User-defined Web Sites for HMI Panels

WinCC flexible 2008

Application Description • April 2011
Industry Automation and Drives Technologies Service & Support Portal

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


If you have any questions about this document, please contact us at the following e-mail address:

online-support.automation@siemens.com
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 accept no 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 from Siemens Industry Sector.
# Table of Contents

Warranty and Liability .................................................................................................................. 4

1 Automation Task ............................................................................................................................ 7
  1.1 Overview of the automation task ............................................................................................... 7
  1.2 Description of the automation task .......................................................................................... 7

2 Automation Solution ...................................................................................................................... 8
  2.1 Overview of the overall solution ................................................................................................ 8
  2.2 Description of the core functionality ....................................................................................... 10
  2.3 Overview and description of the user interface ....................................................................... 11
  2.3.1 STEP 7 configuration ........................................................................................................... 11
  2.3.2 WinCC flexible configuration ............................................................................................... 11
  2.3.3 HTML pages ....................................................................................................................... 15
  2.3.4 Access to the WEB server of the operator unit .................................................................... 19
  2.3.5 Sequence of the core functionality ....................................................................................... 21
  2.4 Hardware and software components used ............................................................................. 23

3 Basics ........................................................................................................................................ 25
  3.1 What is HTML? ......................................................................................................................... 25
  3.1.1 With what “Tool” can HTML pages be created? ................................................................. 26
  3.1.2 Structure of HTML TAGs .................................................................................................... 27
  3.2 HTML instructions .................................................................................................................... 28
  3.2.1 HTML basic structure .......................................................................................................... 28
  3.2.2 General HTML commands ................................................................................................. 30
  3.2.3 Creating tables ..................................................................................................................... 32
  3.3 Access to process tags of the operator unit .............................................................................. 34
  3.3.1 MWSL ................................................................................................................................. 34
  3.3.2 Access to subfolders .......................................................................................................... 34
  3.4 What is CSS? ........................................................................................................................... 35
  3.4.1 Basic structure of the CSS file ............................................................................................ 37
  3.4.2 Commands within the CSS file .......................................................................................... 38
  3.5 Relative path specification ....................................................................................................... 39

4 Configuration ............................................................................................................................... 40
  4.1 STEP 7 configuration ............................................................................................................... 40
  4.2 WinCC flexible configuration .................................................................................................. 40
  4.2.1 Tags used ............................................................................................................................ 40
  4.2.2 Configured screens ............................................................................................................. 42
  4.2.3 “Call_ServiceSite” script .................................................................................................. 47
  4.2.4 Device setting (transferring standard HTML pages) ......................................................... 47
  4.3 Standard HTML pages of the operator units ......................................................................... 48
  4.4 User-defined HTML pages ..................................................................................................... 49
  4.4.1 Preparing measures ............................................................................................................ 49
  4.4.2 Specifying storage location for the HTML pages ............................................................ 50
  4.4.3 Header data (HTML header) .............................................................................................. 51
  4.4.4 “start.html” HTML page ................................................................................................... 53
  4.4.5 “Facility_01.html” HTML page ......................................................................................... 55
  4.4.6 “Facility_02.html” HTML page ......................................................................................... 60
  4.4.7 “Facility_0x.html” HTML page ......................................................................................... 61
  4.4.8 “DesignNew.css” CSS file ............................................................................................... 72
  4.5 Web authorizations on the operator unit .................................................................................. 73

5 Startup of the Application ............................................................................................................ 76
  5.1 Preparing measures .................................................................................................................. 76
  5.2 Installation and Start up .......................................................................................................... 77
Table of Contents

6 Operating the Application .................................................................................. 80
  6.1 Operator unit, “Machine 01” screen 01 .......................................................... 80
  6.2 Operator unit, “Machine 02” screen 2 .......................................................... 81
  6.3 Operator unit, “Service Site” screen 3 ......................................................... 82
  6.4 “start page” HTML page ............................................................................ 82
  6.5 “Facility 01” HTML page ........................................................................... 83
  6.6 “Facility 02” HTML page ........................................................................... 84
  6.7 “Facility 0x” HTML page ........................................................................... 85

7 Notes, Tips and Tricks ..................................................................................... 88
  7.1 Adjusting WebContent file ......................................................................... 88
  7.2 Browser address line and updating of HTML pages .................................... 90

8 Links & Literature ......................................................................................... 91

9 History .......................................................................................................... 91
1 Automation Task

1.1 Overview of the automation task

Modern automation technology increasingly integrates Internet technologies which—together with integrated Ethernet-based communication—enable direct access to the facility via the Intranet or Internet.

For access mechanisms via the Internet or Intranet it is useful to use already existing standards such as, for example, standard Web browsers and common "languages" such as HTML or JavaScript.

The figure below provides an overview of the automation task.

![Figure 1-1](image)

1.2 Description of the automation task

Your customer has different plant sections that are visualized with several operator units.

The operator units are designed in such a way that they include the machine and plant parameters for the corresponding plant section.

The operator units are linked via an Ethernet network.

Selected machine and plant parameters of the individual plant sections are to be output and changed through a PC or laptop.

Accessing the data should be possible from every workstation.

There is no visualization software on the corresponding PCs. For this reason, the machine and plant parameters shall be output via a "standard browser", such as MS Internet Explorer.
2 Automation Solution

2.1 Overview of the overall solution

Display

The figure below shows a schematic illustration of the main components of this solution.

Figure 2-1

SIMATIC HMI operator units*) with PROFINET interface provide the opportunity to access the tag of the HMI operator unit with the help of functions provided by the system.

During the project transfer, standard HTML pages can also be transferred to the operator unit using the “Sm@rtService: HTML pages” function.

Via a standard web browser (e.g. MS Internet Explorer) they access the web server of the operator unit.

To create your individual web page (user-defined web page), you can use any tools such as, for example, Microsoft Frontpage, Notepad, etc..

For designing your web page, you can use various options provided by HTML and CSS.

In the further course of the documentation the individual terms and configuration steps and all necessary settings are described.

*) The following link contains a list of operator units which support this option (Link).
Restriction

For greater clarity, the application only describes the information that is necessary to create the “application example”.

Where necessary, the document provides references to related links and manuals.

Network connection:
The interfaces between the Ethernet network of the office and the Ethernet network of the plant section can be very different.

For information on how to establish a connection between the office network and a plant network, please contact your network administrator.

Figure 2-2

The following network configurations are possible.

- Direct connection of the nodes via an Ethernet connection (PC/laptop with the operator units).
- Adequate components, e.g. routers or gateways, are used to connect two separate Ethernet networks (separate office network and plant network).
- Connection via different network configurations.
- Please note the entry “WinCC flexible remote maintenance” (Entry ID: 19865167).

In this application, both networks (office network and facility network) are directly connected via a switch (same subnet).

Engineering tools:

this application does not include a description of

- the engineering tool STEP 7:
  the application does not describe the STEP 7 engineering tool.
  The program blocks used only serve for the simulation of process values that are displayed on the panel.
  For this reason the STEP 7 configuration is not discussed in detail.

- Engineering tool WinCC flexible:
  this application does not include a description of the WinCC flexible engineering tool.
  Basic knowledge of operating the software is assumed.
2.2 Description of the core functionality

The core of this application is the generation of user-defined web sites for HMI operator units, which enable you to read and assign machine and plant parameters of the respective plant section. The HTML pages can be directly stored on the operator unit or on an external memory card. These HTML pages can be called up via standard web browsers, such as, for example, MS Internet Explorer. Executing the function requires the “Sm@rtService: HTML pages” option on the respective operator unit.

An operator or service technician whose PC is connected with the operator unit via Ethernet can gain an overview of the facility via the HTML pages without being directly on site.

What can the application perform?

The program example on hand contains a STEP 7 program, a WinCC flexible configuration and certain HTML pages adjusted to the configuration.

You do not require knowledge of creating HTML pages.

You can, for example, use the HTML pages on hand as template for your application. In the further course of the documentation the instructions used in the HTML pages are described.

What must or can I do as a user?

“HTML” offers you a number of design options for an “Internet page”. The objective of this application is to create a basis to output machine and plant parameters through a HTML page.

The HTML page can be designed by you according to your individual needs. The Internet offers a number of “HTML seminars”, some available free of charge.

Please refer to this helpful and continuing information.
2.3 Overview and description of the user interface

2.3.1 STEP 7 configuration

A CPU 315-2DP was used for the application. The program blocks used in STEP 7 are only used for the simulation of machine data displayed on the operator unit.

Application example:

- Machine 1
  - Drive ON/OFF
  - Output of current speed
  - Defining the speed
- Machine 2
  - Drive ON/OFF
  - Output of current speed
  - Defining the speed

2.3.2 WinCC flexible configuration

An MP 277 Touch was used for the application. The project consists of three screens

- Screen 1 (machine 1)
- Screen 2 (machine 2)
- Screen 3 (service site)

In the permanent window you can call up the following functions or pages through the respective buttons.

- calling the “Screen 1” page
- calling the “Screen 2” page
- changing language
- stopping runtime
- The “Screen 3” page (service site) is called through the “HTML page”
2 Automation Solution

2.3 Overview and description of the user interface

Screen 1, Machine 01
Figure 2-3

Item (1)
“Motor 01” is controlled through the operator panel.
- Drive ON/OFF
- “Actual speed” output
- “Set speed” specification

Item (2)
This is where the IO fields are configured with different data types. They serve as additional input and output options of values.
- Bool
- Byte
- Integer
- Real
- String

Item (3)
This I/O field of the “String” type is displayed or hidden through the HTML page. Its purpose is, for example, to display a message for the operator at "Runtime".

Note
The displayed values of the I/O fields can be monitored or changed through the created HTML page.
Screen 2, Machine 02

Figure 2-4

Item (1)
“Motor 02” is controlled through the operator panel.
- Drive ON/OFF
- “Actual speed” output
- “Set speed” specification

Item (2)
This is where the I/O fields are configured with various data types. They serve as additional input and output options of values.
- Bool
- Byte
- Integer
- Real
- String

Note
The displayed values of the I/O fields can be monitored through the created HTML page.
Screen 3, Service Site

Screen 3 serves as an example how a page on the operator unit can be called through a user-defined HTML page. This may be useful, for example, if a user was entered in the “unauthorized” group due to repeated incorrect entries of the password.

Through the user administration configured in the screen, the user can be assigned a user group again.

Figure 2-5

Item (1)

User display.
The user display only serves as application example.
(No user rights have been assigned in the application)
2.3.3 HTML pages

“start.html” start page

Figure 2-6

Item (1)
The “standard start page” was expanded with the links for calling the user-defined HTML pages for the various parts of the facility.
2 Automation Solution

2.3 Overview and description of the user interface

“Facility_01.html” facility part

This page outputs machine data. The values are output through an “invisible” table, as a “creative” element.

Figure 2-7

Item (1)
“Back” button, to return to the “start page” again.

Item (2)
Picture of the facility.

Item (3)
Current machine data. The values correspond with the values shown on the HMI operator unit.

Item (4)
To update the values click the “Refresh” button.
“Facility_02.html” facility part 2

This page outputs machine data. These values are output through a “visible” table, as a “creative” element.

Figure 2-8

Item (1)
“Back” button to return to the “start page” again.

Item (2)
Picture of the facility

Item (3)
Current machine data. The values correspond with the values shown on the HMI operator unit.

Item (4)
To update the values click the “Refresh” button.
2 Automation Solution

2.3 Overview and description of the user interface

“Facility_0x.html” facility part 0x
Contrary to the two previously shown “HTML pages”, the machine data can be read out and output through this page. There is furthermore the option to display or hide a text line through a page on the HMI operator unit and to call a “Service Site”.

Figure 2-9

Item (1) “Back” button to return to the “start page” again.

Item (2) Current machine data. The values correspond with the values shown on the HMI operator unit.
In addition, values can be specified and transferred to the operator unit through these I/O fields.

**Item (3)**
To update the values click the “Refresh” button.

**Item (4)**
The text line can be displayed or hidden on the operator unit through the buttons located on the side. The purpose of the text line is to pass a message on to the staff at “runtime”.

**Item (5)**
Through the button, a configured “Service Site” can be called on the operator unit.

### 2.3.4 Access to the WEB server of the operator unit

By activating the “Sm@rtService: HTML pages” option, the standard HTML pages of the operator unit are also transferred when the configuration is loaded.

WinCC flexible Runtime and Panel Runtime provide the following standard pages:
- **start.html**: start page
- **RemoteControl.html**: remote control (only for Internet Explorer)
- **Control.html**: control functions
- **StatusDetails.html**: system diagnostics
- **Browse.html**: File Browser (only for Internet Explorer)

#### Calling the “start.html” start page

**Requirement:**
- The configuration with the enabled “Sm@rtService: HTML pages” option was transferred to the operator unit.
- The runtime has been started on the operator unit.
  (At least the operator unit has to be switched on)
- An existing connection between the operator unit and the PC/laptop

**Approach:**
1. Open your standard browser (for example, MS Internet Explorer).
2. Enter the IP address in the address line of the operator unit for example, http://172.16.34.170.
3. By clicking the “ENTER button”, a connection to the operator unit is established and the start page is called automatically.
2.3 Overview and description of the user interface

4. Depending on the selected function, a user password is required (for example, to access the file system through the “File Browser”)

**Standard name and password:**
Name: Administrator
Password: 100

**Note**
The user rights for this function are set up in the “Control Panel” on the operator unit (Link).

View of the start page:
Figure 2-10
2.3.5 Sequence of the core functionality

The precise sequence for commissioning and operating the application can be found in chapter 5 “Startup of the Application”.

Below, you will find a brief overview of the process.

Table 2-1

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>The configuration incl. the enabled “Sm@rtService: HTML pages” option was transferred to the operator unit. Due to the option, the standard HTML pages are additionally transferred to the operator unit.</td>
<td><img src="image1.png" alt="Image" /></td>
</tr>
<tr>
<td>2.</td>
<td>The user-defined HTML pages, screens etc. are stored on an external memory card.</td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
</tbody>
</table>
| 3.  | The “Start page” of the operator unit is called through a standard browser. The start page is automatically opened after entering the IP address.  
Example: http://IP address of the operator unit | ![Image](image3.png) |
| 4.  | After the login you can load your revised start page to the operator unit through the “File Browser”. The revised start page contains the extended links to call the HTML pages on the external memory card. | ![Image](image4.png) |
| 5.  | Through the revised start page, the user-defined HTML pages can now be called.  
To compare: look at section no 3. | ![Image](image5.png) |
2 Automation Solution

2.3 Overview and description of the user interface

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.</td>
<td>Sample view: The screen on the right provides you with the data of “Machine 1”.</td>
<td></td>
</tr>
</tbody>
</table>

Advantages of this solution

The solution introduced here, to store the HTML pages on an external memory card, offers the following advantages:

- The storage space on the operator unit is not used (storage space is limited)
- In case of an update of the operating system, the stored data is not overwritten. Only the start page may have to be transferred again (see also chapter 7.1)
- The HTML pages can be adjusted more quickly even if there is no connection to the operator unit
- Screens can additionally be included on the HTML pages. No special restrictions on the file size due to the storage space
- Fast exchange and change of data due to exchange of external memory card
- When exchanging the operator unit in the event of a service case, there is no additional configuration effort
- Expandability

You can also store the HTML pages directly on the operator unit. Refer to the information in the online help.
2.4 Hardware and software components used

The application was created with the following components:

**Hardware components**

<table>
<thead>
<tr>
<th>Component</th>
<th>Qty.</th>
<th>MLFB / order number</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIMATIC S7-300, CPU 315-2 DP</td>
<td>1</td>
<td>6ES7 315-2AG10-0AB0</td>
<td>Alternatively, any other S7 300/400 CPU can also be used.</td>
</tr>
<tr>
<td>CP 343-1</td>
<td>1</td>
<td>6GK7 343-1EX20-0XE0</td>
<td>In this application the communication between operator unit and controller is via Ethernet. It could also be, for example, through MPI/PROFIBUS.</td>
</tr>
<tr>
<td>MP 277 10&quot; Touch</td>
<td>1</td>
<td>6AV6 643-0CD01-1AX1</td>
<td>Every operator unit that supports the &quot;Sm@rtService: HTML page&quot; option.</td>
</tr>
<tr>
<td>MMC card</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standard PC / Laptop</td>
<td>1</td>
<td></td>
<td>Standard browser installed such as, for example, MS Internet Explorer.</td>
</tr>
<tr>
<td>Ethernet cable to connect the individual hardware components</td>
<td>3</td>
<td></td>
<td>Standard cable</td>
</tr>
<tr>
<td>Switch</td>
<td>1</td>
<td></td>
<td>To connect all Ethernet nodes</td>
</tr>
</tbody>
</table>

**Standard software components**

<table>
<thead>
<tr>
<th>Component</th>
<th>Qty.</th>
<th>MLFB / order number</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>STEP 7 V5.4 SP5</td>
<td>1</td>
<td>6ES7810-4CC08-0YA7</td>
<td>Or a higher version</td>
</tr>
<tr>
<td>WinCC flexible 2008</td>
<td>1</td>
<td>6AV6612-0AA51-3CA5</td>
<td>As of WinCC flexible 2008 &quot;Standard&quot; or later version</td>
</tr>
<tr>
<td>WinCC flexible 2008 SP2</td>
<td>1</td>
<td>SP2 download <a href="http://support.automation.siemens.com/WW/view/en/38612895">http://support.automation.siemens.com/WW/view/en/38612895</a></td>
<td>Or a higher version</td>
</tr>
<tr>
<td>Sm@rtService</td>
<td>1</td>
<td>6AV6618-7BB01-3AB0</td>
<td></td>
</tr>
</tbody>
</table>
2.4 Hardware and software components used

<table>
<thead>
<tr>
<th>Component</th>
<th>Qty</th>
<th>MLFB / order number</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>MS FrontPage</td>
<td>1</td>
<td></td>
<td>Every other software tool to create and process HTML pages.</td>
</tr>
</tbody>
</table>

**Used addresses**

The following list contains the addresses used. You can adjust them individually.

Table 2-4

<table>
<thead>
<tr>
<th>Device</th>
<th>IP address</th>
<th>PROFIBUS address</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP 277Touch</td>
<td>172.16.34.170 255.255.0.0</td>
<td></td>
</tr>
<tr>
<td>CPU 315-2DP</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>CP 343-1</td>
<td>172.16.34.160 255.255.0.0</td>
<td></td>
</tr>
<tr>
<td>PC</td>
<td>172.16.34.1 255.255.0.0</td>
<td></td>
</tr>
</tbody>
</table>

**Sample files and projects**

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

Table 2-5

<table>
<thead>
<tr>
<th>Component</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>48955975_WinCCflexible_HTML_Site_CODE_v10.zip</td>
<td>This zip file contains the STEP 7 and WinCC flexible project. Additionally it contains the prepared HTML pages.</td>
</tr>
<tr>
<td>48955975_WinCCflexible_HTML_Site_DOKU_v10.pdf</td>
<td>This document</td>
</tr>
</tbody>
</table>
3 Basics

3.1 What is HTML?

At first sight, the HTML code looks a little confusing. However, when reading a little bit about the basics, you will see that only few “instructions/commands” are required to create HTML pages.

This application deals especially with the commands necessary for accessing the HMI tags.

Furthermore, individual commands are described with which you can “design” a HTML page.

Please read the chapter “Basics” if you have no or very little basic knowledge of creating HTML pages.

The Internet offers a number of “HTML seminars”, some available free of charge. Please refer to this helpful and continuing information.

3.1 What is HTML?

HTML is not a programming language. HTML (Hypertext Markup Language) is a text-based markup language to structure contents such as

- texts,
- pictures,
- headings, etc.

HTML documents are the basic of the World Wide Web (WWW) and are displayed by a web browser (for example, MS Internet Explorer).

Apart from the contents of a Web site displayed by the browser, HTML contains additional information in the form of meta data about the language used in the text, the author etc.

The markup language is developed further by the World Wide Web Consortium (W3C).
3 Basics

3.1 What is HTML?

3.1.1 With what “Tool” can HTML pages be created?

HTML documents can be viewed and also created with any text processing program. However, for the development of HTML pages, it is recommended to use configuration tools such as, for example, “FrontPage” (part of MS Office). In addition, you can find numerous configuration tools, free of charge on the Internet.

The screen below shows an example of the different views of two HTML editors.

Table 3-1

<table>
<thead>
<tr>
<th>HTML file opened with “FrontPage”</th>
<th>HTML file opened with a “Text Editor”</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Facility 1: Pumping Station 1" /></td>
<td><img src="image2.png" alt="HTML code example" /></td>
</tr>
</tbody>
</table>

The HTML pages in this application were created with “FrontPage”.

Note: there are numerous programs that support you when creating HTML pages. However, you should try to get some basic knowledge for creating HTML pages. This is especially useful if you have to, for example, make changes on HTML pages or if you have to expand them. You already get quite far with little basic knowledge.
3.1.2 Structure of HTML TAGs

A HTML TAG is an individual HTML command and always has a “<” (less-than sign) and “>” (greater-than sign) structure.

The instructions are between the “< >” signs.
The instruction is an abbreviation.

Example:

HTML code:       <b> this is an example </b>
Browser view:    This is an example

The “b” is an instruction. The instructions are in English and the abbreviation is for bold.

Due to this command, the text between the instruction is displayed in "bold".

The “</b>” end TAG is almost identical with the start TAG. The only difference is that there is a “/” slash after the “<” less-than sign.

Summary:

• The instruction between the “< >” is an abbreviation.

• All HTML TAGs always have to be lower case! This is important for XHTML and valid web sites.

• The HTML end TAG looks like this “</instruction>”
3 Basics

3.2 HTML instructions

Below, there are some instructions that also have been used in the application. The instructions listed already offer you numerous design options.

On the Internet you will find an extensive collection of “HTML instructions” and their application spectrum. You will furthermore find detailed descriptions regarding the individual parameters with which you can configure the instructions. Simply try out the individual instructions and look at the changes with a browser.

3.2.1 HTML basic structure

The screen below shows the general structure of a HTML page. Some “HTML editors” set up this HTML structure automatically when creating a new page. Nevertheless, you can also execute this yourself or copy the existing “basic structure” of a HTML page and paste it to your new page.

Figure 3-1

```xml
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Sample-Project</title>
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1">
<meta name="description" content="" />
<link href="/CSS/DesignNew.css" type="text/css" rel="stylesheet">
</head>
<body>
Your HTML-Code
</body>
</html>
```
Explanation of terms

Please note that a detailed description of the individual items would exceed the extent of the application.

Table 3-2

<table>
<thead>
<tr>
<th>HML Code</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>7. <code>&lt;DOCTYPE ....&gt;</code></td>
<td>Document type declaration.</td>
</tr>
<tr>
<td>8. <code>&lt;html&gt;</code></td>
<td>Start and end of the HTML code. Here, the browser is informed that it is a HTML document.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>9. <code>&lt;head&gt;</code></td>
<td>Header area, where general information is given that is not shown in the browser window=&gt; meta tags. (for example, `&lt;meta name=&quot;author&quot; content=&quot;e.g. your name&quot; (information on the author)).</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>10. <code>&lt;title&gt;</code></td>
<td>The “Title” has several functions:</td>
</tr>
<tr>
<td></td>
<td>• caption of the display window</td>
</tr>
<tr>
<td></td>
<td>• name of set bookmark</td>
</tr>
<tr>
<td>11. Meta data</td>
<td>Part of the “header area”.</td>
</tr>
<tr>
<td>12. Link to the CSS file</td>
<td>Reference to the CSS file (File, in which the “Design” of the HTML page is specified). If, for example, you carry out the formatting directly on the HTML page, you do not need this file.</td>
</tr>
<tr>
<td>13. <code>&lt;body&gt;</code></td>
<td>The actual content of the HTML page starts between this instruction.</td>
</tr>
<tr>
<td></td>
<td>In contrast to the “header area”, this area is shown in a browser window.</td>
</tr>
<tr>
<td>14. Your instructions</td>
<td>This where your instructions are, for example:</td>
</tr>
<tr>
<td></td>
<td>• texts</td>
</tr>
<tr>
<td></td>
<td>• tables</td>
</tr>
<tr>
<td></td>
<td>• pictures etc.</td>
</tr>
</tbody>
</table>
### 3.2 HTML instructions

#### 3.2.2 General HTML commands

<table>
<thead>
<tr>
<th></th>
<th>HTML code</th>
<th>Comment</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>&lt;!-- Text --&gt;</td>
<td>This is how comments are noted in HTML code. The text between the instructions is not shown in the browser.</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td><code>&lt;p&gt; Text &lt;/p&gt;</code></td>
<td>Paragraph: The distance between to text lines is enlarged.</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td><code>&lt;b&gt; Text &lt;/b&gt;</code></td>
<td>The text between the instructions is displayed in &quot;bold&quot;.</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td><code>&lt;hx&gt; Text &lt;/hx&gt;</code></td>
<td>Heading: For the &quot;x&quot; specify a value from 1 to 6. (1 very big, 6 very small)</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td><code>&lt;br /&gt;</code></td>
<td>Forcing line break:</td>
<td></td>
</tr>
<tr>
<td>6.</td>
<td><code>&lt;hr /&gt;</code></td>
<td>Creating dividing line: Creates a line across the entire width of a HTML page.</td>
<td></td>
</tr>
<tr>
<td>7.</td>
<td><code>&lt;img src=&quot;&quot; alt=&quot;&quot;&gt;</code></td>
<td>Reference to picture: A <code>&lt;img&gt;</code> tag is a standalone tag without element content and without end tag. With the help of attributes you determine the details of the graphic reference. Two attributes are mandatory and always have to be specified.</td>
<td></td>
</tr>
</tbody>
</table>

**Example:**

```html
<img src="picture.jpg" alt="name of graphic">
```

- **src** = with the src attribute you specify the desired graphics file.
- **alt** = here, you enter an alternative text in case the graphic cannot be shown.

**Example of formatting for graphic:**

```html
<img src="picture.jpg" width=120 height=110 align="middle" hspace=25 vspace=25 alt="Pumping Station-2" border="1">
```

- **width** = specifying the width of the picture
- **height** = specifying the height of the picture
- **align** = "middle" aligns a graphic in the middle
- **hspace** = specifying the distance between the left and the right edge of the graphic.
- **vspace** = specifying the distance between the graphic and other elements that are located above or below it.
- **border** = specifying whether the graphic shows a border

**Note:**

these specifications can also be made in the CSS file.
### 3.2 HTML instructions

<table>
<thead>
<tr>
<th>HTML code</th>
<th>Comment</th>
<th>Screens</th>
</tr>
</thead>
</table>
| 8. <ul>  | Unsorted bullets | • Text 1  
|           |         | • Text 2  
|           |         | • Text 3 |
| 9. <ol>  | Sorted bullets | 1. Text 1  
|           |         | 2. Text 2  
|           |         | 3. Text 3 |
| 10. class=" " | Class=" " specifies that the HTML element belongs to a certain "style sheet". The element copies formats that have been stored centrally in the CSS file. Example: <table class="View2"> | |
|   | In the CSS file, for example, several groups of formatting for a table is stored (size, background color, frame color etc.). To be able to distinguish, the individual "groups" have names, for example "View2" (any name). This name "links" the desired formatting to the HTML file. |
| 11. <div> | General block element: You can "freely" position several elements, such as text, graphics, tables etc., and include them in a joint area on your HTML page. Formatting is done through "CSS". The figure shows possible specification for "formatting" the "Block Elements". In the application you can find an example for it. | |

![Diagram of CSS properties](image-url)
3 Basics

3.2 HTML instructions

<table>
<thead>
<tr>
<th>HTML code</th>
<th>Comment</th>
<th>Screens</th>
</tr>
</thead>
</table>
| 12. `<label>`
   `</label>` | For form elements such as input fields or selection lists, there normally is no logical labeling option. Although you can put text in front of such an element, for example, “E-Mail address:” but this is normal HTML text which has no defined relationship to the element which it is intended to label. With the help of labels you can create such a logic relationship between form element and labeling text. | |
| 13. `<form>`
   `</form>` | With the instruction you define a form. Everything between the introductory `<form>` tag and the final `</form>` tag is part of the form. This is mainly elements such as input fields, selection lists or buttons. | |
| 14. `<a href="start.html"> a name </a>` | Internal link: “A name” stands for any text which may, for example, contain link specifications. | |
| 15. `<a href="http://www.xyz.de/"> a name </a>` | External link: “A name” stands for any text which may, for example, contain link specifications. | |

### 3.2.3 Creating tables

Tables are often used to design HTML pages. You can define tables, for example, to display data in tables or to distribute texts and graphics more appealingly on the screen.

#### Table 3-4

<table>
<thead>
<tr>
<th>HTML code</th>
<th>Comment</th>
</tr>
</thead>
</table>
| 1. `<table>`
   `</table>` | The basic is the `<table>` HTML TAG. Every table starts like this and it ends with the corresponding END TAG. It is important that they are both set. |
| 2. `<tr>`
   `</tr>` | Creating the table row. “Tr” stands for “table row”. |
| 3. `<td>`
   `</td>` | Creating table cell. “Td” stands for “table data”. |
| 4. `<table border=1>` | Displaying the border of the table. By specifying `border =1`, the border of the table will be displayed. This can be especially useful when creating the table, to be able to see the structure of the table better when viewing it in the “browser”. The design of the table, such as, for example, formatting "with border" or "without border is done through “CSS” in this application! |
### 3.2 HTML Instructions

#### 5. Avoiding errors with tables:
- For clarity and to avoid problems, clean codes are essential, especially for tables. For a better overview, you should definitely work with indentation!
- When working with tables, it is best to number the individual cells at the beginning and to also set the border attribute straight away for the “<table>” HTML TAG, this means: `<table border="1">`.

**Example:**
```
<table border="1">
  <tr>
    <td>Row/Zeile 1</td>
    <td>Row/Zeile 2</td>
  </tr>
  <tr>
    <td>Row/Zeile 11</td>
    <td>Row/Zeile 22</td>
  </tr>
</table>
```

**Appearance of the table** (Browser view):
![Table Appearance](image)

- **colspan="x"** Merging the cells over columns to join them in a row.
  
  **"x" = number of rows that are to be merged.**

**Example:**
```
<table border="1">
  <tr>
    <td colspan="3">A</td>
  </tr>
  <tr>
    <td>1</td>
    <td>2</td>
    <td>3</td>
  </tr>
</table>
```

**View of the table** (Browser view):
![Table View](image)

- **rowspan="x"** Merging cells in a column.
  
  **"x" = number of columns that are to be merged.**

**Example:**
```
<table border="1">
  <tr>
    <td>A</td>
    <td rowspan="2">B</td>
    <td>C</td>
  </tr>
  <tr>
    <td>1</td>
    <td>2</td>
  </tr>
</table>
```

**View of the table** (Browser view):
![Table View](image)
3.3 Access to process tags of the operator unit

3.3.1 MWSL

MWSL: Mini Web Server Language

If the “Sm@rtService: HTML pages” option is installed on the operator unit, the “miniWeb” web server which is integrated in WinCC flexible is automatically executed with the start of the project.

The web server makes it possible to access the tag area of the operator unit through the integrated script language “MWSL”.

The “MWSL” language is similar to JavaScript.

The script is executed on the server (operator unit).

Access to the process tags of the operator unit:

Example:

```html
<BODY>
....
Tag value: <MWSL><!-- write(GetVar("Tag_01")); --></MWSL>
....
</BODY>
```

A precise description of the configuration and application can be found in chapter 3.3

**Note**

In HTML the “<!-- Text -->” instruction has the effect that the text between the instruction is interpreted as “comment”.

Due to the preceding instruction “<MWSL>” the operator unit detects that the text is not a “comment” but an instruction to access the tag area.

3.3.2 Access to subfolders

If you are using your WinCC flexible configuration to structure the “subfolder” tags, you have to indicate this with the tag declaration on the HTML page (Link).

The respective folders are then separated with a “\”.

**Example:**

The “Tag_01” tag is located in the “Machine_01/Tag_01” folder

**HTML code:**

```html
<BODY>
....
Tag value: <MWSL><!-- write(GetVar("Machine_01\Tag_01")); --></MWSL>
....
</BODY>
```
3.4 What is CSS?

CSS is a file that defines the appearance (design) of a HTML page. Originally “formatting” such as, for example,
- background colors,
- distances,
- font size,
- text alignment,
- text color,
- length of a row for a table etc.,

was done within the HTML page. This made the HTML code very confusing. The separation of content (HTML page) and design (CSS) distinguishes the use of current HTML.

The particularity of CSS is that by outsourcing the “formatting” to a separate file, the “formatting” is valid for all created HTML pages. Accordingly, changes on the “design” are only carried out in one central place.

“Linking” the individual formatting to the CSS file is done by the Class="" instruction.

Example: (Excerpt from a HTML page)
<table class="View2">

Description:
The “table” is assigned a formatting with the reference name “View2”. In the CSS file there is a template for table formatting with the name “View2”. Due to the assignment the table is output through the HTML browser according to the formatting.
3 Basics

3.4 What is CSS?

The table below shows the difference between a HTML page, which is shown once with the appropriate CSS file and once without.

Table 3-5

<table>
<thead>
<tr>
<th>Layout of HTML page through the CSS file</th>
<th>Layout of HTML page without CSS file</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="HTML page with CSS" /></td>
<td><img src="image2.png" alt="HTML page without CSS" /></td>
</tr>
</tbody>
</table>

**Facility 1: Pumping Station 1**

- Set-Speed: 200 rpm
- Actual-Speed: 200 rpm
- Bool: false
- Byte: 0
- Int: 0
- Real: 0

**Show text message**

- Your message:
- Show text line on the panel: False
- Hidden text line on the panel: False
- Text line active: False

**Activate Screen: Service site**

- Activate screen: Call Screen
- Reset screen request: Reset Call
- Service site active: False
3.4.1 Basic structure of the CSS file

In chapter 3.1.2 “Structure of HTML TAGs” you found out how HTML instructions are structured.
Example:
<td> (introduces a normal data cell)

In the CSS file the “td” instruction is assigned to any “reference name”. Between the instruction and the reference name there is “dot” as “separator”.
Example:
`td.View1 (instruction.reference name)`

The assigned formatting is listed within a “curly bracket”.
Example:
`td.View1 {
    formatting 1
}

If you want to assign more than one formatting specification to an instruction, the individual formatting specification separated with a “;” (“semicolon”).
Example:
`td.View1 {
    formatting 1;
    formatting 2;
    formatting 3;
}

Color specifications are usually indicated in “Hex format”.
Example:
`#808080 (gray)`

In this application size specifications are in pixel “px”.
Example:
`50px`

In the following chapter you find a list of the “formatting” used in the application. Special tables regarding the issue “CSS formatting” can be found on the Internet (“CSS commands”).
3 Basics
3.4 What is CSS?

3.4.2 Commands within the CSS file

Below is a list of the commands used in the configuration.

Table 3-6

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Comment/heading:</td>
<td>/* Text */</td>
</tr>
<tr>
<td></td>
<td>Instruction for a comment or a heading within the CSS file.</td>
<td></td>
</tr>
</tbody>
</table>
3.5 Relative path specification

The form of relative addressing within web projects is highly recommended. The reason is, for example, that the "folder structure" is always the same, but the name of the folder may change, due to the project.

**Example:**
Relative path specification: `src="../images/bild.jpg"`
Absolute path specification: `src="/images/bild.jpg"`

The two versions in the example only differ by "two dots" (`../`).
The two dots in the relative link mean: "go up in a directory".
The current directory is referenced by `./`.
For the absolute specification there are no dots but only a leading Slash (`/`). This "Slash" indicates that it is an absolute link. It always starts from the top available directory.

Storage paths of the application:

Figure 3-2

- Root directory (operator unit)
  - Start.html
- CSS
- Memory card (MMC Karte)
- HTML data
  - Facility_01.html
  - Facility_02.html
- Picture
  - Pumping_01.png
  - Pumping_02.png
- CSS

Linking the pictures within the “Facility_01.html” HTML page
`<img src="/Picture/Pump_01.png"`

Linking of the “Facility_01.html to start.html” HTML page
`<a href=".././start.html"`
4 Configuration

4.1 STEP 7 configuration

Two parts of the facility are configured as application example.

- Machine 01
- Machine 02

The following functions are executed through the STEP 7 program.

- Drive on or off
- Analyzing the speed
- Analyzing the machine mode
  (drive off/drive running/error/drive off and speed <> 0)

The STEP 7 program plays a subordinate role in this application. This is why there will be no detailed description of the program.

4.2 WinCC flexible configuration

4.2.1 Tags used

For each part of the facility, a separate folder for the tags was created. In the folders are the necessary tags for the part of the facility. In addition, a “general” tag folder was created.

View of the folders used.

Figure 4-1
“General_Tags.Call_ServiceSite” tag

Through a HTML page, a page (Service Site) can be called on the operator unit. In order to execute the function, additional properties are configured on the tag.

All other tags used in the configuration have no additional properties configured.

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>&quot;General_Tags.Call_ServiceSite&quot;</td>
<td><img src="Image" alt="General_Tags.Call_ServiceSite (Tag)" /></td>
</tr>
<tr>
<td></td>
<td>General:</td>
<td><img src="Image" alt="General_Tags.Call_ServiceSite (Tag)" /></td>
</tr>
<tr>
<td></td>
<td>Acquisition mode: Cyclic</td>
<td><img src="Image" alt="General_Tags.Call_ServiceSite (Tag)" /></td>
</tr>
<tr>
<td></td>
<td>Acquisition cycle: 100ms</td>
<td><img src="Image" alt="General_Tags.Call_ServiceSite (Tag)" /></td>
</tr>
<tr>
<td></td>
<td>Note: the acquisition mode was selected in a way that the tag in the “background” is updated.</td>
<td><img src="Image" alt="General_Tags.Call_ServiceSite (Tag)" /></td>
</tr>
</tbody>
</table>

2. | "General_Tags.Call_ServiceSite" | ![General_Tags.Call_ServiceSite (Tag)](Image) |
| | Events: if there is a “Change value”, the "Call_ServiceSite” script is called. | ![General_Tags.Call_ServiceSite (Tag)](Image) |
| | Through this script the configured “Service Site” is called [Link](Image). | ![General_Tags.Call_ServiceSite (Tag)](Image) |
4.2 WinCC flexible configuration

4.2.2 Configured screens

In the enclosed configuration three screens are configured. The functions configured in the screens are described below.

Permanent window

Table 4-2

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td><strong>Permanent window:</strong> four buttons are configured in the permanent window. These buttons</td>
</tr>
<tr>
<td></td>
<td>• call the individual screens directly</td>
</tr>
<tr>
<td></td>
<td>• changes the language of the user interface (English/German)</td>
</tr>
<tr>
<td></td>
<td>• stop the runtime</td>
</tr>
<tr>
<td></td>
<td>An animation is configured on the “Screen 1” and “Screen 2” buttons. When the respective screen is loaded, the background color of the button is displayed in green. The tag is set when the screen is loaded and is reset when the screen is cleared (Link).</td>
</tr>
</tbody>
</table>
### Screen 1, Machine 01

**Table 4-3**

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.</td>
<td>“Screen 1” – Properties of the screen:</td>
<td><img src="image1.png" alt="Image" /></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>the background color of the “Screen 1” button is changed through an animation. When loading the screen, the “Bit” of the “Screen_1_Activ” tag is set and when clearing the screen it is reset (Link).</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Open the properties of “Screen 1”.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Events:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Entry loaded/cleared “SetBit”/”ResetBit”</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>“Screen 1” – Control unit:</td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
<tr>
<td></td>
<td>in the top part of the screen there is a control unit to control “Machine 01”. Through an animation of the motor graphic (background color) the current operating mode is output.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>“Motor” graphic:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Animation – Design:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>the tags are assigned to the individual “operating modes” through the STEP 7 program.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Drive off (gray)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Drive running (green)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Drive error (red)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Drive off and speed &lt;&gt; 0 (orange)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Buttons:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>When a button is pressed, a bit is set through the buttons and when the button is released it is reset.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>I/O fields:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Through the two I/O fields the values of the speed are specified or read out. (Mode: input/output or output).</td>
<td></td>
</tr>
</tbody>
</table>
### 4.2 WinCC flexible configuration

#### Screens

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
</tr>
</thead>
</table>
| 4.  | “Screen 1” – Tag fields:  
In the bottom part of the screen there are various “tag fields”. According to the text on the right, the tags are used with the specified data type used in the I/O fields. The fields only serve for additional input or output of values for the configured HTML page ([Link]).  
All tags used in this area do not have PLC connection.  
In addition, a second I/O field of the “String” data type is used. This I/O field can be displayed or hidden through an instruction on a “HTML page” ([Link]).  
“String” I/O field:  
- Animation – Visibility:  
  The used tag is assigned the value “1” or “0” through a HTML page.  
  - Value “0”: I/O field invisible  
  - Value “1”: I/O field visible |

---

**Screen 2, Machine 02**

#### Screens

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
</tr>
</thead>
</table>
| 1.  | “Screen 2” – Screen properties:  
The background color of the “Screen 2” button is changed through an animation. When loading the screen the “Bit” of the “Screen_2_Activ” tag is set and when clearing the screen it is reset ([Link]).  
Open the properties of “Screen 2”.  

Events:  
- Entry loaded/cleared  
  “SetBit”/“ResetBit” |
2. **“Screen 2” – Control unit:**
   In the top part of the screen there is a control unit to control “Machine 02”.
   Through an animation of the motor graphic (background color) the current
   operating mode is output.

   **“Motor” graphic:**
   - **Animation – Design:**
     the tags are assigned to the individual “operating modes” through
     the STEP 7 program.
     - Drive off (gray)
     - Drive running (green)
     - Drive error (red)
     - Drive off and speed <> 0
       (orange)

   **Buttons:**
   When a button is pressed, a bit is set through the buttons and when the button
   is released it is reset.

   **I/O fields:**
   Through two I/O fields the values of the speed are specified or read out.
   (Mode: input/output or output).

3. **“Screen 2” – Tag fields:**
   In the bottom part of the screen there are various “tag fields”. According to the text
   on the right, the tags are used with the specified data type used in the I/O fields.
   The fields only serve for additional output of values for the configured HTML page
   (Link).
   All tags used in this area do not have PLC connection.
   In contrast to “Screen 1”, no second I/O field of the “String” data type is used.
### Screen 3, Service Site

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screenshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.</td>
<td>“Screen 3” – User administration: this page can only be called through the configured HTML page (<a href="#">Link</a>). The configured user display only serves as application example. In the project itself no user rights were assigned. This page shows how a page on the operator unit can be called through a “HTML” page.</td>
<td><img src="image.png" alt="Screenshot" /></td>
</tr>
</tbody>
</table>
4.2.3 “Call_ServiceSite” script

The script is executed if there are value changes of the “General_Tags.Call_ServiceSite” tag.
“Screen 3” (Service Site”) is then called through the script.

**Background:**
the “General_Tags.Call_ServiceSite” tag is assigned a fixed value (“0” or “1”) through a HTML page.
A function can be configured through the “Events -> Value change” parameter.
If, for example, the “enable picture” function would be configured directly to the tag, the function would be executed every time there is a value change of the tag.
In order for the function only to be executed at a defined value, in this case at a tag value of “1”, the "Call_ServiceSite" script is used.

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screenshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>“Call_ServiceSite” script</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
</tbody>
</table>

**Row 4, current request:**
“Screen 3” is only called when the “General_Tags.Call_ServiceSite” tag has the value "1" (True).

**Row 6, set value of tag to “Zero”**: Due to this instruction, the value of the tag is reset to “zero” (false) again.

4.2.4 Device setting (transferring standard HTML pages)

To be able to transfer the standard HTML pages with the configuration to the operator unit, you have to enable the “Sm@rtService: HTML pages” option.

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screenshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>“Sm@rtService: HTML pages” option</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
</tbody>
</table>

The “Sm@rtService: HTML pages” option can be found in the menu tree under “Device settings > Device settings” and there under the “Runtime services” item.
4.3 Standard HTML pages of the operator units

For the operator units listed below, there are “Standard HTML pages” that can be transferred with the configuration and the enabled “Sm@rtService: HTML pages” option to the respective operator units:

- 270 series
- 370 series
- PC with WinCC flexible Runtime
- XP177B
- MP177
- Mobile Panel 177 PN
- Mobile Panel 277
- Mobile Panel 277 IWLAN
- Mobile Panel 277 IWLAN V2
- Only in view mode
  - Mobile Panel 277F IWLAN
  - Mobile Panel 277F IWLAN (RFID Tag)
  - Mobile Panel 277F IWLAN V2

**Storage path of the “Standard HTML pages”**

The “WebContent” zip file with the standard HTML pages for the operator units can be found in the installation directory of WinCC flexible.

**Example:**

C:\Program Files\Siemens\SIMATIC WinCC flexible\WinCC flexible 2008\Transfer\1.3\operator unit\WebContent.zip

You can also use the Windows search function. (Search word: WebContent).
4.4 User-defined HTML pages

The structure of the configured HTML pages is basically identical.

- Document type/declaration
- Header data
- Body
- Texts, references and pictures

To structure the pages a “table structure” is used. This makes it possible to “simply” align individual texts and tags on the HTML page.

The “Design” (formatting), for example, of the “table” etc. is stored in a separate CSS file.

**Note**

In the application, only the “standard HTML start page” of the operator unit is adjusted. All other existing HTML pages remain unchanged. The newly created user-defined HTML pages are stored separately on an external memory card.

4.4.1 Preparing measures

**Standard HTML pages of the operator units:**

First of all carry out the points listed below, before you process the “standard HTML pages” of the operator unit.

- Select the required “WebContent.zip file” for your operator unit
- Create a safety copy of the selected “WebContent.zip file”
- Unzip the “WebContent.zip file” in a separate folder

**View of the unzipped “WebContent” zip file:**

![Unzipped WebContent.zip file]

The further course of the application describes how you continue with the processed files.
4.4 User-defined HTML pages

4.4.2 Specifying storage location for the HTML pages

You should consider upfront where to store your user-defined HTML pages and the files, such as, for example, the CSS file or images on the operator unit. This has the advantage that you do not have to adjust the links within the HTML pages later.

“start.html” HTML page

The processed “start.html” start page is stored in the “root directory” of the operator unit.

The “original” start page is replaced by the revised start page.

User-defined HTML pages and other files

The user-defined HTML pages, the CSS file and the “facility pictures” are stored on an external MMC memory card on the operator unit.

The screen below shows the file structure on the MMC card.

Figure 4-3

The user-defined “HTML pages” and the “CSS” and “Picture” folder are stored in the “HTML_Data” folder.

The “DesignNew.css” file is located in the “CSS” folder.

The facility pictures are in the “Picture” folder.

Details to the individual HTML files and the CSS file are described in the chapters that follow.
4.4.3 Header data (HTML header)

If a new HTML page is created, for example, through a “HTML editor”, depending on the setting, a “HTML header” with “meta information” is created automatically. The “header data” used in this application is used as an example and is sufficient for this application. (The HTML page is displayed correctly even without “meta information”)

On the Internet you will find further information regarding the “HTML header” issue.

Figure 4-4

<table>
<thead>
<tr>
<th>Row</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row 1</td>
<td>For a HTML page that is conforming to the standard, it is recommended to note down a “Doctype declaration”. (Can be used like this as template)</td>
</tr>
<tr>
<td>Row 3</td>
<td>Start of the HTML code. Here, the browser is informed that it is a HTML document.</td>
</tr>
<tr>
<td>Row 5</td>
<td>Here, the description of the header part (HTML header) starts.</td>
</tr>
<tr>
<td>Row 7</td>
<td>Page title: With the instruction the name of the HTML document is specified. (You should adjust this name individually)</td>
</tr>
<tr>
<td>Row 9</td>
<td>With this “meta information” the used set of character is specified. (Can be used like this as template)</td>
</tr>
<tr>
<td>Row 10</td>
<td>Page description: With this “meta information” you can give brief information about the “HTML page”. It should not exceed 256 characters. (You should adjust the content individually)</td>
</tr>
<tr>
<td>Row 11</td>
<td>Author: This specification identifies you as the author of the HTML document.</td>
</tr>
<tr>
<td>Row 13</td>
<td>Link to the “CSS file” The design of the HTML page is through the CSS file. (Can be omitted, if the formatting is done within the HTML page).</td>
</tr>
<tr>
<td>Row 15</td>
<td>This application ends with the “header part”.</td>
</tr>
</tbody>
</table>
4 Configuration

4.4 User-defined HTML pages

**Row marking for orientation**

Figure 4-5

For better orientation, most “HTML editors” display a “pagination bar” on the left. The documentation always refers to it.

**Example:**
Copy the content of the rows 3 to 7.

**“Split” FrontPage view**

The “FrontPage” HTML editor provides various views to look at the HTML code.

To create the HTML pages, the “Split” view was used.

In the top part of the window, the HTML code is displayed and in the bottom part of the window, the preview view is displayed.

Figure 4-6
4.4.4 "start.html" HTML page

This application uses the standard "start.html" HTML page which is included in the delivery with the installation of WinCC flexible.

The "start.html" page is only expanded by the links for accessing the newly created HTML pages.

You can individually specify where you want to locate the new links. The positioning made here, only serves as an example.

Table 4-9

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Open the &quot;start.html&quot; page with a HTML editor. The new &quot;links&quot; are placed between the &quot;Control Functions&quot; and &quot;System Diagnostics&quot; fields.</td>
<td><img src="image1.png" alt="Image" /></td>
</tr>
<tr>
<td>2.</td>
<td>Copying existing links as template:</td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
<tr>
<td></td>
<td>• mark rows &quot;84 to 89&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• copy the marked rows</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• put the cursor in row &quot;89&quot; behind the &lt;br&gt; instruction and press the ENTER button to insert a new row.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• paste the previously copied rows in the new row</td>
<td></td>
</tr>
<tr>
<td></td>
<td>You can look at the &quot;result&quot; in the preview window.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Note: make sure that the newly added rows are located below the &quot;old&quot; cells. This serves for better visibility and readability.</td>
<td></td>
</tr>
</tbody>
</table>

View in preview window:

![Image](image3.png)
### 4 Configuration

#### 4.4 User-defined HTML pages

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.</td>
<td>Adjusting copied link:</td>
<td>Links to the “Control.html” page:</td>
</tr>
<tr>
<td></td>
<td>Look at row “94”. Among others, you will find the entries • href=… • Control Functions</td>
<td><code>&lt;a href=&quot;/control.html&quot; class=&quot;ad_new_link&quot;&gt;Control Functions&lt;/a&gt;</code></td>
</tr>
<tr>
<td></td>
<td>Replace the /Control.html link with the link you want to refer to.</td>
<td><code>View of the changed link:</code> <code>&lt;a href=&quot;/StorageCardMMC/HTML_Data/Facility_01.html&quot; class=&quot;ad_new_link&quot;&gt;Facility 01&lt;/a&gt;</code></td>
</tr>
<tr>
<td></td>
<td>In this case: /StorageCardMMC/HTML_Data/Facility_01.html (storage location/folder/name of file)</td>
<td><code>View in preview window:</code> <img src="image1.png" alt="Image" /></td>
</tr>
<tr>
<td></td>
<td>Replace the “Control Function” name with the name that refers to the link.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>In this case: Facility 01</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The other specifications are not changed.</td>
<td>Note: If necessary, you can also change the other instructions (layout) in row “94”. However, this requires further knowledge of HTML.</td>
</tr>
<tr>
<td>4.</td>
<td>Adding other links:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>to add other links, it is best to copy the newly inserted rows with the changes made there. Afterwards only adjust the link and the name of the link.</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>Complete view of the processed “start.html” page with three inserted links.</td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
</tbody>
</table>
4.4.5 “Facility_01.html” HTML page

Through the “Facility_01.html” HTML page, machine specific data of “Machine 01” is output.

As compared to the “Facility_02.html” HTML, the values are output in a structured way in an “invisible” table.

The class=”” information listed in the instructions relates to formatting specification that is stored in the CSS file.

---

**Note**

To follow the description, open the enclosed “Facility_01.html” HTML page. Due to the limited column width, the enclosed pictures cannot always reproduce the code in “detail”.

---

### Table 4-10

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Row 1 to 18: Information on the “header data” (Link) issue and the basic structure of a HTML page (Link) has already been described before.</td>
<td><img src="image" alt="HTML code" /></td>
</tr>
<tr>
<td>2.</td>
<td>Row 21 to 29: Table in which a button is integrated. Through the button you get back to the previous page.</td>
<td><img src="image" alt="HTML code" /></td>
</tr>
</tbody>
</table>

**HTML code:**

```html
<!-- HTML code -->
```

**Browser view:**

![Browser view](image)
### 4 Configuration

#### 4.4 User-defined HTML pages

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Row 21 to 22:</strong></td>
<td>Comment (not displayed on the HTML page).</td>
<td></td>
</tr>
<tr>
<td><strong>Row 23 and 29:</strong></td>
<td>Table structure (start and end of table). How to create a table can be read in chapter 3.2.3</td>
<td></td>
</tr>
</tbody>
</table>
| **Row 26:** | (View without information regarding formatting) | *<p><input type="button" value="-- back" onclick="location.href='../../start.html'"></p>*  
  - The instruction was used to be able to center the button in the table.  
  - **Parameter for a button**  
    *<input type="button" value="-- back" onclick="location.href='../../start.html'">*  
    You define a button with the `input type="button"`. The labeling of the button is specified with the `value` attribute.  
    - `onclick ="location.href='../../start.html'">`  
      With `onclick` you specify when the event is triggered (in this case when pressing the button). After specifying `href` the link that is to be called is entered. In this case the “start.html” page is called.  
      **Note:** the “Facility_01.html” HTML page is stored in a folder on an external memory card. The “start.html” HTML page is located directly in the “Root directory” of the operator unit. For the “link” to work on the “start.html” page, enter a “relative path”.  
      - Related to this example, this means that the “start.html” file is 2 levels higher that the “Facility_01.html” HTML page.  
      - This is expressed by the “./../” specification. |         |
| **Row 32 to 36:** | General “block element”: The block element only serves for the “optical” structure of the HTML page. |         |
| **Row 32 to 33:** | Comment (not displayed on the HTML page). |         |
| **Row 34 to 36:** | Instruction for the “block element”. |         |

```html
3 32 <!-- Allgemeines Block-Element zur Gliederung -->
33 <!-- General Block-Element for structuring -->
34 </div class="flext">
35
37 </div>
```

**Browser view:**

```
---
---
---
---
---
---
---
---
---
---
---
---
---
---
---
---
---
---
```
### 4 Configuration

#### 4.4 User-defined HTML pages

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
</table>
| 4.  | **Row 39 to 41:**  
Comment (not displayed on the HTML page).  
---  
**Row 41:**  
Instruction for a heading  
\texttt{<h2>Facility 1: Pumping Station 1</h2>} | **HTML code:**  
\[33\]  
\[39\] \texttt{<h2>Facility 1: Pumping Station 1</h2>} |
| 5.  | **Row 44:**  
Link to the facility picture.  
---  
**Total instruction:**  
\texttt{<img src="/Picture/Pump_01.png" width=120 height=99 align="middle" hspace=25 vspace=25 alt="Facility-1 Pumping Station-1" border="1">}  
\texttt{<img src="/Picture/Pump_01.png" alt="Facility-1 Pumping Station-1">} | **HTML code:**  
\[5\]  
\[39\] \texttt{<h2>Facility 1: Pumping Station 1</h2>}  
---  
**Browser view:**  
Facility 1: Pumping Station 1  
---  
Note:  
In this case the formatting (width and height etc.) of the picture is not done through the “CSS file”.  
Below, the same instruction without “formatting information”:  
\texttt{<img src="/Picture/Pump_01.png" alt="Facility-1 Pumping Station-1">} |
| 6.  | **Row 47 to 87:**  
Table with different tags to output machine values. In addition, the “HTML page” can be updated through a button.  
The table consists of 5 columns and 6 rows.  
In this case, column “4” serves as “spacer” so that the button in column “5” has a defined distance to the other cells.  
The formatting of the table is through the CSS file. In this example, the columns of the table are not displayed (invisible table).  
How to create a table can be read in chapter 3.2.3  
---  
**Row 47 to 48:**  
Comment (not displayed on the HTML page). | **Example of table view:**  
<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

Browser view:

<table>
<thead>
<tr>
<th>Set speed:</th>
<th>1 rpm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual speed:</td>
<td>1 rpm</td>
</tr>
<tr>
<td>Book:</td>
<td>Name/Unit</td>
</tr>
<tr>
<td>Byte:</td>
<td>Name/Unit</td>
</tr>
<tr>
<td>Integer:</td>
<td>Name/Unit</td>
</tr>
<tr>
<td>Real:</td>
<td>Name/Unit</td>
</tr>
</tbody>
</table>

Refresh
### 4.4 User-defined HTML pages

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row 52:</td>
<td>Through the following instruction, tags can be read out from the operator unit.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>&lt;MWSL&gt;&lt;!-- write(GetVar(&quot;Machine_01\Machine_01.Set_Speed_01&quot;)); --&gt;&lt;/MWSL&gt;</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The &quot;Machine_01.Set_Speed_01&quot; tag is stored on the operator unit under the &quot;Machine_01&quot; folder.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The &quot;&quot; separator precedes each subfolder.</td>
<td></td>
</tr>
<tr>
<td>Note:</td>
<td>the instruction used only has read access to the tag of the operator unit.</td>
<td></td>
</tr>
<tr>
<td>Row 55:</td>
<td>The HTML page is updated through the following instruction.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>`(View without information regarding formatting).</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>&lt;input type=&quot;button&quot; value=&quot;Refresh&quot; onclick=&quot;window.location.reload()&quot;&gt;</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Parameter for a button</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>&lt;input type=&quot;button&quot; value=&quot;Refresh&quot; onclick=&quot;window.location.reload()&quot;&gt;</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td>You define a button with the <code>input type=&quot;button&quot;</code> attribute.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The labeling of the button is specified with the <code>value</code> attribute.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- <code>onclick=&quot;window.location.reload()&quot;&gt;</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td>With <code>onclick</code> you specify when the event is to be triggered (in this case when pressing the button).</td>
<td></td>
</tr>
<tr>
<td></td>
<td>After the instruction, According to this instruction a &quot;Window function&quot; is called. The function has the effect that the HTML page is updated.</td>
<td></td>
</tr>
<tr>
<td>Row 60 / 66 / 72 / 78 / 84:</td>
<td>Other tags of the operator unit are read out in listed cells.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>General instruction for the tags of the operator unit.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>&lt;MWSL&gt;&lt;!-- write(GetVar(&quot;tag of the operator unit&quot;); --&gt;&lt;/MWSL&gt;</code></td>
<td></td>
</tr>
<tr>
<td>Row 90 to 97:</td>
<td>In this example, a &quot;form element&quot; and a &quot;block element&quot; are combined with each other. This serves as an alternative display to a table.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>In the example, a tag of the &quot;String&quot; type is read by the operator unit.</td>
<td></td>
</tr>
<tr>
<td>HTML code:</td>
<td></td>
<td><img src="image" alt="HTML code" /></td>
</tr>
<tr>
<td>Browser view:</td>
<td></td>
<td><img src="image" alt="Browser view" /></td>
</tr>
</tbody>
</table>
### 4.4 User-defined HTML pages

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Row 90 to 91:</strong></td>
<td>Comment (not displayed on the HTML page).</td>
<td></td>
</tr>
<tr>
<td><strong>Row 92 and 97:</strong></td>
<td>Form Element.</td>
<td></td>
</tr>
<tr>
<td><strong>Row 93 and 96:</strong></td>
<td>Block Element.</td>
<td></td>
</tr>
<tr>
<td><strong>Row 94 and 95:</strong></td>
<td>Label for element.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Between the &quot;label element&quot; there is the &quot;name&quot; and the &quot;instruction&quot; for the access to the tag of the operator unit.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>&lt;label&gt;String-Text : &lt;/label&gt;</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>&lt;label&gt;&lt;MWSL&gt;&lt;!-- write(GetVar(&quot;Machine_01\Test_Value_String1&quot;)); --&gt;&lt;/MWSL&gt;&lt;/label&gt;</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Row 100 to 104:</strong></td>
<td><strong>HTML code:</strong></td>
</tr>
</tbody>
</table>
|  | General "block element": The block element only serves for the "optical" structure of the HTML page. | ```
90  <!-- Allgemeines Block-Element zur Strukturierung -->
91  <!-- General Block-Element for structuring -->
92  <div class="view">  
93  </div>
94  </div>
``` |
|  | **Row 102 to 104:** | **Browser view:** |
|  | Instruction for the "block element". | ![Browser view](image)

---

9. **Row 100 to 104:**
General "block element":
The block element only serves for the "optical" structure of the HTML page.

Line 100 to 101
Comment (not displayed on the HTML page).

Row 102 to 104:
Instruction for the "block element".

<div>  
</div>
**4 Configuration**

**4.4 User-defined HTML pages**

### 4.4.6 “Facility_02.html” HTML page

Through the “Facility_02.html” HTML page, machine specific data of “Machine 02” is output.

As compared to the “Facility_01.html” HTML, the values are output in a structured way in a “visible” table.

The class=”” information listed in the instructions relates to formatting specification that is stored in the CSS file.

**Note**

The configuration is identical with the configuration of the “Facility_01.html” HTML page.

The difference is in the “formatting” of the table and in the tag declaration.

---

**Difference between the “Facility_01.html” and “Facility_02.html” HTML page.**

In the table below, two “properties” are listed that explain the differences.

Table 4-11

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>“Facility_01.html” HTML page: “Invisible” table. The formatting of the table is through the CSS file.</td>
<td><img src="image1" alt="screenshots" /></td>
</tr>
<tr>
<td>2.</td>
<td>“Facility_02.html” HTML page “Visible” table. The formatting of the table is through the CSS file.</td>
<td><img src="image2" alt="screenshots" /></td>
</tr>
</tbody>
</table>
| 3.  | Tag declaration Machine 01:  

```xml
<MWSL><!-- write(GetVar("Mashine_01\Mashine_01.Actual_Speed_01")); --></MWSL>
```

| 4.  | Tag declaration Machine 02:  

```xml
<MWSL><!-- write(GetVar("Mashine_02\Mashine_02.Actual_Speed_01")); --></MWSL>
```
4.4.7 “Facility_0x.html” HTML page

Through the “Facility_0x.html” HTML page, machine specific data of “Machine 01” is output.

In contrast to the “Facility_01.html” HTML page, from this page you can

- specify machine-specific data through the HTML page and transfer it to the operator unit,
- call a page on the operator unit through which the HTML pages are called,
- display or hide a text row on the operator unit through the HTML page.

The class=”” information listed in the instructions relates to formatting specification that is stored in the CSS file.

Note

To follow the description, open the enclosed “Facility_01.html” HTML page. Due to the limited column width, the enclosed pictures cannot always reproduce the code in “detail”.

Table 4-12

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
</table>
| 1.  | Row 1 to 18:  
Information regarding the “header data” (Link) issue and the basic structure of a HTML page (Link) has already been described before. | ![Image](image1.png) |
| 2.  | Row 21 to 29:  
Table in which a button is integrated. Through the button you get back to the previous page. | ![Image](image2.png) |

HTML code:

```html
<!-- HTML code goes here -->
```

Browser view:

![Image](image3.png)
4.4 User-defined HTML pages

Row 21 to 22:
Comment (not displayed on the HTML page).

Row 23 and 29:
Table structure (start and end of table).
How to create a table can be read in chapter 3.2.3

Row 26:
(View without information regarding formatting)
\(<p><input type="button" value="<-- back" onclick ="location.href='../../start.html'">\)</p>

- \(<p>\)
The instruction was used to be able to center the button in the table.

- **Parameter for a button**
\(<input type="button" value="<-- back"\>
You define a button with the `input type="button"`. The labeling of the button is specified with the `value` attribute.

- `onclick ="location.href='../../start.html'"`
With `onclick` you specify when the event is to be triggered (in this case when pressing the button). After specifying `href` the link that is to be called is entered. In this case the “start.html” page is called.

**Note:**
the “Facility_01.html” HTML page is stored in a folder on an external memory card. The “start.html” HTML page is located directly in the “Root directory” of the operator unit. For the “link” to work on the “start.html” page, enter a “relative path”.
- Related to this example, this means that the “start.html” file is 2 levels higher that the “Facility_01.html” HTML page.
This is expressed by the “../../” specification.

3. Row 32 to 36:
General “block element”:
The block element only serves for the “optical” structure of the HTML page.

Row 32 to 33:
Comment (not displayed on the HTML page).

Row 34 to 36:
Instruction for the “block element”.
\(<div> \</div>\)
Row 39 to 73:
Declaration of the tags used.

In contrast to the HTML pages described before, in this application example all tags that are to be described through the HTML page, have to be "declared" first.

The instructions are commented out with the preceding <!-- instruction. However, this has no influence on the function itself. The operator unit interprets the application accordingly.

4.4 User-defined HTML pages

Excerpt from the tag declaration

Row 39 to 40:
Comment (not displayed on the HTML page).

Row 41:
<MWSL> instruction
Access to the tag area of the operator unit.

Row 42:
<!-- instruction
Instruction to comment out the following tag declaration so that it does not have to be displayed on the HTML page.
Due to the preceding “MWSL” instruction, the operator device does not analyze the subsequent instructions as “comment”.

Row 43 to 44: (Example for a variable declaration)
If (ExistVariable("Set-Speed", "URL"))
{SetVar("Machine_01\Machine_01.Set_Speed_01", GetVar("Set-Speed", "URL"));}

Set-Speed: Reference name for the later I/O field. The name must only be used once in the HTML page (see description under -> “Name”, Link).

Machine_01\Machine_01.Set_Speed_01: Storage path of the tags

Notes:
• the complete instruction could also be in one line.
• Note the "bracket terms" used
• It is best to copy the complete instruction and only adjust the reference name and the storage path to avoid "typing errors" within the instruction.

Row 72:
--> instruction
End of the instruction to comment out the tag declaration (not displayed on the HTML page).

Row 73:
</NWSL> instruction
End of the instruction for accessing the tag area of the operator unit.
### 4 Configuration

#### 4.4 User-defined HTML pages

<table>
<thead>
<tr>
<th>Row 76 to 77:</th>
<th>HTML code:</th>
</tr>
</thead>
</table>
| Comment (not displayed on the HTML page). | ```
75 <!-- Überschrift -->
76 <!-- Headline -->
77 <h2>Facility 1: Pumping Station 1</h2>
79 ``` |

**Browser view:**

Facility 1: Pumping Station 1

<table>
<thead>
<tr>
<th>Row 81 to 82:</th>
<th>Example of table view:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comment (not displayed on the HTML page).</td>
<td>Table with different tags to output machine values. In addition, the &quot;HTML page&quot; can be updated through a button. The table consists of 5 columns and 6 rows. In this case, column &quot;4&quot; serves as &quot;spacer&quot; so that the button in column &quot;5&quot; has a defined distance to the other cells. The formatting of the table is through the CSS file. In this example, the columns of the table are not displayed (invisible table). How to create a table can be read in chapter 3.2.3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Browser view:**

Example of table view:
In contrast to the "Facility_01.html" and "Facility_02.html" HTML pages, values can be read-in and read-out through these I/O fields. Additionally, a registration is required to operate the I/O fields.

```
<form>
  <tr>
    <td>Actual-Speed:</td>
    <td><input class="View0" name="Actual-Speed" type="text" <MWSL>
      if (ShareRealm("NO_REALM")){write("disabled");} </MWSL>
      value="<MWSL>WriteVar("Machine_01\Machine_01.Actual_Speed_01");
      </MWSL>" readonly></td>
    <td>rpm</td>
  </tr>
</form>
```

Row 94 and 100: (<form> and </form>)
For each tag within the table, a "form area" is defined. By clicking the "ENTER button", the value is copied in the I/O field and transferred to the "operator unit".

Row 96:
Text field

Row 97: (Parameter)
Input: The <input> tag creates an input field in a form.
Name: Reference text. This name has to match the text that you used at the beginning of the "tag declaration" (see also Link)
Type="text": For the <input> tag, various attributes can be assigned. In this case a "text entry" is declared.
Value: Assigns the path to the I/O field (path specification and name of the tags).
Name and Value form a "name/value pair".
The data within the form is transferred to the operator unit.
Readonly: Due to this instruction, the "I/O field becomes a pure output field". If you leave the instruction out, the values can be input and output through the I/O field.
MWSL: The instruction accesses the tag area of the operator unit.
ShareRealm: To be able to access the tags of the operator unit, you have to be logged on at the operator unit first. If there is no authorization, you cannot specify values through the I/O fields.

Note:
it is best to copy the complete instruction and only adjust the reference name (name="" and the tag name incl. storage location (value=""), in order to avoid "typing errors" within the instruction.
4 Configuration

4.4 User-defined HTML pages

7. Row 136 to 143:
In this example, a “form element” and a “block element” are combined with each other. This serves as alternative display to a table.
In the example, a tag of the “String” type can be read-in and read out from the operator unit.

HTML code:

```html
<form>
  <label>String:</label>
  <input class="View5" name="String_01" type="text" value="\(\text{WriteVar("Machine_01\Test_Value_String1")}\)">
</form>
```

Browser view:

String: This is a sample Text from the HMI-Device MP 277 Touch

Row 136 to 137:
Comment (not displayed on the HTML page).

Row 138 and 143:
Form Element.

Row 139 and 142:
Block Element.

Row 140 and 141:
Label for element.

Between the “label element” there is the “name” and the “instruction” for the access to the tag of the operator unit.

```html
<label>String:</label>
<input class="View5" name="String_01" type="text" value="\(\text{WriteVar("Machine_01\Test_Value_String1")}\)">
```

Note:
it is best to copy the complete instruction and only adjust the reference name (name="") and the tag name incl. storage location (value=""), in order to avoid “typing errors” within the instruction.

8. Row 146 to 150:
General “block element”. The block element only serves for the “optical” structure of the HTML page.

HTML code:

```html
<dl>
  <dt>String: </dt>
  <dd>\(\text{WriteVar("Machine_01\Test_Value_String1")}\)
</dl>
```

Browser view:

---

Row 146 to 147:
Comment (not displayed on the HTML page).

Row 148 to 150:
Instruction for the “block element”.

```html
<dl>
  <dt>String: </dt>
  <dd>\(\text{WriteVar("Machine_01\Test_Value_String1")}\)
</dl>
```
9. **Row 153 to 154:**
Comment (not displayed on the HTML page).

**HTML code:**
```html
153 <!-- Überschrift -->
154 <!-- Headline -->
155 <h2>Show text message</h2>
```

**Browser view:**

Show text message

---

10. **Row 158 to 165:**
In this example, a "form element" and a "block element" are combined with each other. This is used as an alternative display of a table.

In the example, a tag of the "String" type can be read-in and read out from the operator unit.

In the further course it is shown how the "text message" can be displayed or hidden on the operator unit.

**HTML code:**
```html
158 <!-- Tag: form element -->
159 <input type="text" name="string" id="text" placeholder="Enter value"/>
```

**Browser view:**

Your message: Please contact Mr. Schmidt. Tel.: 4789

---

11. **Row 168 to 169:**
Comment (not displayed on the HTML page).

**Row 170 to 193:**
Table as design element for structuring texts and buttons.

The table consists of 2 columns and 3 rows.

How to create a table can be read in chapter 3.2.3

**HTML code:**
```html
170 <table>
171   <tr>
172     <td>Text line on the panel</td>
173     <td>Hidden text line on the panel</td>
174   </tr>
175   <tr>
176     <td>Text line active</td>
177     <td>Hidden text</td>
178   </tr>
179 </table>
```

**Browser view:**

Show text line on the panel: 
Fade In Text

Hidden text line on the panel: 
Hidden Text

Text line active: 
false
### 4 Configuration

#### 4.4 User-defined HTML pages

<table>
<thead>
<tr>
<th>Row 171 to 177: (button to display the text message)</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;form&gt;</code></td>
</tr>
<tr>
<td><code>input name=&quot;Text_Line_OnOff&quot; type=&quot;hidden&quot;</code></td>
</tr>
<tr>
<td><code>&lt;MWSL&gt;</code> if (ShareRealm(&quot;NO_REALM&quot;))(write(&quot;disabled&quot;);) <code>&lt;/MWSL&gt;</code> value=&quot;1&quot;&gt;</td>
</tr>
<tr>
<td><code>&lt;tr&gt;</code></td>
</tr>
<tr>
<td><code>&lt;td&gt;</code>Show text line on the panel:<code>&lt;/td&gt;</code></td>
</tr>
<tr>
<td><code>&lt;td&gt;</code>input type=&quot;submit&quot; value=&quot;Fade In Text&quot;&gt;&lt;/td&gt;`</td>
</tr>
<tr>
<td><code>&lt;/tr&gt;</code></td>
</tr>
<tr>
<td><code>&lt;/form&gt;</code></td>
</tr>
</tbody>
</table>

**Row 171 and 177:** `<form> and </form>`
The HTML form tag creates the border of the form and contains the control information for further processing.
By clicking the button, the **Form data** for further processing is transferred. In this case, the value "1" is transferred to the specified tag.

**Row 172:**
`input name="":` Reference text. This name has to match the text that you used at the beginning of the "tag declaration" (see Link).
`type="hidden":` Through the "hidden" parameter, the "input field" is not displayed on the screen (invisible).
`MWSL:` The instruction accesses the tag area of the operator unit.
`ShareRealm:` To be able to access the tags of the operator unit, you have to be logged in at the operator unit first. If there is no authorization, you cannot specify values through the I/O fields.
`value="1":` The tag is given a fixed value of "1".

**Row 174:**
Text field

**Row 175:** (button)
`input type ="submit":` With `<input type="submit">`, you define a send button. When clicking these buttons the form data is sent.

**Note:**
It is best to copy the complete instructions and only adjust the reference name to avoid "typing errors" within the instruction.

<table>
<thead>
<tr>
<th>Row 179 to 185: (button for hiding the text message)</th>
</tr>
</thead>
<tbody>
<tr>
<td>See previous description (row 171 to 177).</td>
</tr>
</tbody>
</table>
### 4.4 User-defined HTML pages

#### Row 187 to 192: (Value of the tags)

To see whether the button for displaying the text message has already been clicked, the current value of the tag is output.

```html
<form>
  <tr>
    <td>Text line active:</td>
    <td>
      <input name="Text_Line_OnOff" type="text" 
      value="\{\text{ShareRealm(\"NO_REALM\") if (\text{ShareRealm(\"NO_REALM\")}) \text{write("disabled");}} \text{\text{WriteVar("Machine_01\Test_Value_Visible");}} \text{\text{\"\) \text{readonly}}}">
    </td>
  </tr>
</form>
```

#### Row 189:

Text field

#### Row 190: (I/O field)

**input name="Text_Line_OnOff":** Reference text. This name has to match the text that you used at the beginning of the "tag declaration" (see Link).

**Type="text":** For the `<input>` tag, various attributes can be assigned. In this case a "text entry" is declared.

**MWSL:** The instruction accesses the tag area of the operator unit.

**ShareRealm:** To be able to access the tags of the operator unit, you have to be logged in at the operator unit first. If there is no authorization, you cannot specify values through the I/O fields.

**Value** Assigns the path to the I/O field (path specification and name of the tags).

**Name** and **Value** form a "name/value pair".

**Readonly:** Due to this instruction, the "I/O field becomes a pure output field".

**Note:** it is best to copy the complete instruction and only adjust the reference name (**name=""**) and the tag name incl. storage location (**value=""**), in order to avoid "typing errors" within the instruction.

#### Row 196 to 200:

General "block element".

The block element only serves for the "optical" structure of the HTML page.

#### Row 196 to 197:

Comment (not displayed on the HTML page).

#### Row 198 to 200:

Instruction for the "block element".

```html
<div>  </div>
```
### 4.4 User-defined HTML pages

<table>
<thead>
<tr>
<th>Row</th>
<th>Description</th>
<th>HTML Code</th>
<th>Browser View</th>
</tr>
</thead>
<tbody>
<tr>
<td>13.</td>
<td><strong>Row 203 to 204:</strong> Comment (not displayed on the HTML page).</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| | **Row 205:** Instruction for a heading  
<h2>Activate Screen: Service site</h2> |  |  |
| 14. | **Row 208 to 209:** Comment (not displayed on the HTML page). |  |  |
| | **Row 210 to 233:** table as design element for structuring texts and buttons.  
The table consists of 2 columns and 3 rows.  
How to create a table can be read in chapter 3.2.3 |  |  |
| | **Row 211 to 217:** (Button to “enable” the service site on the operator unit)  
The functionality is the same as the one as described previously for the rows “171 to 177”.  
Due to the value assignment the tags change their value, this calls a script which in turn calls the “Service Site” ([Link](#)). |  |  |
| | **Row 219 to 225:** (button for disabling the call of the service site)  
See previous description (row 211 to 217). |  |  |

**Note:**  
The bit *) which is set through the “Call Screen” button, is subsequently reset automatically through a script ([link](#)).  
It has to be noted that the previous call in the “command line” of the Internet Browsers is “active” until a new instruction is triggered.  
This is why the instruction can be reset again through the “Reset Call” ([Link](#)).  

* To implement the function, the tag used needs PLC connection.
### 4.4 User-defined HTML pages

#### Row 227 to 232: (Value of the tags)
To see whether the button to enable the "Service Site" has already been clicked, the current value of the tag is output.

#### 15. Row 236 to 240:
**General "block element".**
The block element only serves for the "optical" structure of the HTML page.

**Row 236 to 237:**
Comment (not displayed on the HTML page).

**Row 238 to 240:**
Instruction for the "block element".

<code>&lt;div&gt; &lt;/div&gt;</code>

<table>
<thead>
<tr>
<th>HTML code:</th>
</tr>
</thead>
<tbody>
<tr>
<td>235</td>
</tr>
<tr>
<td>236</td>
</tr>
<tr>
<td>237</td>
</tr>
<tr>
<td>238</td>
</tr>
<tr>
<td>239</td>
</tr>
<tr>
<td>240</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Browser view:</th>
</tr>
</thead>
<tbody>
<tr>
<td>--------------</td>
</tr>
</tbody>
</table>
4.4.8 “DesignNew.css” CSS file

The “design” of the HTML page in this application is through the CSS file. For this purpose, look at the information in chapter 3.4.1 “Basic structure of the CSS file” or chapter 3.4.2 “Commands within the CSS file”. There is no further description here.

The screen below shows a section from the CSS file.

Figure 4-7

```css
/* Hintergrundfarbe der HTML Seite */
body { background-color: #000000; }

/* Abstract Design */
/* New paragraph Design */
p, p01 { text-align: center; }

/* Table Design */
/* Table Design */
table, td { width: 400px; border: 1px solid; background-color: #FFFF00; }

/* Table Design */
/* Table Design */
td { text-align: left; vertical-align: middle; width: auto; height: auto; }

td02 { text-align: left; vertical-align: middle; width: 100px; height: auto; }

td03 { text-align: left; vertical-align: middle; width: 100px; height: auto; }

td04 { text-align: left; vertical-align: middle; width: 100px; height: auto; }
```
4.5 Web authorizations on the operator unit

Introduction

The user-related access to the services used with Sm@rtAccess and Sm@rtService is controlled with the user administration for the web server.

In this application the created HTML pages are not directly stored on the operator unit in the “root directory” but on an external memory card. To be able to access these HTML pages through a standard browser, you have to be logged in at the web server before.

Logging in on the web server is also necessary, for example, if you want to access the tags of the configuration.

Note

If you save the user-defined HTML pages directly in the “root directory” of the operator unit and if you only have read-access to the tags of the configuration than logging into the web server is not required.

Setting up user-rights on the web server

Table 4-13

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>WinCC Internet Settings:</td>
<td><img src="image" alt="WinCC Internet Settings" /></td>
</tr>
<tr>
<td></td>
<td>Open the “WinCC Internet Settings” through the control panel on the operator unit.</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>Web Server:</td>
<td><img src="image" alt="Web Server" /></td>
</tr>
<tr>
<td></td>
<td>Open the “Web Server” tab in the menu screen and subsequently click the “Unser Administration” button.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>After clicking the “User Administration” button, a password prompt appears. Enter the standard password “100” and confirm the entry with “OK”. The “UserDatabase-Edit” menu is opened:</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> You can change the Administrator password anytime in the “UserDatabase-Edit” menu described below.</td>
<td></td>
</tr>
</tbody>
</table>
4.5 Web authorizations on the operator unit

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.</td>
<td>UserDatabase-Edit:</td>
<td>![Image]</td>
</tr>
</tbody>
</table>

The respective “User” can be selected through the “drop-down list”.

“User Manager” tab
- Serves to select, newly create or delete a user.
  By clicking “New” and entering the password twice, a new user is created.
  By clicking “Remove” a user is deleted.
  “Apply” makes the new creations or deletion valid.

“Description” tab
- Description of the user selected in the “User Manager” tab for precise identification (optional).

“Authorizations” tab
- Enables the granting or withdrawal of web authorizations.
  Web authorizations are included in the list.
  By clicking “Add” or double-clicking the respectively marked web authorization, the rights are added to the user selected in the “User Manager” tab.
  By clicking “Remove”, the respectively marked web authorization is removed from the rights of the selected user in the “User Manager” tab.

Note:
For access to the file system and the stored HTML pages you need the “FileBrowserUser” web authorization.

Confirm your entries “OK” and the window is closed again.
4.5 Web authorizations on the operator unit

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.</td>
<td>Restart of the web server</td>
<td>![Restart window screenshot]</td>
</tr>
<tr>
<td></td>
<td>After confirming the entries with the “OK” button, the note on the right appears. Confirm the restart of the Web server by clicking the “Yes” button. The window is closed again.</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>Creating other users:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>to create other users, repeat the steps as described under point 3 If you do not want to make any other entries, close the window with the “OK” button. Access authorizations for the web server are thus completed.</td>
<td></td>
</tr>
</tbody>
</table>
5 Startup of the Application

5.1 Preparing measures

5 Startup of the Application

5.1 Preparing measures

Table 5-1

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.</td>
<td>Linking all nodes:</td>
</tr>
</tbody>
</table>

Link all nodes through the Ethernet and make sure that there is a connection to all nodes.

- Control panel
- Controlling
- PC / Laptop

7. Transferring WinCC flexible configuration:

Transfer the WinCC flexible configuration. Make sure that the “Sm@rtService: HTML pages” option is also selected in the device settings (Link).

8. Transferring S7 configuration:

Transfer the STEP 7 configuration to the controller.

9. Preparing memory card:

The user-defined HTML pages the CSS file and the facility pictures are stored in this application on a MMC memory card.

Copy the files listed below to the MMC card. Note the used folder structure for the individual files. For this purpose, look at chapter 4.4.2 “Specifying storage location for the HTML pages”:

- Facility_01.html
- Facility_02.html
- Facility_0x.html
- DesignNew.css
- Facility pictures

Subsequently insert the MMC in the intended storage space on the operator unit.

Note:
You can also create the individual folders through the operator unit and copy the files to the MMC memory card (Link).
## 5.2 Installation and Startup

Table 5-2

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Calling homepage of the operator unit:</td>
<td><img src="image" alt="Screenshot of SIMATIC HMI Miniweb on HMI Panel" /></td>
</tr>
<tr>
<td></td>
<td>enter the IP address of the operator unit in the address line of the Web browser and confirm the entry with the ENTER button.</td>
<td><img src="image" alt="Screenshot of SIMATIC HMI Miniweb on HMI Panel" /></td>
</tr>
<tr>
<td></td>
<td>The homepage of the operator unit is called.</td>
<td><img src="image" alt="Screenshot of SIMATIC HMI Miniweb on HMI Panel" /></td>
</tr>
<tr>
<td></td>
<td><strong>In this example:</strong></td>
<td><img src="image" alt="Screenshot of SIMATIC HMI Miniweb on HMI Panel" /></td>
</tr>
<tr>
<td></td>
<td><a href="http://172.16.34.170">http://172.16.34.170</a></td>
<td><img src="image" alt="Screenshot of SIMATIC HMI Miniweb on HMI Panel" /></td>
</tr>
</tbody>
</table>
### 5.2 Installation and Startup

#### 5 Startup of the Application

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.</td>
<td><strong>Loading user-defined start page:</strong></td>
<td><img src="image" alt="File Browser" /></td>
</tr>
</tbody>
</table>

Transfer the processed “start.html” start page directly in the “root directory” of the operator unit.

- Login to the Web server.
  - Name: **Administrator**
  - Password: **100**
- Click the “File Browser” entry on the left of the navigation bar.
- Click the “www-Root” folder. The processed start page is inserted in this directory.

**View of the “www-Root” directory.**

**Selecting the processed “start.html” page:**

- Click the “Browse…” button in the “Directory Operations” field and go to the folder in which your processed “start.html” HTML file is located.
- Mark the “start.html” file
- Click the “Upload File” button. The selected file is transferred to the operator unit.

**Calling start page:**

- Click the “Start page” entry on the left of the navigation bar.

The new start page with the links to the call of the user-defined HTML pages is displayed.
3. **Loading user-defined HTML pages:**

   If you have not yet saved the necessary files on the MCC card or you would like to add other files, then you can also have this performed through the operator unit.

   - Login to the Web server an (if you are no longer logged on).
     - Name: **Administrator**
     - Password: **100**
   - Click the “File Browser” entry on the left of the navigation bar.
   - Click the “Storage Card MMC” folder.

   ![File Browser Screenshot](image)

   View of the directory with the “Simatic” and “HTML_Data” folders.

3. **Selecting user-defined HTML pages, CSS files and pictures:**

   **Note:**
   
   Note the used folder structure for the individual files. For this purpose, look at chapter 4.4.2 “Specifying storage location for the HTML pages”.

   - Open the folder in which the files are to be inserted by double-clicking it.
   - Create a new folder through the “MkDir” button. Before clicking the button, enter the name of the new folder in the text field next to it.
   - Click the “Browse…” button in the “Directory Operations” field and go to the folder in which files are located.
   - Mark the corresponding file
   - Click the “Upload File” button. The selected file is transferred to the MMC card.

   Repeat the previously described steps until all files have been transferred to the MMC card. Afterwards go back to the start page.

4. **Calling start page:**

   - Click the “Start page” entry on the left of the navigation bar.

   The start page is displayed.
6 Operating the Application

6.1 Operator unit, “Machine 01” screen 01

The facility picture of machine 01 is called through the “Screen 1” button.

Table 6-1

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td><strong>ON/OFF button:</strong> Through the “On” or “Off” button, the simulated drive can be switched on or off.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Set speed:</strong> by clicking the I/O field next to the “Set speed” button, a speed can be specified.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Actual speed:</strong> The output field next to the “Actual speed” text field outputs the actual speed.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Machine screen:</strong> The machine screen shows the current “operating mode”.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- gray: drive in standstill</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- green: drive is running</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- yellow: drive off and speed &lt;&gt; 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The values for the “Set speed” and the “Actual speed” are also output through the “Facility_01.html” HTML page. The “Set speed” can be specified through the “Facility_0x.html” HTML page.</td>
<td></td>
</tr>
</tbody>
</table>

I/O fields:

In the bottom part of the screen are the various I/O fields. They serve as additional input/output options of values. The last “String” field can be displayed or hidden through the “Facility_0x.html” HTML page.

**Note:** all values are output through the “Facility_01.html” HTML page. Additionally, “Facility_0x.html” values can be specified through the HTML page.
6.2 Operator unit, “Machine 02” screen 2

The facility picture of machine 02 is called through the “Screen 2” button.

Table 6-2

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
</table>
| 1.  | **ON/OFF button:** Through the “On” or “Off” button, the simulated drive can be switched on or off. **Set speed:** by clicking the I/O field next to the “Set speed” button, a speed can be specified. **Actual speed:** The output field next to the “Actual speed” text field outputs the actual speed. **Machine screen:** the machine screen shows the current “operating mode”.  
  - gray: drive in standstill  
  - green: drive is running  
  - yellow: drive is off and speed <> 0 **Note:** the values for the “Set speed” and the “Actual speed” are output through the “Facility_02.html” HTML page. | ![Diagram](image_url) |

**I/O fields:**

In the bottom part of the screen are the various I/O fields. They serve as additional input/output options of values.

**Note:** all values are output through the “Facility_02.html” HTML page.
### 6.3 Operator unit, “Service Site” screen 3

“Screen 3” can only be called through the “Facility_0x.html” HTML page.

#### Table 6-3

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>User Administration:</strong></td>
</tr>
<tr>
<td></td>
<td>The configured “user display” was only</td>
</tr>
<tr>
<td></td>
<td>configured as application example.</td>
</tr>
<tr>
<td></td>
<td>In the configuration itself no user rights</td>
</tr>
<tr>
<td></td>
<td>have been assigned.</td>
</tr>
</tbody>
</table>

#### 6.4 “start page” HTML page

#### Table 6-4

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The individual user-defined HTML pages</td>
</tr>
<tr>
<td></td>
<td>are called through the revised start page.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Facility 01</strong></td>
</tr>
<tr>
<td></td>
<td>• <strong>Facility 02</strong></td>
</tr>
<tr>
<td></td>
<td>• <strong>Facility 0x</strong></td>
</tr>
</tbody>
</table>

Before you make your selection, you have to login to the Web server.

- **Name:**
  - name you have assigned under the Web authorization on the operator unit [Link].
  - Standard “Administrator”.

- **Password:**
  - password you have assigned under the Web authorization on the operator unit [Link].
  - Standard “100”.

**Note:**
apart from the “start page” no other “Standard HTML pages” of the operator unit have been changed.
6.5 “Facility 01” HTML page

The facility data for machine 01 is output through the “Facility 01” page. The data is output through an “invisible” table.

The facility cannot be operated through this page.

Table 6-5

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screenshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>“&lt; back” button: You get back to the start page of the operator unit through the “&lt; back” button.</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>I/O fields: The facility data is output through the I/O fields. The values correspond 1:1 to the values, as they are displayed at the operator unit.</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>“Refresh” button: the displayed values are updated through the “Refresh” button.</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> values cannot be transferred to the operator unit through the page.</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
</tbody>
</table>
6 Operating the Application

6.6 “Facility 02” HTML page

The facility data for machine 02 is output through the “Facility 02” page. The data is output through a "visible" table.

The facility cannot be operated through this page.

Table 6-6

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screenshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>“&lt; back” button: You get back to the start page of the operator unit through the “&lt; back” button.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>I/O fields: The facility data is output through the I/O fields.</td>
<td><img src="http://example.com/image.png" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>The values correspond 1:1 to the values, as they are displayed at the operator unit.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>“Refresh” button: the displayed values are updated through the “Refresh” button.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Note: values cannot be transferred to the operator unit through the page.</td>
<td></td>
</tr>
</tbody>
</table>
6.7 “Facility 0x” HTML page

The facility data for machine 01 is output through the “Facility 0x” page. In contrast to the “Facility 01” HTML page, you can also change facility data through this page. The facility cannot be operated through this page.

Table 6-7

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td><strong>Section 1:</strong>&lt;br&gt;“&lt;- back” button:&lt;br&gt;You get back to the start page of the operator unit through the “&lt;- back” button.</td>
<td>![Screenshot of the “&lt;- back” button]</td>
</tr>
<tr>
<td>2.</td>
<td><strong>Section 2:</strong>&lt;br&gt;I/O fields:&lt;br&gt;the facility data is output through the I/O fields.&lt;br&gt;In addition, values can be specified and transferred to the operator panel through these IO fields.&lt;br&gt;For this purpose, specify a value and complete the entry through the “ENTER button”.&lt;br&gt;The values correspond 1:1 to the values, as they are displayed at the operator unit.&lt;br&gt;“Refresh” button:&lt;br&gt;the displayed values are updated through the “Refresh” button.</td>
<td>![Screenshot of the “Refresh” button]</td>
</tr>
</tbody>
</table>
6 Operating the Application

6.7 “Facility 0x” HTML page

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.</td>
<td><strong>Section 3:</strong> This section serves to transfer an arbitrary message to the user at runtime. When the text field is displayed on the operator unit there is, for example, the option that the user sends you a message through the operator unit. <strong>Application:</strong> communication with the operator without additional tools such as cell phone or telephone. <strong>Text field:</strong> you can specify an arbitrary text with up to a max. of 55 characters through the text field (limited through the specified I/O length on the operator unit). <strong>“Fade In Text”/“Hidden Text” button:</strong> through the button you can have a text row in screen 1 of the operator unit displayed or hidden. The content of the text is specified through the previously written text field. <strong>I/O fields:</strong> The “Text line active:” I/O field shows whether the text row is visible on the panel.</td>
<td><img src="image" alt="Show text message" /></td>
</tr>
</tbody>
</table>
### 4. Section 4:
this section serves, for example, to call a service site (user display) on the operator unit.

- **Application:**
  if, for example, the Admin password has been forgotten and it has to be renewed without retransferring the configuration.

**“Call Screen”/“Rest Call” button:**
the “Service Site” on the operator unit is called through the “Call Screen” button. the request for calling the screen is reset through the “Reset Call” button.

**I/O fields:**
the I/O field shows whether the function has already been enabled.

**Note:**
the function is only executed if there is a connection to the PLC. (The tag used has PLC connection, Link).

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.</td>
<td><strong>Section 4:</strong> this section serves, for example, to call a service site (user display) on the operator unit. <strong>Application:</strong> if, for example, the Admin password has been forgotten and it has to be renewed without retransferring the configuration. <strong>“Call Screen”/“Rest Call” button:</strong> the “Service Site” on the operator unit is called through the “Call Screen” button. the request for calling the screen is reset through the “Reset Call” button. <strong>I/O fields:</strong> the I/O field shows whether the function has already been enabled. <strong>Note:</strong> the function is only executed if there is a connection to the PLC. (The tag used has PLC connection, Link).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Activate Screen: Service site</strong></td>
</tr>
<tr>
<td>Activate screen:</td>
</tr>
<tr>
<td>Reset screen request:</td>
</tr>
<tr>
<td>Service site active:</td>
</tr>
</tbody>
</table>
7 Notes, Tips and Tricks

7.1 Adjusting WebContent file

The standard HTML pages for the operator units are located in the “WebContent” file. The content of this file is always transferred when the configuration is transferred to the operator unit and the “Sm@rtService: HTML pages” option is enabled.

If your revised start page is saved on the operator unit and you transfer your configuration afterwards, the revised start side is overwritten with the start page from the “WebContent” file. Therefore you have to transfer the revised start page again to the operator unit.

Remedy:

- you can insert the revised start page directly in the “WebContent” file.

Table 7-1

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screenshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Storage path of the “WebContent” file:</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>The “WebContent” zip file with the standard HTML pages for the operator units can be found in the installation directory of WinCC flexible.</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>C:\Program Files\Siemens\SIMATIC WinCC flexible\WinCC flexible 2008\Transfer\1.3\operator unit\WebContent.zip</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>You can also use the Windows search function. (Search word: WebContent).</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>Note:</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>create a safety copy of the original “WebContent” file before you process it.</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td>2.</td>
<td>Copying “WebContent” file:</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>copy the “WebContent” file in a separate folder.</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td>3.</td>
<td>Unzipping the “WebContent” file:</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>unzip the “WebContent.zip” file in a separate folder.</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>the screen on the right shows the file content for the MP 277 Touch.</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>Note:</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>do not change the file structure.</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td>4.</td>
<td>Inserting processed start page:</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>replace the existing original start page with the revised “start.html” start page.</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td>5.</td>
<td>Zipping the “WebContent” file to a zip file again:</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
<tr>
<td></td>
<td>from the files, create a “WebContent.zip” file again.</td>
<td><img src="image" alt="Screenshot" /></td>
</tr>
</tbody>
</table>
## Replacing the “WebContent” file:

replace the existing original "WebContent" file with the revised "WebContent" file.

This completes the expansions. With the next transfer of the configuration, the revised start page is also transferred to the operator unit.
7.2 Browser address line and updating of HTML pages

HTML pages are usually “static” pages this means the data within the HTML pages is only updated when the page is called.

To avoid having to call the page new every time you want to update the data, there is a “Refresh” button on the created user-defined HTML pages.

What do I have to observe?

If you specify a value through the “Facility 0x” HTML page and click the ENTER button, the specified value is transferred to the operator unit.

You can read this “instruction” in the command line of the “Web” browser.

In the table below, the “Facility 0x” HTML page is called.

Table 7-2

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Screens</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>View of the command line after calling the HTML page through the start page. Look at the text in the “command line”.</td>
<td><img src="http://172.16.34.170/StorageCard/PMC/HTML_Docs/Facility_0x.html" alt="Facility 0x - Windows Internet Explorer" /></td>
</tr>
<tr>
<td>2.</td>
<td>View of the HTML page according to the “Set-Speed of 3,500” and confirming the entry with the “ENTER button”. Look at the text in the “command line”.</td>
<td><img src="http://172.16.34.170/StorageCard/PMC/HTML_Docs/Facility_0x.html" alt="Facility 0x - Windows Internet Explorer" /></td>
</tr>
</tbody>
</table>

Description:

apart from the instruction for the call of the HTML page, the instruction for increasing the set speed (Set-Speed=3500) is also listed.

This “instruction” remains in the address line until a new instruction follows.

What does this mean for the updating of the HTML page?

If the button for updating the HTML page is clicked, then the instruction in the address line is executed again. This may have the effect, for example, that an accidental specification of the set speed is performed.

Note these general properties for HTML pages if you update the data on a HTML page.
8    Links & Literature

Internet links

This list is by no means complete and only presents a selection of suitable information. Links, especially those not referring to SIEMENS pages may change or be no longer available.

Table 8-1

<table>
<thead>
<tr>
<th>Topic</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1| Reference to this entry</td>
<td><a href="http://support.automation.siemens.com/WW/view/en/EntryID">http://support.automation.siemens.com/WW/view/en/EntryID</a></td>
</tr>
<tr>
<td>2| Siemens I IA/DT Customer</td>
<td><a href="http://support.automation.siemens.com">http://support.automation.siemens.com</a></td>
</tr>
<tr>
<td>3| Learning HTML</td>
<td>Learning HTML and creating your own homepage</td>
</tr>
<tr>
<td>3| Learning HTML</td>
<td><a href="http://www.html-seminar.de/index.htm">http://www.html-seminar.de/index.htm</a></td>
</tr>
<tr>
<td>4| Learning HTML</td>
<td>Creating your own HTML files</td>
</tr>
<tr>
<td>4| Learning HTML</td>
<td><a href="http://de.selfhtml.org/index.htm">http://de.selfhtml.org/index.htm</a></td>
</tr>
<tr>
<td>5| HTML Tutorial</td>
<td>HTML Code Tutorial</td>
</tr>
<tr>
<td>5| HTML Tutorial</td>
<td><a href="http://www.htmlcodetutorial.com/">http://www.htmlcodetutorial.com/</a></td>
</tr>
</tbody>
</table>

9    History

Table 9-1

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1.0</td>
<td>16.03.2011</td>
<td>First issue</td>
</tr>
</tbody>
</table>