasons.

With JavaScript, you can, among other things, expand the HTML page by the following functions:

- Processing of keyboard entries

- Dynamic modification of the web page

Integration of JavaScript in HTML

There are several ways to integrate JavaScript commands in an HTML file:

- between the tags `<script>` and `</script>`
- in an external file
- as HTML link
- as parameter of an HTML tag

Usually, JavaScript commands are defined in the head of the HTML file, i.e. between the tags `<head>` and `</head>`. The syntax for JavaScript is:

```
<script type="text/javascript"> </script>
```

3.1 General principles of web pages

3.1.4 Automatic refreshing of the web page

Options

In principle, HTML is static and does not respond to modifications of the content. Therefore, if values change in the S7 program, it is useful to have the changed values displayed in the web browser.

There are several ways to refresh the display of the web page:

- Manual refresh with "F5"
- Automatic refresh with an HTML meta date in the head of the HTML file
- Automatic refresh with JavaScript in the body of the HTML file

Manual refresh

With the function key "F5" (Internet Explorer: "View > Refresh"; Mozilla Firefox: "View > Reload"), the display in the web browser is refreshed manually.

Refreshing with HTML

With the following code line in the head of the HTML file, the display in the web browser is refreshed cyclically:

```
<meta http-equiv="refresh" content="2; URL=Example.htm">
```

The refresh cycle is entered in seconds. With "content="2", the refresh cycle is 2 seconds.

Via "URL=", the web page to be refreshed is specified, in this example the file "Example.htm".

Refreshing with JavaScript

In the body of the HTML file, the following JavaScript refreshes the display in the web browser every 5 seconds:

```
<script language="JavaScript">  
window.setTimeout("location.reload()",5000);  
</script>
```

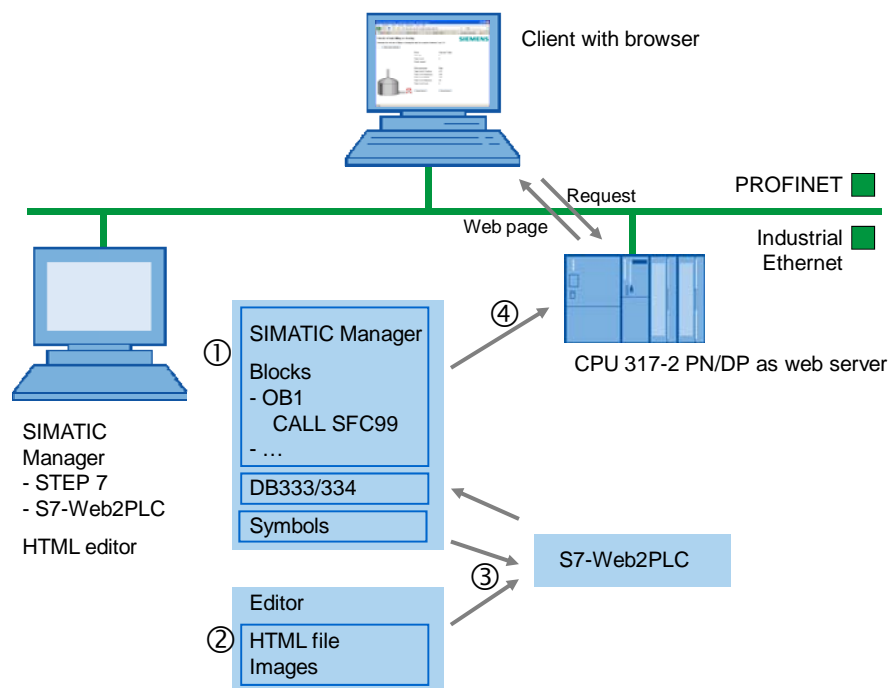
3.2 Principles of user-defined web pages

The following Sections provide basic knowledge of user-defined web pages referred to the application.

Context-related information can be found in the online help of S7-Web2PLC and on SFC99.

3.2.1 Creating user-defined web pages

Figure 3-1



Procedure

1. With an HTML editor, you create the HTML file for the CPU. To be able to access variables of the CPU, a corresponding command syntax (AWP commands) is provided.
2. In STEP 7, you assign a symbolic name to variables which you want to use on the web page.
3. With S7-Web2PLC, you import the symbols and integrate the HTML file. S7-Web2PLC creates the HTML file incl. all related files (e.g. images) in DB333 and – depending on the data volume – in DB334 to DB349 (DB numbers are freely configurable).
4. With STEP 7, you create an S7 program. To exchange data between the web browser and CPU as web server, you call SFC99.
5. With the SIMATIC Manager, you transfer all data blocks to the CPU.
6. Via a web browser, you open the web page of the CPU:
The web browser requests the web page of the CPU via the http protocol; the CPU as web server provides the web page.

Access to the web server of the CPU is possible independently of the configuration computer; every output device with an integrated web browser and access to the PN interface of the CPU can display the web page.

Access to the web page is password-protected.

3.2.2 Blocks required for user-defined web pages

SFC99

With the help of SFC99, the CPU interprets the data blocks and can use these as user-defined web pages.

DB333, DB334 and others

The basis of the web page designed by you is an HTML file (or several connected HTML files with images):

To enable the CPU to interpret the HTML file, it is stored in data blocks together with further required files. S7-Web2PLC is used for that:

DB333 (web control DB) contains the following:

- Status and control variables of the web page
- Communication status (e.g. whether a request from the web browser to the web server is pending)
- Error information

In addition to DB333, there are up to 16 other "fragment DBs" starting with DB334. These DBs contain the coded web pages and media data (e.g. images).

All DBs (DB333 and fragment DBs) together must only contain data of a maximum of 1 Mbyte.

Typical AWP commands

The following table provides an overview of the most important AWP commands

Table 3-2

Representation	Function	Example	Information
<code>:= "<Name> "</code>	Display CPU variable	<code>:= "TankLevelMinimum" :</code>	Section 3.3
<code><!-- AWP_In_Variable Name = ' "<Name> " ' --></code>	Configuration to be able to write a variable on the CPU with a separate "POST" method	<code><!-- AWP_In_Variable Name= ' "OpenValve" ' --></code>	Section 3.4
<code><!-- AWP_Enum_Ref Name= ' "<Name> " ' Enum= "<Variable> " --></code>	Assignment of enumerations (texts) to the value of a variable	<code><!-- AWP_Enum_Ref Name= "Alarm" ' Enum= "AlarmValue" --></code>	Section 0

3.3 Displaying variables from the CPU on the web page

The following graphic illustrates the interaction between web browser and CPU:

Figure 3-2

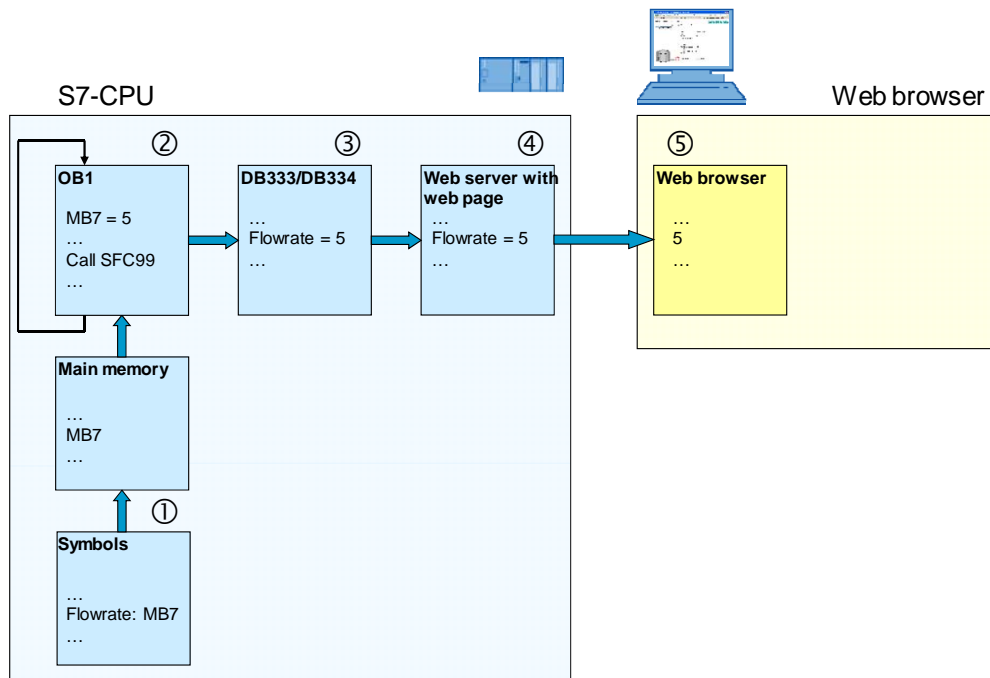


Table 3-3

No.	Function
1	Variables which are displayed on the web page must have a symbolic name.
2	In the S7 program, SFC99 is called.
3	By calling SFC99, the web control DB (DB333) is initialized.
4	The web server of the CPU converts the data with the help of the information in the web control DB (DB333) to a format (= web page) which can be interpreted by a web browser. The web page of the CPU is called in a web browser via the IP address of the CPU.
5	With each request from the web browser, the web page is refreshed (manually or automatically). Information on the refreshing of a web page can be found in Section 0 Automatic refreshing of the web page. A request to the web server is also created with the "POST" method in the HTML file. After having "sent" the web page, the entire web page is refreshed.

3.3.1 Requirements

To be able to display variables of the CPU on the web page, the following preconditions apply:

Table 3-4

S7 program	HTML file
<ul style="list-style-type: none"> Each variable must be assigned a symbolic name. The variable can only be displayed on the web page via symbolic names. SFC99 must be called (if variables are pre-processed in the S7 program, cyclic call) For variables, all data types are approved. 	<ul style="list-style-type: none"> It is not necessary to declare variables via an AWP command in the HTML file.

3.3.2 Procedure

S7 program:

No particularities

HTML file:

A variable can be displayed at any position on the HTML page. The syntax is as follows:

```
:= "<Variable>" :
```

Example of the variable "TankLevelMaximum":

```
<p>:= "TankLevelMaximum" : </p>
```

3.4 Writing variables on the CPU with the help of the web page

The following graphic illustrates the interaction between web browser and CPU:

Figure 3-3

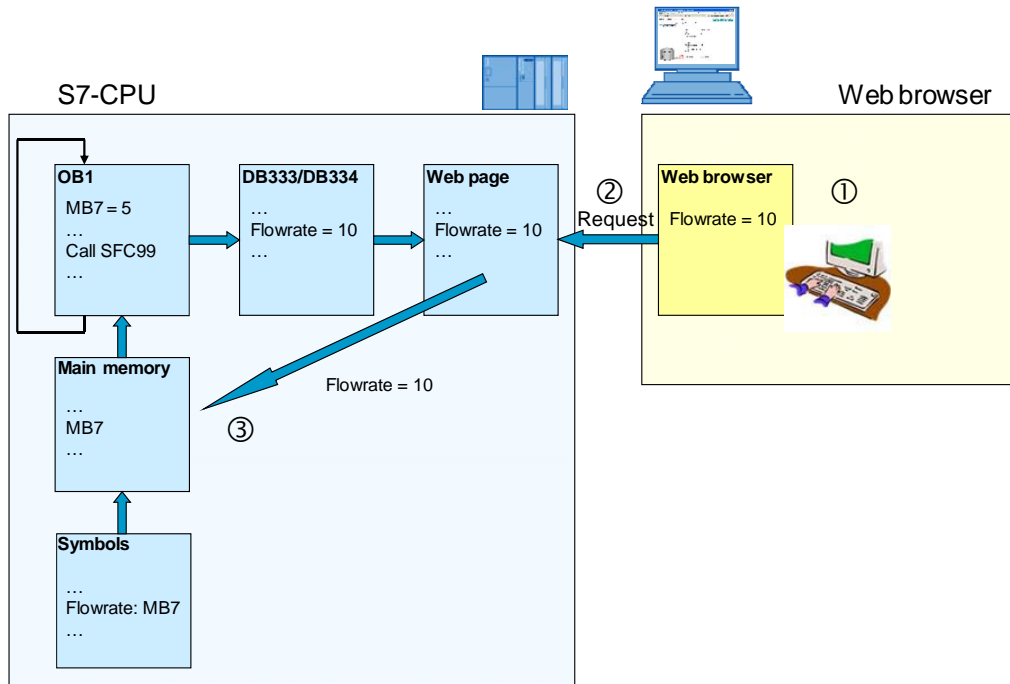


Table 3-5

No.	Function
1	Via the web page, the user changes the "Flowrate" variable to the value "10".
2	The web browser reports a request ("POST" method).
3	The S7 program accepts the changed "Flowrate" variable, the display in the web browser is refreshed, the new values are displayed.

3.4.1 Requirements

To be able to write variables on the CPU via the web page, the following preconditions apply:

Table 3-6

S7 program	HTML file
<ul style="list-style-type: none"> Each variable must be assigned a symbolic name. A variable can only be addressed via symbolic names. SFC99 must be called (if variables are pre-processed in the S7 program, cyclic call) For variables, all data types are approved. 	<ul style="list-style-type: none"> Variables must be declared via the AWP command (<code><!-- AWP_In_Variable ... --></code> in the HTML file). The variables must be transferred to the CPU (e.g. POST method in the HTML file).

3.4.2 Procedure

AWP command for variables

The AWP command via which variables can be written in the CPU is as follows:

```
<!-- AWP_In_Variable Name="Variable" -->
```

Example of how to write the "Flowrate" variable:

```
<!-- AWP_In_Variable Name="Flowrate" -->
```

The AWP command typically stands at the beginning of the HTML file.

Transferring the variables from the web browser

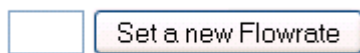
When calling the form, the POST method is selected for transferring the data from the web browser to the web server. The form consists of two units:

- A field for entering the value:
The field is named via a variable and designates the variable from the AWP command, e.g.
`<!-- AWP_In_Variable Name='Flowrate' -->`.
- A button with which the entry of the value is confirmed.

Via "submit", the form data are transferred.

Example:

Appearance on the web page:



The image shows a simple web form. On the left is a rectangular text input field. To its right is a rectangular button with a blue border and the text "Set a new Flowrate" inside.

Code:

```
<form method="post" action="">  
  <input type="text" name="Flowrate" size="6">  
  <input type="submit" value="Set as new value">  
</form>
```

3.5 Linking variables with texts in the HTML file

Occasionally, it makes sense on a web page to output indications directly as a text and not as a variable. Web2PLC provides enumerations for that. With an enumeration, you can link values with concrete texts. These texts can be created in one language or in several languages. Our application contains single-language text messages.

The following graphic illustrates the interaction between web browser and CPU:

Figure 3-4

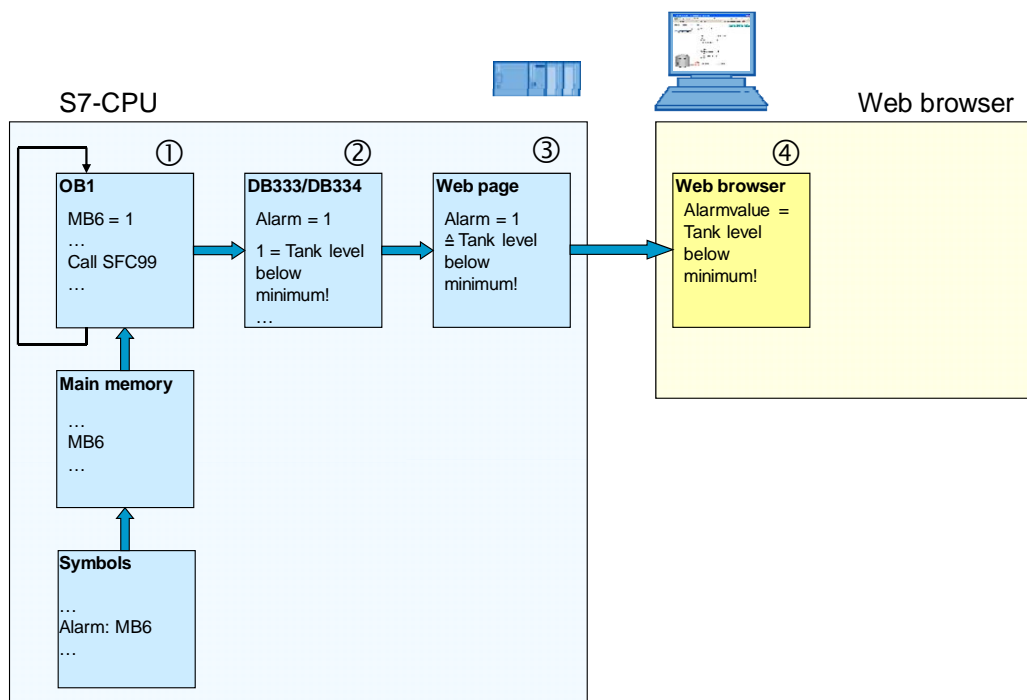


Table 3-7

No.	Function
1	The S7 program calls SFC99 and sets the value of MB6 ("Alarm") to "1".
2	Due to the cyclic calling of SFC99, the "Alarm" variable in DB333/334 is also refreshed.
3	The web server links the "Alarm" value with the related text.
4	In the web browser, the related text is output instead of the "Alarm" value.

3.5.1 Requirements

To output indications as text, the following preconditions apply:

Table 3-8

S7 program	S7-Web2PLC	HTML file
<ul style="list-style-type: none">• Each variable must be assigned a symbolic name. A variable can only be addressed via symbolic names.• SFC99 must be called (if variables are pre-processed in the S7 program, cyclic call)• For variables, all numerical data types are approved.	<ul style="list-style-type: none">• All language-dependent files incl. the HTML file must be stored in the same directory.• In S7-Web2PLC, the possible values of the variable must be linked with texts.	<ul style="list-style-type: none">• It is not necessary to declare variables via an AWP command in the HTML file, because they are only read but not written.

3.5.2 Procedure

S7-Web2PLC

1. Start S7-Web2PLC, load your project and select "File > Change Project Settings".
2. Branch to the "ENUM types" tab and enter the name of the variable via the "New" button.
3. Assign the texts to the variable values via the "Insert enum value" button.

HTML file:

The syntax for the displaying of texts instead of the value is as follows, e.g. for the "Alarm" variable:

```
<!-- AWP_Enum_Ref Name=' "Alarm" ' Enum="AlarmValue" --  
>:="Alarm":
```

The texts for enumerations are created with the help of S7-Web2PLC.

A detailed explanation will be given later in Section 5.3 Exporting symbols, project settings and enumerations with S7-Web2PLC

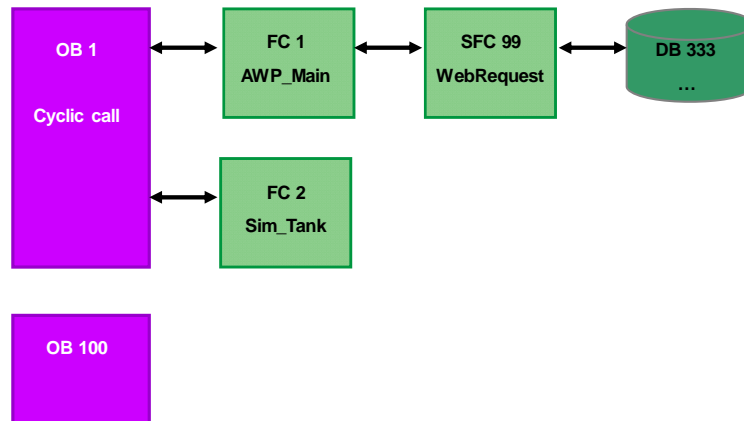
4 Functional Mechanisms of this Application

4.1 Functional principle of the S7 program

The S7 program of this application only serves for representing individual functional principles of S7-Web2PLC by way of example.

The call structure in the S7 program looks as follows:

Figure 4-1

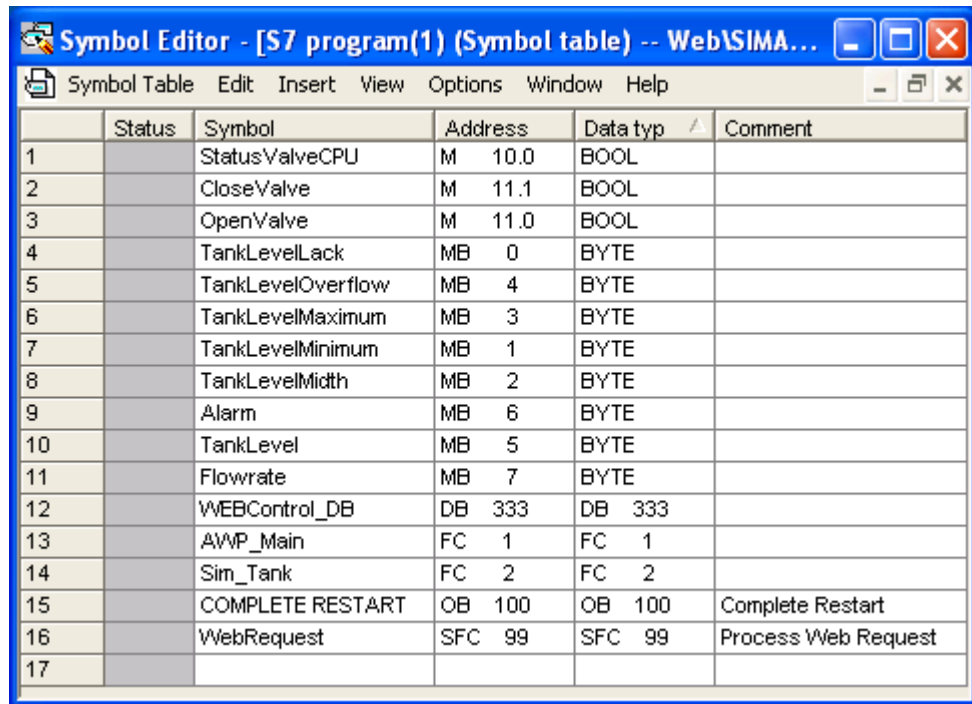


4 Functional Mechanisms of this Application

4.1 Functional principle of the S7 program

The following symbols are used:

Figure 4-2



	Status	Symbol	Address	Data typ	Comment
1		StatusValveCPU	M 10.0	BOOL	
2		CloseValve	M 11.1	BOOL	
3		OpenValve	M 11.0	BOOL	
4		TankLevelLack	MB 0	BYTE	
5		TankLevelOverflow	MB 4	BYTE	
6		TankLevelMaximum	MB 3	BYTE	
7		TankLevelMinimum	MB 1	BYTE	
8		TankLevelMidth	MB 2	BYTE	
9		Alarm	MB 6	BYTE	
10		TankLevel	MB 5	BYTE	
11		Flowrate	MB 7	BYTE	
12		WEBControl_DB	DB 333	DB 333	
13		AWP_Main	FC 1	FC 1	
14		Sim_Tank	FC 2	FC 2	
15		COMPLETE RESTART	OB 100	OB 100	Complete Restart
16		WebRequest	SFC 99	SFC 99	Process Web Request
17					

4.1.1 OB1

In OB1, FC1 and FC2 are called cyclically.

Figure 4-3

```
OB1 : "Main Program Sweep (Cycle)"
Netzwerk 1: Titel:
          CALL "AWP_Main"           FC1
          CALL "Sim_Tank"           FC2
```

4.1.2 OB100

In the startup OB OB100, a start value for the "Flowrate" is stored.

Figure 4-4

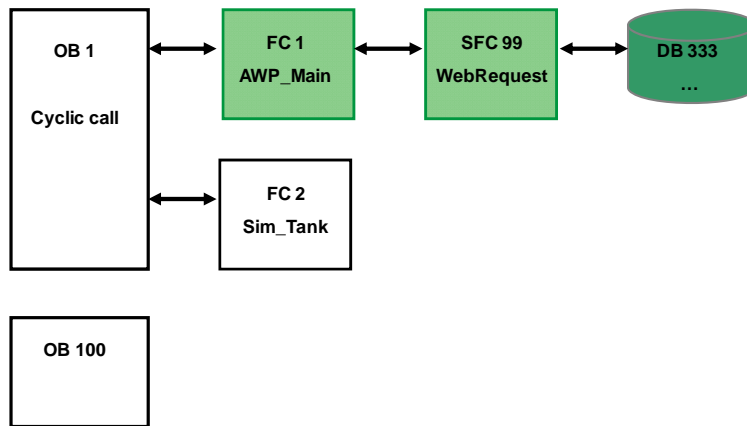
```
OB100 : "Complete Restart"
Netzwerk 1: Register AWP Application
          L    W#16#5
          T    "Flowrate"           MB7
```

4.1.3 FC1 (AWP_Main)

In FC1, the status of DB333 is polled cyclically to be able to recognize a request from the web browser. The cause for a request is that a variable changed by the user is to be transferred from the web browser to the web server.

Note: When the user refreshes the display of the web page in the web browser, this does not have any effect in the S7 program.

Figure 4-5



By polling the status "8010" it is queried whether the initialization of the web control DB has possibly failed.

Figure 4-6

```

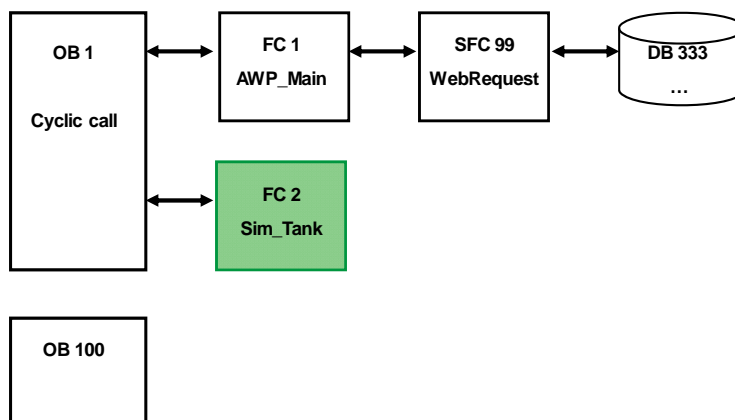
FC1 : AWP_Main
Netzwerk 1: Synchronize Web Control DB
CALL "WebRequest"           SFC99           -- Process Web Request
   CTRL_DB:="WEBControl_DB" DB333
   RET_VAL:=#SFCState_333  #SFCState_333

L   #SFCState_333          #SFCState_333
UW  W#16#8010
BE
    
```

4.1.4 FC2 (Sim_Tank)

Functional principle of FC2

Figure 4-7



In FC2, the filling or clearing of a tank is simulated, dependent on the flow rate and the valve position.

The user can define the flow rate via the "Flowrate" variable on the web page. The flow rate is multiplied with a basic factor of 20 ms and transferred to a timer. After the expiry of the timer, the tank filling level is increased or decreased by 1. The current filling level is stored in "TankLevel".

Via the two variables "OpenValve" and "CloseValve", the valve position is read in and stored in the CPU in the "StatusValveCPU" variable.

Dependent on the tank filling level, the following heights are displayed:

- Tank has been cleared (TankLevelLack)
- Tank filling level is at minimum (TankLevelMinimum)
- Tank filling level is 50 % (TankLevelMidth)
- Tank filling level is at maximum (TankLevelMaximum)
- Tank is flowing over (TankLevelOverflow)

Via the "Alarm" variable, the tank filling level is output in clear text (also as enumeration)

Defining the limit values of the variables

First of all, the limit values of the variables are defined. Later, these are also displayed on the web page:

Figure 4-8

```
FC2 : Increment and Decrement the Tank Level
Netzwerk 1: Titel:
L      W#16#0
T      "TankLevelLack"           MB0
L      W#16#30
T      "TankLevelMinimum"       MB1
L      W#16#7F
T      "TankLevelMidth"         MB2
L      W#16#D0
T      "TankLevelMaximum"       MB3
L      W#16#FF
T      "TankLevelOverflow"      MB4
```

Time loop with "Flowrate"

To ensure that the filling or clearing of the tank is not performed too fast, a timer is loaded and multiplied with "Flowrate".

Figure 4-9

```

Netzwerk 2: Titel:
L   "Flowrate"                MB7
L   W#16#2
*I
ITB
T   MW   21

UN  T   1
L   MW   21

SV  T   1
NOT
BEB
L   MB   100
INC  1
T   MB   100

U   M   100.1
=   M   0.1
=   M   0.2

UN  M   100.1
=   M   0.3

```

Polling of the "OpenValve" or "CloseValve" buttons

Additionally, the status of the "OpenValve" and "CloseValve" buttons is polled by the web page. If one of the buttons has been clicked, the status is stored in "StatusValveCPU".

Figure 4-10

```

Netzwerk 3: Check Valve website
CLR
U   "OpenValve"                M11.0
S   "StatusValveCPU"          M10.0
U   "CloseValve"              M11.1
R   "StatusValveCPU"          M10.0

```

4.1 Functional principle of the S7 program

Valve status

Via the "StatusVentilCPU" bit, the button pressed last (OpenValve or CloseValve) is memorized.

Dependent on this bit, the tank is either cleared or filled.

Figure 4-11

```
Netzwerk 4 : Titel:
      CLR
      U   "StatusValveCPU"      M10.0
      SPB  Clr
      SPA  Fill
```

Filling the tank

The filling of the tank starts with a query whether the tank is already full.

If the tank is not full, the tank filling level is increased by one and then compared with the specifications for the limit values of the tank filling level.

Depending on the filling level reached, the values "0" to "5" are stored in the "Alarm" variable. With the value of the "Alarm" variable, texts (enumerations) are stored in S7-Web2PLC, which display the filling level of the tank in clear text.

Clearing the tank

The clearing of the tank is analogous to the filling of the tank. The tank filling level is decreased by "1".

4.2 Functional principle of the HTML file

The following Section provides a detailed explanation of the individual sections of the HTML file.

4.2.1 AWP commands

Basic information

AWP commands can be located at any position in the HTML file. However, for reasons of clarity it is appropriate to state the central AWP commands at the beginning of the HTML file.

In HTML editors which can interpret tags, AWP commands are typically displayed in gray. The HTML editor always interprets the tag "`<!-- ... -->`" as a comment.

Figure 4-12

```
1 <!-- AWP_In_Variable Name='OpenValve' -->
2 <!-- AWP_In_Variable Name='CloseValve' -->
3 <!-- AWP_In_Variable Name='Flowrate' -->
```

Explanations

Table 4-1

Code	Explanation
<pre><!-- AWP_In_Variable Name='OpenValve' --></pre>	All variables transferred to the CPU must be identified as In_Variable.
<pre><!-- AWP_In_Variable Name='CloseValve' --></pre>	Note: Keep in mind that the quotation marks are nested. The variable is written between quotation marks and framed by an inverted comma (' ... ').
<pre><!-- AWP_In_Variable Name='Flowrate' --></pre>	

4.2.2 Information on doctype and head of the HTML file

Basic information

The following information must be contained in every HTML file so that it is HTML-compliant. The only exception is the tag "`<meta http-equiv="refresh" ...>`": If you do without an automatic refresh of the page and work with "F5" instead, you can omit this tag.

Figure 4-13

```

6 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
7 <html>
8
9 <head>
10 <title>Userdefined Website - Application Example</title>
11 <meta http-equiv="Content-Language" content="en" >
12 <meta http-equiv="Content-Type" content="text/html; charset=utf-8" >
13 <meta http-equiv="refresh" content="10; URL=Example.htm">
14 <link rel="stylesheet" type="text/css" href="demo.css">
15 </head>
16
17 <body>

```



```

116 </body>
117 </html>

```

Explanations

Table 4-2

Code	Explanation
<code><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"></code>	Definition of the HTML document type: It is the document type HTML in the language version V4.01 in the "Transitional" variant. The language code "EN" refers to the language of the tags, i.e. English. The document type always stands before the tag " <code><html></code> ".
<code><html> ... </html></code>	Contains the HTML content
<code><title>Userdefined Website - Application Example</title></code>	Title of the web page which will later be displayed in the head of the web browser.
<code><meta http-equiv="Content-Language" content="en" ></code>	Language of the file content
<code><meta http-equiv="Content-Type" content="text/html; charset=utf-8" ></code>	With "content="text/html", the MIME type is specified, followed by the used character set UTF-8.
<code><meta http-equiv="refresh" content="10; URL=Example.htm"></code>	Optional meta date: With this command, the web page is refreshed every 10 seconds. Especially for process monitoring it is appropriate to have the web page refreshed cyclically. Further information on the refreshing of the web page can be found in Section 0 Automatic refreshing of the web page.

Code	Explanation
<code><link rel="stylesheet" type="text/css" href="demo.css"></code>	Via <link...>, a CSS file is referenced which contains all information on the optical design of the web page, e.g. white background color, etc..
<code><body> ...</body></code>	Contains the text body

4.2.3 Displaying of images

Basic information

There are several images used in the HTML file:

- Static images
- Dynamic image which is changed dependent on a status bit in the CPU.

Explanations

Figure 4-14

```

18 
19
101 
    
```

Table 4-3

Code	Explanation
<pre> </pre>	<p>Images are integrated via the "img" tag. The width is specified with "width", the height with "height". These specifications are optional. The position of the image is defined with "align", "right" means right-aligned.</p>
<pre> </pre>	<p>This image depends on the "OpenValve" variable. This variable can take the states "0" and "1".</p> <p>The stored images have the designation Valve0.png (valve closed) and Valve1.png (valve open).</p> <p>When the valve is closed, "OpenValve" has the value 0: the image call consists of: "Valve" + "0" + ".png" = Valve0.png</p> <p>With "old", you specify a text which will be displayed if the image cannot be called by the web page.</p>

4.2.4 Creating an invisible table with texts

Basic information

To ensure that no contents are shifted on the web page dependent on the window size, the use of an invisible table is recommendable.

Of course, you can also define a table centrally for your web page via CSS (Cascading Style Sheet).

Explanations

In the following figure, only the first two lines of the table are shown for reasons of clarity.

Figure 4-15

```

30 <table border="0" width="80%" cellpadding="2" id="table2">
31   <tr>
32     <td width="26%" rowspan="14" valign="bottom">
33       <p align="center">
34         
35       </td>
36     </tr>
37   <tr>
38     <td width="9%" height="30">&nbsp;</td>
39     <td width="135" height="30"><b>Data</b></td>
40     <td height="30" width="31%"><b>Current Value</b></td>
41   </tr>
114 </table>

```

Table 4-4

Code	Explanation
<pre> <table border="0" width="80%" cellpadding="2" id="Tabelle"> ... </table> </pre>	<p>Table with a width of 80 % of the window size and an internal distance (cellpadding) in the cell of 2 pixels.</p> <p>The stroke width (border) of the table is "0".</p>
<pre> <tr> <td width="31" height="30">&nbsp;</td> <td width="286" height="30">Data </td> <td height="30">Current Value </td> </tr> </pre>	<p><tr> stands for table row.</p> <p>The content of a cell stands between <td> (table data) and </td>.</p>
<pre> &nbsp;</td> </pre>	<p>Non-breaking space; occasionally, web browsers have problems presenting table cells without content.</p>
<pre> Data </pre>	<p>Content of the header, text is "data" and displayed in bold.</p>

4.2.5 Outputting CPU variables

Explanations

Variables of the CPU are always displayed via the symbol name:

Figure 4-16

```
46 <td width="31%">:="Flowrate":</td>
```

Instead of "Flowrate", always the current value from the CPU is output on the web page.

4.2.6 Outputting texts via enumerations

Explanations

Via enumerations, texts can be allocated to the individual values of a CPU variable.

Figure 4-17

```
56 <b><!-- AWP_Enum_Ref Name='Alarm' Enum="AlarmValue" -->:="Alarm":</b>
```

Instead of the individual values of "Alarm", the texts allocated in S7-Web2PLC before are output. These texts were stored as enum type "AlarmValue". These texts are transferred to the web page via DB333.

Note: Since the enumeration is located in a table, the tag "<td> ... </td>" is displayed additionally here. The result is output in bold, which is indicated by the tag " ... ".

4.2.7 Setting variables in the CPU with value and button

Basic information

To be able to transfer variables to the CPU via the web page, you have to work with forms and, for example, the "POST" method.

Explanations

Figure 4-18

```

23 <form method="post" action="">
24   <input type="text" name="Flowrate" size="6">
25   <input type="submit" value="Set a new Flowrate">
26
27 </form>

```

Table 4-5

Code	Explanation
<pre> <form method="post" action=""> <input type="text" name="Flowrate" size="6"> <input type="submit" value="Set a new Flowrate"> </form> </pre>	<p>Calling the form with the "POST" method. Under "action", no details are required because with "action" the current page is called by default.</p> <p>With input type="text", an input field is linked the content of which is sent to the web server of the CPU with "submit". "Submit" is controlled via a button called "Set a new Flowrate".</p>

4.2.8 Setting variables in the CPU via a button only

Basic information

To assign variables in the CPU a predefined value, you have to work with a form, the "POST" method and a hidden value.

Explanations

Note: Light text belongs to the superordinate table and is irrelevant in the context of this Section.

Figure 4-19

```

101 <form method="post" action="">
102   <input type="submit" value="OpenValve">
103   <input type="hidden" name="OpenValve" size="20" value="1">
104   <input type="hidden" name="CloseValve" size="20" value="0">
105 </form></td>
106
107 <td height="70" width="31%">
108   <form method="post" action="">
109     <input type="submit" value="CloseValve">
110     <input type="hidden" name="CloseValve" size="20" value="1">
111     <input type="hidden" name="OpenValve" size="20" value="0">
112   </form></td>

```

Table 4-6

Code	Explanation
<pre> <form method="post" action=""> <input type="submit" value="OpenValve"> <input type="hidden" name="OpenValve" size="20" value="1"> <input type="hidden" name="CloseValve" size="20" value="0"> </form> </pre>	<p>Calling the form with the "POST" method. Under "action", no details are required because with "action" the current page is called by default.</p> <p>With input type="hidden", the variable "OpenValve" is assigned the value 1, the variable "CloseValve" the value 0.</p> <p>With "submit", the values of the variables are sent to the web server of the CPU.</p>
<pre> <form method="post" action=""> <input type="submit" value="CloseValve"> <input type="hidden" name="CloseValve" size="20" value="1"> <input type="hidden" name="OpenValve" size="20" value="0"> </form> </pre>	<p>Reverse action to the row above: Calling the form to assign the value 1 to "CloseValve" and the value 0 to "OpenValve".</p>

5 Configuration and Settings

In this Section

This Section contains all information on how to create and operate a web page for a CPU with PN interface. The CPU 317-2 PN/DP is used as an example in this Section. All steps are presented by means of the completed example application.

If you just want to take the completed example application into operation, please continue reading in Chapter 6 Installation

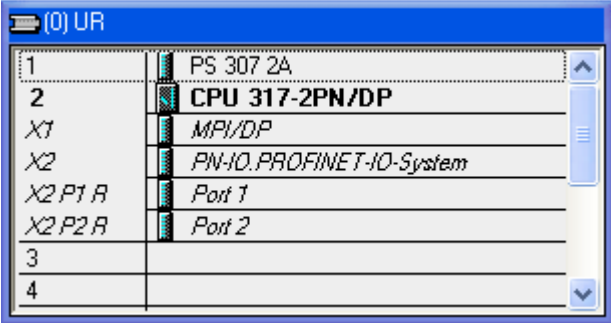
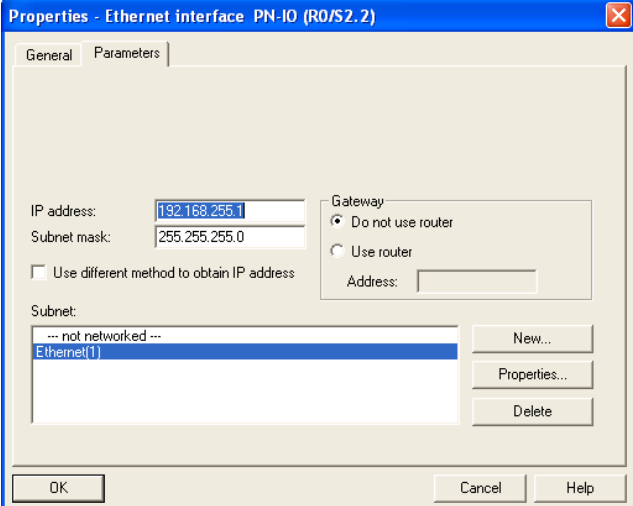
Procedure for creating a web page

The configuration and settings in STEP 7 and the writing of the HTML file are closely linked. The following procedure is recommendable for that:

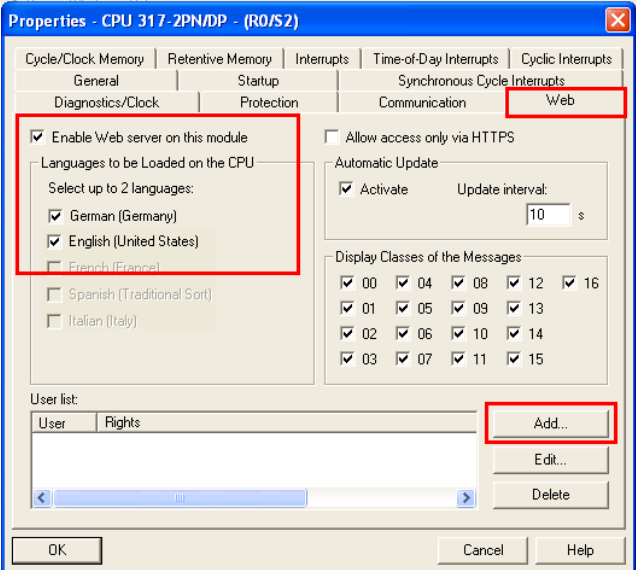
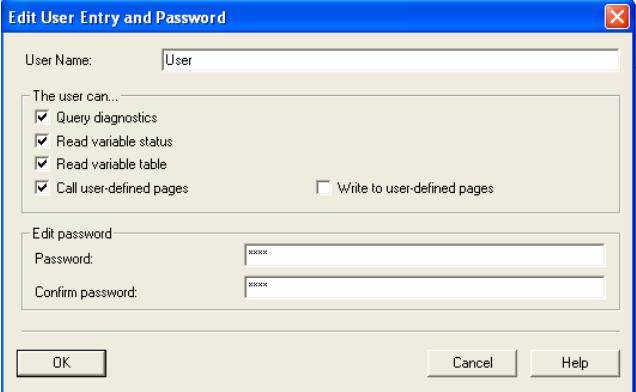
1. Configuration of the S7-300 station with the CPU 317-2 PN/DP
2. Creating the symbols in the symbol file
3. Exporting symbols, project settings and enumerations with S7-Web2PLC
4. Creating the HTML file
5. Generating project settings, enumerations and DB333 with S7-Web2PLC
6. Creating the S7 program
7. Calling the web page with a web browser

5.1 Configuration of the S7-300 station with the CPU 317-2 PN/DP

Table 5-1

No.	Action	Comment
1.	Start the SIMATIC Manager and create a new project via "File > New" with the name "web2plc_CPU317pn_dp".	
2.	Insert an S7-300 station via "Insert > Station > SIMATIC 300 Station".	
3.	Double-click "SIMATIC 300(1)" and then "Hardware".	
4.	<p>Configure the hardware.</p> <ul style="list-style-type: none"> • Drag and drop a profile rail into the station by selecting "SIMATIC S7-300 > Rack-300 > Rail" in the hardware catalog. • Drag a power supply to slot 1, e.g. the PS 307 2A. • Drag the CPU 317-2 PN/DP from the hardware catalog to slot 2. <p>The properties of the Ethernet interface are opened.</p>	
5.	In the properties of the Ethernet interface, click "New" and confirm with "OK".	
6.	<p>Select "Ethernet" as the subnet and assign the IP address of the CPU to the Ethernet interface.</p> <p>Via this IP address, you will later access the web page of the CPU with your web browser.</p> <p>Confirm all dialogs with "OK".</p>	

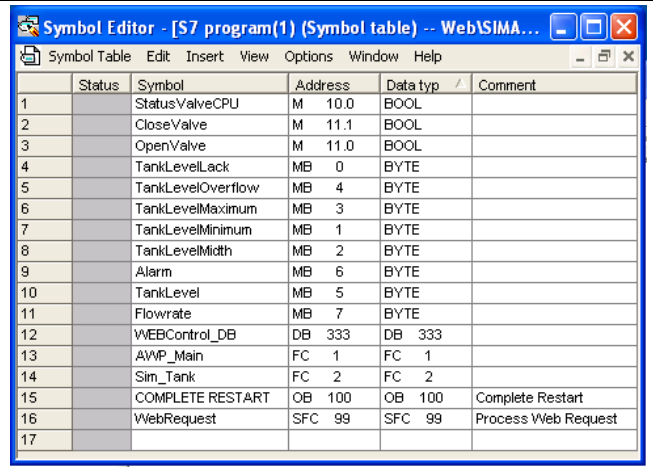
5.1 Configuration of the S7-300 station with the CPU 317-2 PN/DP

No.	Action	Comment
7.	<p>In the HW configuration, double-click the CPU 317-2 PN/DP and branch to the "Web" tab.</p> <p>Activate the web server on the CPU and then the supported languages.</p> <p>Then, click "Add".</p>	
8.	<p>Allocate users via "Add". Assign the rights and a password. Thus, the CPU will later be protected against unauthorized access.</p> <p>In the application example, "user" is used as the user name and "user" as the password.</p> <p>Confirm any message windows with "OK".</p>	
9.	<p>Save and compile the hardware configuration.</p> <p>Thus, the hardware configuration is completed.</p>	

Copyright © Siemens AG 2010 All rights reserved

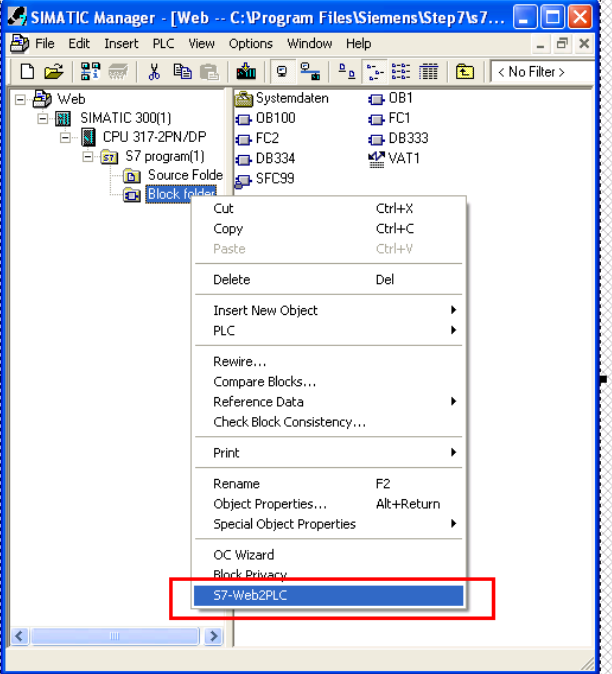
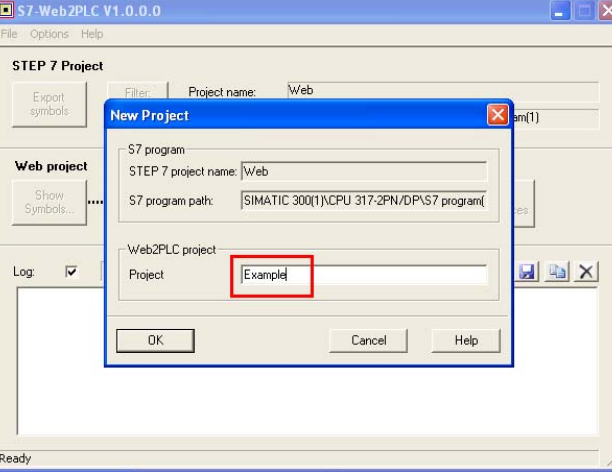
5.2 Creating the symbols in the symbol file

Table 5-2

No.	Action	Comment																																																																																										
1.	Open the symbol file by double-clicking "Symbol table" in the SIMATIC Manager of the S7 program.																																																																																											
2.	Edit the symbols and store them afterwards.	 <p>The screenshot shows the 'Symbol Editor' window for 'S7 program(1) (Symbol table)'. The window contains a table with the following data:</p> <table border="1"> <thead> <tr> <th>Status</th> <th>Symbol</th> <th>Address</th> <th>Data typ</th> <th>Comment</th> </tr> </thead> <tbody> <tr><td></td><td>StatusValveCPU</td><td>M 10.0</td><td>BOOL</td><td></td></tr> <tr><td></td><td>CloseValve</td><td>M 11.1</td><td>BOOL</td><td></td></tr> <tr><td></td><td>OpenValve</td><td>M 11.0</td><td>BOOL</td><td></td></tr> <tr><td></td><td>TankLevelLack</td><td>MB 0</td><td>BYTE</td><td></td></tr> <tr><td></td><td>TankLevelOverflow</td><td>MB 4</td><td>BYTE</td><td></td></tr> <tr><td></td><td>TankLevelMaximum</td><td>MB 3</td><td>BYTE</td><td></td></tr> <tr><td></td><td>TankLevelMinimum</td><td>MB 1</td><td>BYTE</td><td></td></tr> <tr><td></td><td>TankLevelMidth</td><td>MB 2</td><td>BYTE</td><td></td></tr> <tr><td></td><td>Alarm</td><td>MB 6</td><td>BYTE</td><td></td></tr> <tr><td></td><td>TankLevel</td><td>MB 5</td><td>BYTE</td><td></td></tr> <tr><td></td><td>Flowrate</td><td>MB 7</td><td>BYTE</td><td></td></tr> <tr><td></td><td>WEBControl_DB</td><td>DB 333</td><td>DB 333</td><td></td></tr> <tr><td></td><td>AWP_Main</td><td>FC 1</td><td>FC 1</td><td></td></tr> <tr><td></td><td>Sim_Tank</td><td>FC 2</td><td>FC 2</td><td></td></tr> <tr><td></td><td>COMPLETE RESTART</td><td>OB 100</td><td>OB 100</td><td>Complete Restart</td></tr> <tr><td></td><td>WebRequest</td><td>SFC 99</td><td>SFC 99</td><td>Process Web Request</td></tr> <tr><td></td><td></td><td></td><td></td><td></td></tr> </tbody> </table>	Status	Symbol	Address	Data typ	Comment		StatusValveCPU	M 10.0	BOOL			CloseValve	M 11.1	BOOL			OpenValve	M 11.0	BOOL			TankLevelLack	MB 0	BYTE			TankLevelOverflow	MB 4	BYTE			TankLevelMaximum	MB 3	BYTE			TankLevelMinimum	MB 1	BYTE			TankLevelMidth	MB 2	BYTE			Alarm	MB 6	BYTE			TankLevel	MB 5	BYTE			Flowrate	MB 7	BYTE			WEBControl_DB	DB 333	DB 333			AWP_Main	FC 1	FC 1			Sim_Tank	FC 2	FC 2			COMPLETE RESTART	OB 100	OB 100	Complete Restart		WebRequest	SFC 99	SFC 99	Process Web Request					
Status	Symbol	Address	Data typ	Comment																																																																																								
	StatusValveCPU	M 10.0	BOOL																																																																																									
	CloseValve	M 11.1	BOOL																																																																																									
	OpenValve	M 11.0	BOOL																																																																																									
	TankLevelLack	MB 0	BYTE																																																																																									
	TankLevelOverflow	MB 4	BYTE																																																																																									
	TankLevelMaximum	MB 3	BYTE																																																																																									
	TankLevelMinimum	MB 1	BYTE																																																																																									
	TankLevelMidth	MB 2	BYTE																																																																																									
	Alarm	MB 6	BYTE																																																																																									
	TankLevel	MB 5	BYTE																																																																																									
	Flowrate	MB 7	BYTE																																																																																									
	WEBControl_DB	DB 333	DB 333																																																																																									
	AWP_Main	FC 1	FC 1																																																																																									
	Sim_Tank	FC 2	FC 2																																																																																									
	COMPLETE RESTART	OB 100	OB 100	Complete Restart																																																																																								
	WebRequest	SFC 99	SFC 99	Process Web Request																																																																																								

5.3 Exporting symbols, project settings and enumerations with S7-Web2PLC

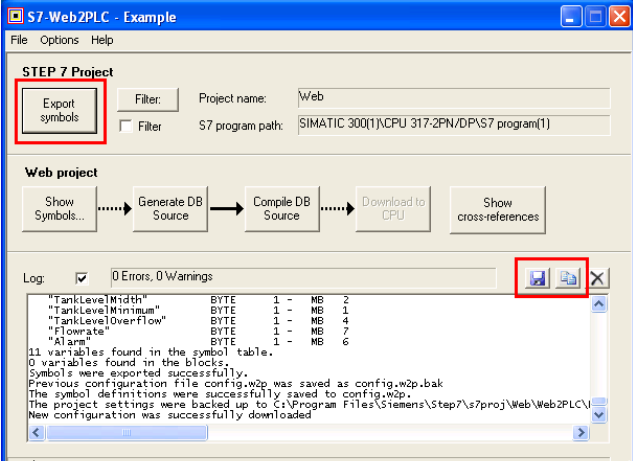
Table 5-3

No.	Action	Comment
1.	<p>Start S7-Web2PLC by right-clicking "Block folder". S7-Web2PLC is opened.</p>	
2.	<p>Create a new project via "File > New Project" and name the project "Example". Confirm with "OK". The project settings are stored in the directory "..\Step7\s7proj\Web2PLC\Example\config.w2p".</p>	

Copyright © Siemens AG 2010 All rights reserved

5 Configuration and Settings

5.3 Exporting symbols, project settings and enumerations with S7-Web2PLC

No.	Action	Comment																								
3.	<p>Export the symbols from the S7 program by clicking "Export symbols".</p> <p>Thus, you have a summary of all symbols at hand for creating the HTML file.</p> <p>You can, for example, copy or save the protocol in the clipboard via the respective symbols.</p>	 <p>The screenshot displays the 'STEP 7 Project' window in S7-Web2PLC. The 'Export symbols' button is highlighted with a red box. Below it, a workflow shows 'Show Symbols...' leading to 'Generate DB Source', 'Compile DB Source', 'Download to CPU', and 'Show cross-references'. A log window at the bottom shows a list of symbols and their addresses, with a red box around the 'Copy' and 'Save' icons.</p> <table border="1"><thead><tr><th>Symbol</th><th>Address</th><th>Size</th><th>Memory Area</th></tr></thead><tbody><tr><td>"TankLevelWidth"</td><td>1</td><td>MB</td><td>2</td></tr><tr><td>"TankLevelMinimum"</td><td>1</td><td>MB</td><td>1</td></tr><tr><td>"TankLevelOverflow"</td><td>1</td><td>MB</td><td>4</td></tr><tr><td>"Flowrate"</td><td>1</td><td>MB</td><td>7</td></tr><tr><td>"Alarm"</td><td>1</td><td>MB</td><td>6</td></tr></tbody></table> <p>Log: <input checked="" type="checkbox"/> 0 Errors, 0 Warnings</p> <p>11 variables found in the symbol table. 0 variables found in the blocks. Symbols were exported successfully. Previous configuration file config.w2p was saved as config.w2p.bak The symbol definitions were successfully saved to config.w2p. The project settings were backed up to C:\Program Files\Siemens\Step7\s7proj\Web\Web2PLC\... New configuration was successfully downloaded</p>	Symbol	Address	Size	Memory Area	"TankLevelWidth"	1	MB	2	"TankLevelMinimum"	1	MB	1	"TankLevelOverflow"	1	MB	4	"Flowrate"	1	MB	7	"Alarm"	1	MB	6
Symbol	Address	Size	Memory Area																							
"TankLevelWidth"	1	MB	2																							
"TankLevelMinimum"	1	MB	1																							
"TankLevelOverflow"	1	MB	4																							
"Flowrate"	1	MB	7																							
"Alarm"	1	MB	6																							

5.4 Creating the HTML file

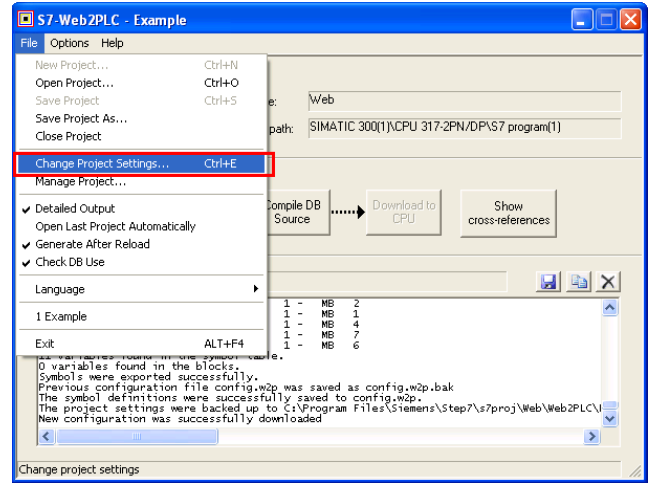
To create the HTML file, you need the list of symbols from Section 5.2 Creating the symbols in the symbol file and a corresponding editor. Convenient editors such as Microsoft Frontpage are recommendable, which automatically create tags or mark in color correct inputs already during creation, or simple editors such as Notepad.

Table 5-4

No.	Action	Comment
1.	Create the HTML file with an editor. Save the HTML file with the necessary images in the directory <code>"..\Step7\s7proj\Web\Web2PLC\Example\html"</code> .	Detailed information on the creation of the HTML file can be found in Section 3.2 Principles of user-defined web pages and 4.2 Functional principle of the HTML file

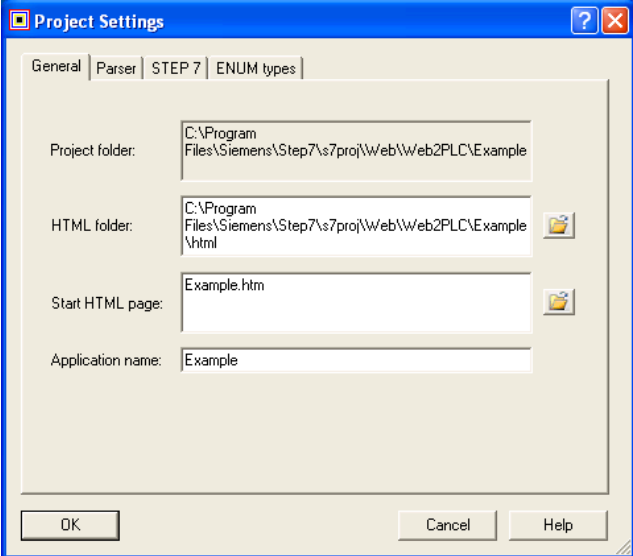
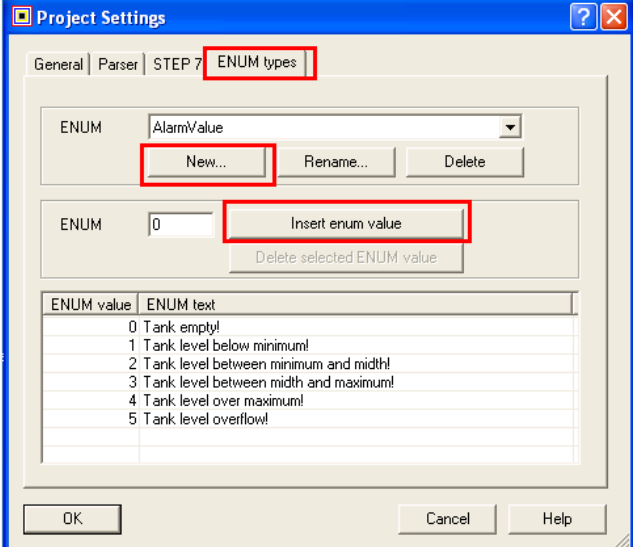
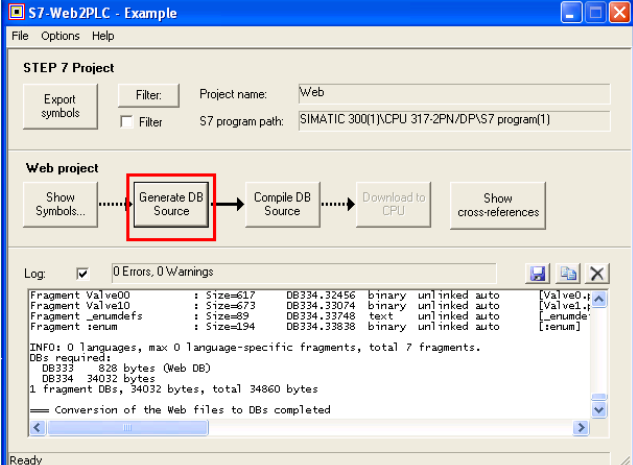
5.5 Generating project settings, enumerations and DB333 with S7-Web2PLC

Table 5-5

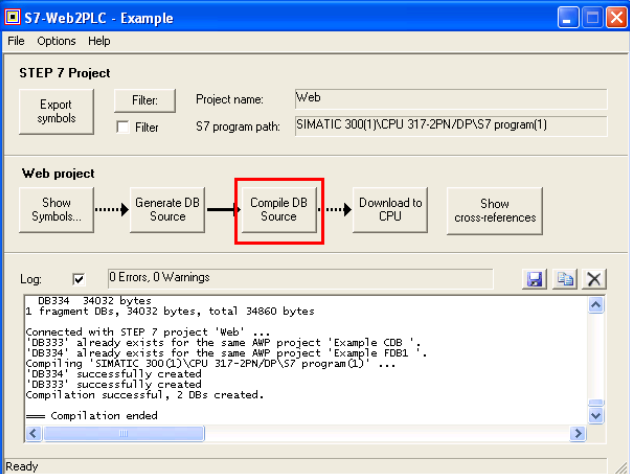
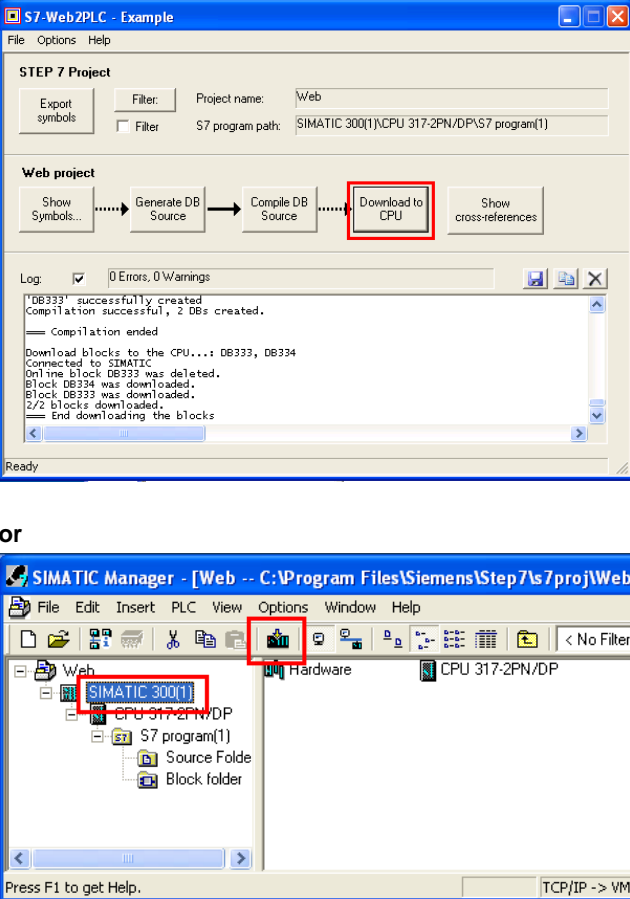
No.	Action	Comment
1.	Change the project settings, if necessary, or enter the texts for the enumerations via "File > Change Project Settings". The project settings are opened.	

5 Configuration and Settings

5.5 Generating project settings, enumerations and DB333 with S7-Web2PLC

No.	Action	Comment
2.	In the "General" tab, change the path, if necessary, and assign a start HTML page and an application name.	
3.	Texts for enumerations are entered in the "ENUM types" tab: <ul style="list-style-type: none"> Assign an ENUM type via "New". In our example, it is the "Alarm" variable from the S7 program. Via "Insert enum value", assign the texts to the ENUM type dependent on the value for "Alarm", which are later to be displayed on the web page. Confirm all dialogs with "OK". 	
4.	Generate DB333 and DB334 by clicking "Generate DB Source" in S7-Web2PLC. S7-Web2PLC verifies the project with regard to the variables, loads the complementary files such as the enumerations or images, reads the variables of the HTML file, verifies the fragments, and writes all data in DB333 and DB334. The generation of the DB source is successful if no error messages marked in red appear but the conversion of the web files to DBs is ended.	

5.5 Generating project settings, enumerations and DB333 with S7-Web2PLC

No.	Action	Comment
5.	<p>Compile the DBs by clicking "Compile DB Source".</p> <p>Thus, DB333 and DB334 are compiled.</p> <p>The compilation is successful if no error messages marked in red appear and the message "Compilation ended" appears.</p>	 <p>The screenshot shows the S7-Web2PLC interface. The 'Web project' section has a workflow: Show Symbols... -> Generate DB Source -> Compile DB Source (highlighted in red) -> Download to CPU -> Show cross-references. The log window displays the following text:</p> <pre> DB334 34032 bytes 1 fragment DBs, 34032 bytes, total 34860 bytes Connected with STEP 7 project 'Web' ... 'DB333' already exists for the same AMP project 'Example CDB'. 'DB334' already exists for the same AMP project 'Example CDB'. Compiling 'SIMATIC 300(L)\CPU 317-2PN/DP\S7 program(1)' ... 'DB334' successfully created 'DB333' successfully created Compilation successful, 2 DBs created. Compilation ended </pre>
6.	<p>There are two options for loading DB333 and DB334 into the CPU:</p> <ul style="list-style-type: none"> • If only DB333 and DB334 have been changed, click "Download to CPU" in S7-Web2PLC". Only these two DBs are transferred to the CPU. • To transfer all blocks to the CPU, select the S7-300 station in the SIMATIC Manager and click the "Download" symbol. Confirm all dialogs with "OK". All blocks are transferred to the CPU. <p>Since no block has been transmitted to the CPU yet, select the S7-300 station and transfer the entire project to the CPU.</p>	 <p>The top screenshot shows the S7-Web2PLC interface with the 'Download to CPU' button highlighted in a red box. The log window displays the following text:</p> <pre> 'DB333' successfully created Compilation successful, 2 DBs created. Compilation ended Download blocks to the CPU...: DB333, DB334 Connected to SIMATIC Online block DB333 was deleted. Block DB334 was downloaded. Block DB333 was downloaded. 2/2 blocks downloaded. End downloading the blocks </pre> <p>The bottom screenshot is titled 'SIMATIC Manager - [Web -- C:\Program Files\Siemens\Step7\projWeb]'. It shows a tree view of the project structure. The 'SIMATIC 300(1)' station is highlighted with a red box. The tree view includes:</p> <ul style="list-style-type: none"> Web <ul style="list-style-type: none"> CPU 317-2PN/DP <ul style="list-style-type: none"> S7 program(1) <ul style="list-style-type: none"> Source Folde Block folder

Copyright © Siemens AG 2010 All rights reserved

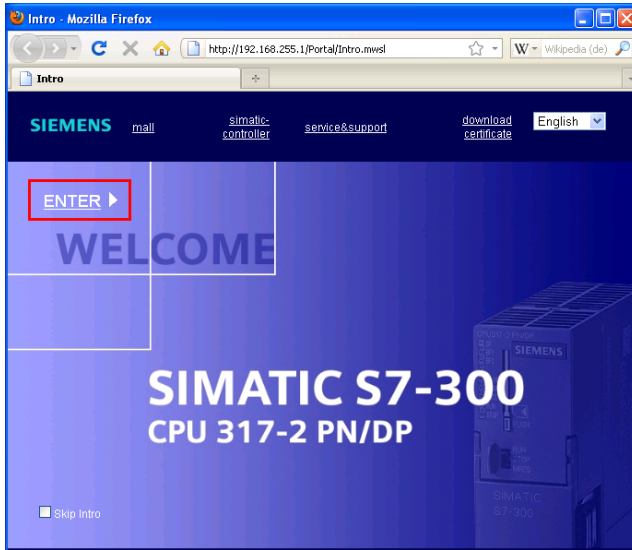
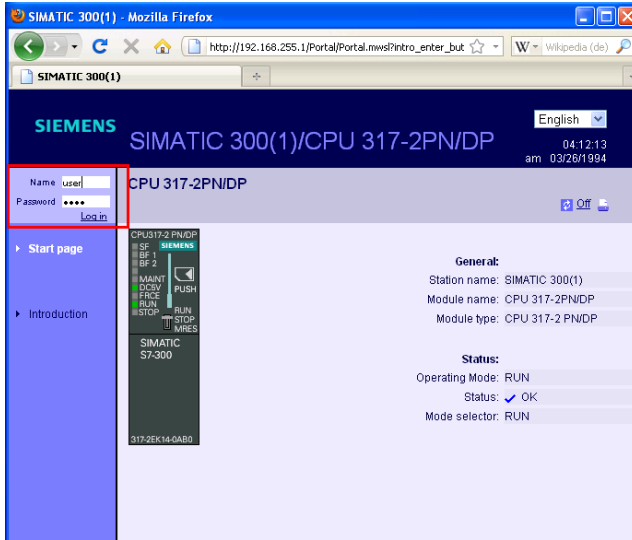
5.6 Creating the S7 program

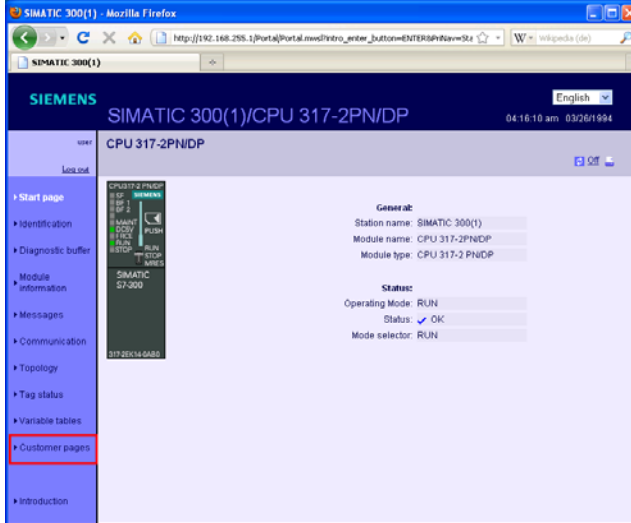
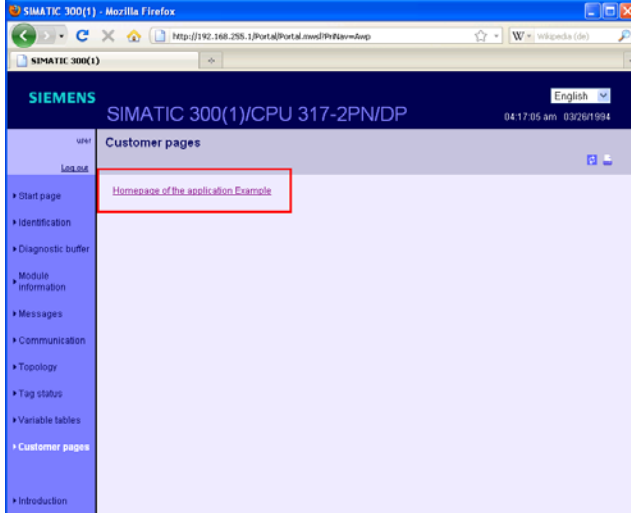
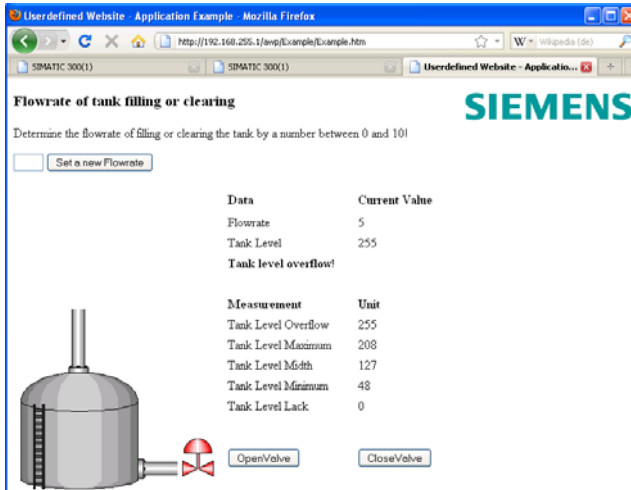
An exemplary S7 program can be found in the Attachment to this entry. The following aspects must be considered when creating the S7 program:

- Call the SFC99. The SFC99 initializes the web server of the CPU. With the cyclic calling of SFC99 you ensure that changed variables of the CPU can be displayed on the web page. The cyclic calling of SFC00 can be effected in OB1.
- To be able to evaluate variables from the web page in the S7 program, the web control DB (DB333) must be evaluated. In this application, this is done by calling FC1.

5.7 Calling the web page with a web browser

Table 5-6

No.	Action	Comment
1.	<ul style="list-style-type: none"> • Start a web browser, e.g. Internet Explorer or Mozilla Firefox. Enter the IP address of the CPU as the address, e.g. http://192.168.255.1. Note: Your PC and the CPU must be located in the same subnet. <p>The start web page of the CPU is opened.</p> <ul style="list-style-type: none"> • Click "ENTER". 	
2.	<p>Enter your name and the password which you stored in the properties of the CPU in the "Web" tab (see Section 5.1 Configuration of the S7-300 station with the CPU 317-2 PN/DP). Then, click "Login".</p> <p>The complete web page of the CPU is opened.</p>	

No.	Action	Comment																				
3.	Click "Customer pages" to go to the user-defined web page.	 <p>The screenshot shows the SIMATIC 300(1) web interface in Mozilla Firefox. The browser address bar shows the URL: http://192.168.255.1/Portal/Portal.mwd?ntro_enter_button=ENTER&view=Sta. The page title is SIMATIC 300(1). The main content area displays 'SIMATIC 300(1)/CPU 317-2PN/DP' and 'CPU 317-2PN/DP'. A sidebar menu on the left contains several items, with 'Customer pages' highlighted by a red rectangular box.</p>																				
4.	<p>On the "Customer pages" web page, all user-defined web pages belonging to the CPU are listed.</p> <p>To start the example application, click the "Homepage of the application Example".</p> <p>The "Example" web page is opened.</p>	 <p>The screenshot shows the SIMATIC 300(1) web interface with the 'Customer pages' section selected. The browser address bar shows the URL: http://192.168.255.1/Portal/Portal.mwd?ntro_enter_button=ENTER. The main content area displays 'Customer pages' and a link labeled 'Homepage of the application Example', which is highlighted by a red rectangular box.</p>																				
5.	A detailed explanation of the operation of the example web page can be found in Section 7 Application Operation.	 <p>The screenshot shows the 'Userdefined Website - Application Example' web page in Mozilla Firefox. The browser address bar shows the URL: http://192.168.255.1/awp/Example/Example.htm. The page title is 'Userdefined Website - Application Example'. The main content area displays 'Flowrate of tank filling or clearing' and 'Determine the flowrate of filling or clearing the tank by a number between 0 and 101'. There is a 'Set a new Flowrate' input field. Below this, there is a table with 'Data' and 'Current Value' columns, and a 'Measurement' and 'Unit' table. At the bottom, there is a diagram of a tank with an 'Open Valve' and 'Close Valve' button.</p> <table border="1" data-bbox="925 1556 1149 1646"> <thead> <tr> <th>Data</th> <th>Current Value</th> </tr> </thead> <tbody> <tr> <td>Flowrate</td> <td>5</td> </tr> <tr> <td>Tank Level</td> <td>255</td> </tr> <tr> <td colspan="2">Tank level overflow!</td> </tr> </tbody> </table> <table border="1" data-bbox="925 1668 1149 1792"> <thead> <tr> <th>Measurement</th> <th>Unit</th> </tr> </thead> <tbody> <tr> <td>Tank Level Overflow</td> <td>255</td> </tr> <tr> <td>Tank Level Maximum</td> <td>208</td> </tr> <tr> <td>Tank Level Midth</td> <td>127</td> </tr> <tr> <td>Tank Level Minimum</td> <td>48</td> </tr> <tr> <td>Tank Level Lack</td> <td>0</td> </tr> </tbody> </table>	Data	Current Value	Flowrate	5	Tank Level	255	Tank level overflow!		Measurement	Unit	Tank Level Overflow	255	Tank Level Maximum	208	Tank Level Midth	127	Tank Level Minimum	48	Tank Level Lack	0
Data	Current Value																					
Flowrate	5																					
Tank Level	255																					
Tank level overflow!																						
Measurement	Unit																					
Tank Level Overflow	255																					
Tank Level Maximum	208																					
Tank Level Midth	127																					
Tank Level Minimum	48																					
Tank Level Lack	0																					

6 Installation

6.1 Hardware and software installation

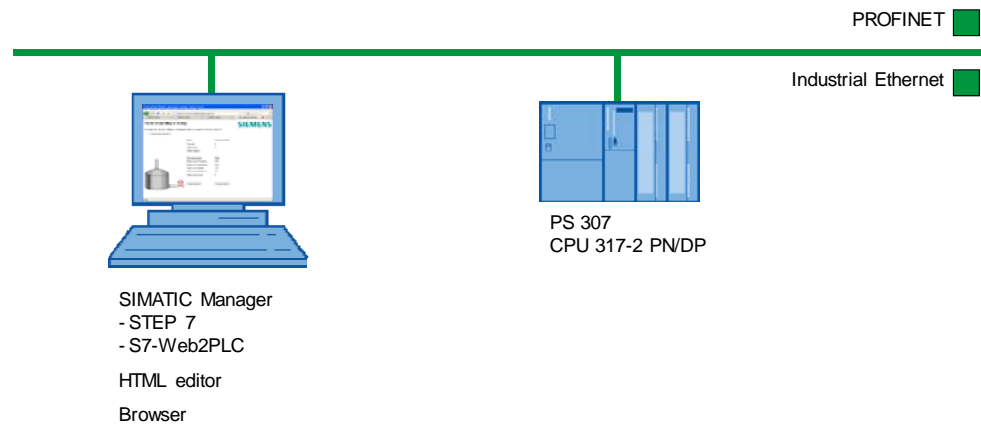
Installation of the hardware

The following figure shows the hardware structure of the example application.

The PC with the web browser must be connected to the CPU via Industrial, e.g.

- directly at the PN interface of the CPU
- via a switch

Figure 6-1



Note

Please observe the installation and connection guidelines from the corresponding manuals.

Software installation

Table 6-1

No.	Action	Comment
1	Install STEP 7 V5.5 or higher. S7-Web2PLC is included in STEP 7 V5.5 or higher.	
2	Install S7-Web2PLC.	S7-Web2PLC can be found on the STEP 7 CD under "Optional Components".
3	Install a tool for creating the web page, e.g. MS Frontpage, on the PC with which you want to create the web page.	
4	Install a web browser on the PC with which you want to access the web page of the CPU.	

6.2 Installation of the application example

Table 6-2

No.	Action	Comment
1	Start the SIMATIC Manager.	
2	Unpack the file 44212999_Web2PLC_CODE_v10.zip into the STEP 7 directory.	
3	Open the project in the SIMATIC Manager.	
4	Go to the hardware configuration.	
5	If you are using a different CPU, change the hardware configuration.	
6	In the CPU properties, assign the IP address of your CPU to the Ethernet interface.	Information in Section 5.1 Configuration of the S7-300 station with the CPU 317-2 PN/DP
7	In the SIMATIC Manager, set the access of your PG/PC to "TCP/IP" via the network card at "Options > Set PG/PC Interface". Do not use the "TCP/IP Auto" interface!	
8	Select the S7-300 station and load the entire project into the CPU.	
9	Start a web browser and call the web page of your CPU via the IP address.	Information in Section 5.7 Calling the web page with a web browser Calling the web page with a web browser

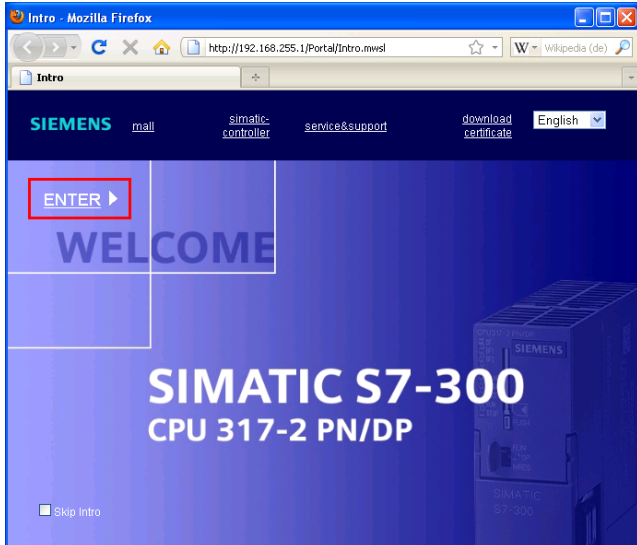
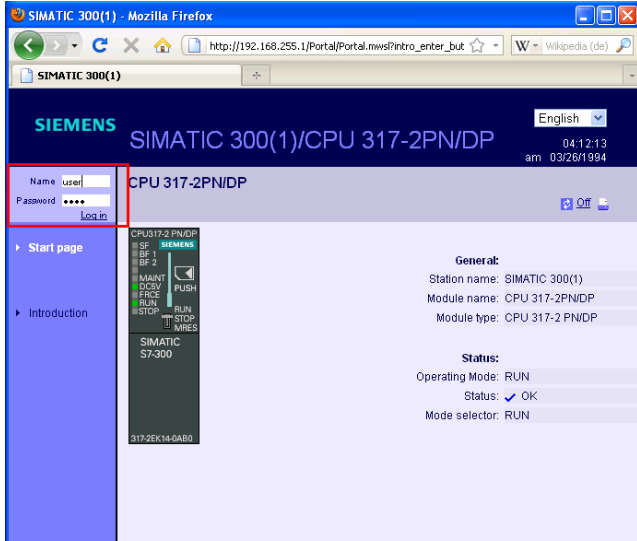
7 Application Operation

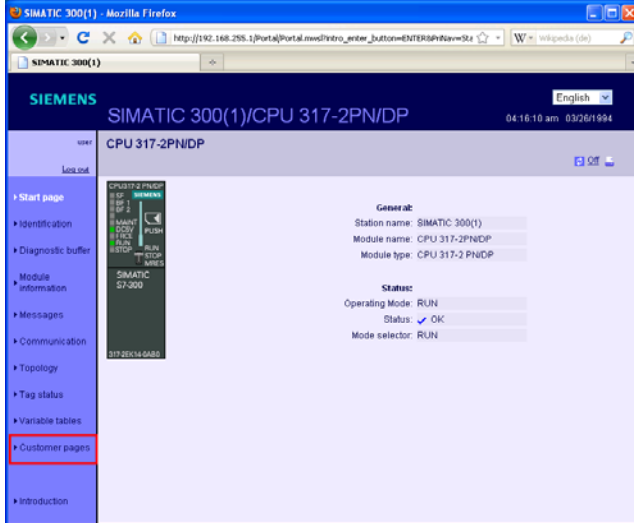
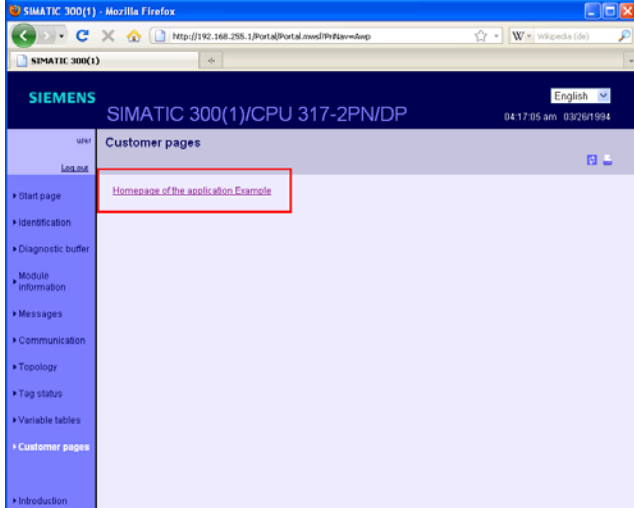
In this Section

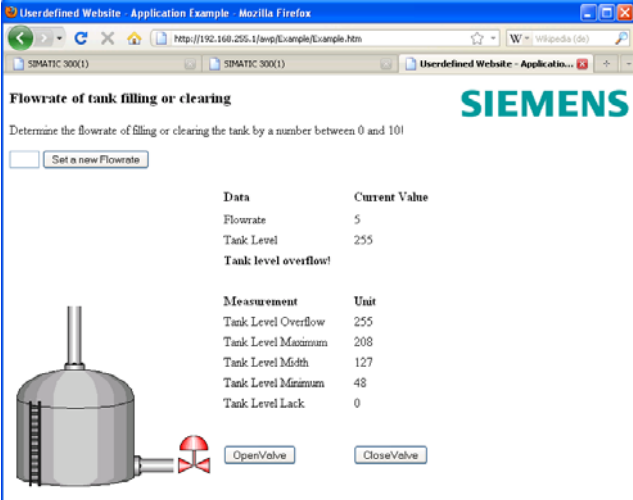
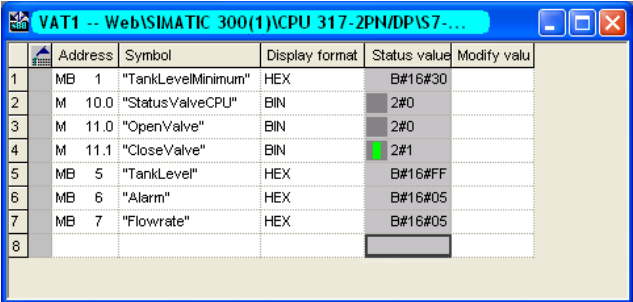
This Section provides information on how to operate the example application.

Operation

Table 7-1

No.	Action	Comment
1.	<ul style="list-style-type: none"> Start a web browser, e.g. Internet Explorer or Mozilla Firefox. Enter the IP address of the CPU as the address, e.g. http://192.168.255.1. The start web page of the CPU is opened. Click "ENTER". 	
2.	<p>Enter the name and password (stored in the properties of the CPU in the "Web" tab). In this example application, the name is "user" and the password is "user". Then, click "Login". The complete web page of the CPU is opened.</p>	

No.	Action	Comment
3.	Click "Customer pages" to go to the user-defined web page.	 <p>The screenshot shows the SIMATIC 300(1) web interface in Mozilla Firefox. The browser address bar shows the URL: http://192.168.255.1/Portal/mwd/Intro_enter_customENTER&view=Sta. The page title is 'SIMATIC 300(1)'. The main content area displays 'SIMATIC 300(1)/CPU 317-2PN/DP' and 'CPU 317-2PN/DP'. A sidebar menu on the left contains several items, with 'Customer pages' highlighted by a red rectangular box. Other items include 'Start page', 'Identification', 'Diagnostic buffer', 'Module information', 'Messages', 'Communication', 'Topology', 'Tag status', 'Variable tables', and 'Introduction'.</p>
4.	<p>On the "Customer pages" web page, all web pages belonging to the CPU are listed.</p> <p>To start the example application, click the "Homepage of the application Example".</p> <p>The "Example" web page is opened.</p>	 <p>The screenshot shows the SIMATIC 300(1) web interface in Mozilla Firefox. The browser address bar shows the URL: http://192.168.255.1/Portal/mwd/IntroENTER&view=App. The page title is 'SIMATIC 300(1)'. The main content area displays 'SIMATIC 300(1)/CPU 317-2PN/DP' and 'Customer pages'. A list of links is visible, with 'Homepage of the application Example' highlighted by a red rectangular box. The sidebar menu on the left is the same as in the previous screenshot, with 'Customer pages' selected.</p>

No.	Action	Comment
5.	<p>Via the web page, you have direct access to the CPU:</p> <ul style="list-style-type: none"> • Via "TankLevel" you can see the current filling level of the tank. Additionally, the filling level is commented via clear text. • The key figures of the filling level are displayed below. • When clicking the "OpenValve" button, the tank is cleared. Every 10 seconds, the web page is refreshed and the values are adapted. • The flow rate can be entered manually. In the S7 program, a medium flow rate of 5 is preset. The lower the value for "Flowrate" is set, the faster becomes the flow rate. • The valve is closed via "CloseValve" – the tank is filled again. 	
6.	<p>In parallel, you can monitor the change of the variables in the variable table in the SIMATIC Manager.</p>	

8 Glossary

AWP

AWP stands for the German term "Anwenderprogrammierbare Webseiten" (user-programmable web pages).

AWP command

An AWP command is understood as the special command syntax with which data are exchanged between the CPU and the HTML file.

CSS

CSS (Cascading Style Sheets) defines how a section or content marked in HTML is displayed.

HTML file

HTML files are the basis of the World Wide Web and are displayed by a web browser.

In this document, we refer to the HTML file when you are editing the web page, e.g. with Frontpage. When you are working with the web page in a web browser, we refer to it as the web page.

MIME type

With the help of the Multipurpose Internet Mail Extensions (MIME) standard, the web browser is informed – e.g. during an HTTP transfer – which data the web server sends, for example whether it is clear text, an HTML document or a PNG image.

UTF-8

UTF-8 (abbreviation for 8-bit UCS Transformation Format) is the most widely used coding for unicode characters.

In that, each unicode character is assigned a specially coded byte chain of a variable length. UTF-8 supports up to four bytes on which all unicode characters can be displayed.

Web browser

Web browsers are visualization programs for web pages and can communicate with web servers.

Typical web browsers are:

- Microsoft Internet Explorer
- Mozilla Firefox
- Google Chrome

Web server

A web server stores web pages and makes them available. The web server is a software which transfers documents with the help of standardized transfer protocols (http, HTTPS) to a web browser.

In a CPU with PROFINET interface, a web server is integrated which you can expand by self-designed web pages (user-defined web pages) with the help of S7-Web2PLC.

Web page

See HTML file.

9 References

9.1 Literature

The following list is by no means complete and only provides a selection of appropriate sources.

Table 9-1

	Topic	Title
/1/	STEP7	Automatisieren mit STEP7 in AWL und SCL Hans Berger Publicis MCD Verlag ISBN 3-89578-113-4
/2/	HTML	HTML und CSS, Praxisrezepte für Einsteiger Robert R. Agular mitp ISBN 978-3-8266-1779-9
/3/	HTML	HTML Handbuch Stefan Münz/Wolfgang Nefzger Franzis Verlag ISBN 3-7723-6654-6
/4/	Javascript	JavaScript und Ajax, Das umfassende Handbuch Christian Wenz Galileo Press ISBN 978-3-8362-1128-4

9.2 Internet links

The following list is by no means complete and only provides a selection of appropriate sources.

Table 9-2

	Topic	Title
/1/	Reference to this entry	http://support.automation.siemens.com/WW/view/en/44212999
/2/	Siemens I IA/DT Customer Support	http://support.automation.siemens.com
/3/	HTML, JavaScript	http://www.selfhtml.de/
/4/	Information on Java applets	http://support.automation.siemens.com/WW/view/en/24843906 http://support.automation.siemens.com/WW/view/en/22061339
/5/	Information on S7 Java Beans	http://support.automation.siemens.com/WW/view/en/16832609 http://support.automation.siemens.com/WW/view/en/21623882
/6/	Information on Sm@rt Service with WinCC flexible	http://support.automation.siemens.com/WW/view/en/18900228
/7/	Information on the WinCC Web Navigator	http://support.automation.siemens.com/WW/view/en/37436594

10 History

Table 10-1

Version	Date	Changes
V1.0	30.11.2010	First issue