LOVATO ELECTRIC S.P.A.

24020 GORLE (BERGAMO) ITALIA
VIA DON E. MAZZA, 12
TEL. 035 4282111
E-mail info@LovatoElectric.com
Web www.LovatoElectric.com

I578 GB 06 20

GB **PROGRAMMING SOFTWARE FOR HMI LHR SERIES**

Instructions manual

LRH SW

CE

Contents

1 Getting started

LRH SW is a software application designed to create graphical HMI pages. LRH SW has a drag-and-drop interface that makes it easy to create complex pages. Many of the features found in common Windows applications are also available in LRH SW.

This document is divided into chapters that describe the key functions of LRH SW and explain how to use them. Each chapter is presented in a standalone manner, allowing you to jump from chapter to chapter, depending on the task at hand.

Assumptions

We assume that readers have a basic understanding of computers, Microsoft Windows, and the specific network environment where the application will run.

Installing the application

LRH SW installation contains:

- LRH SW: an application for designing custom HMI projects in a user-friendly manner, along with a variety of objects in its built-in library, the Widget Gallery.
- LRH SW Client: a light-weight application that can be used on Windows computers to remotely view and manage a project running on an HMI device.
- LRH SW HMI Runtime: a standalone application that runs on the HMI devices. The HMI Runtime is installed via LRH SW.

LRH SW system requirements

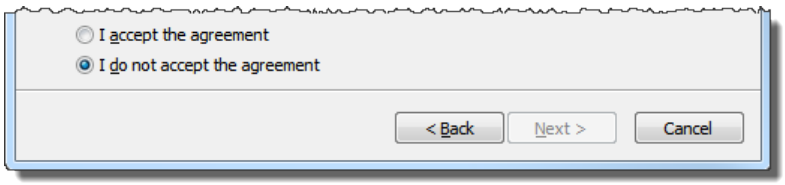
LRH SW has the following system requirements:

Operating System	Windows Embedded Standard (WES 2009) Windows Server 2003 Windows Vista Business/Ultimate Windows 7 Professional Windows Embedded Standard 7 Windows 8 Windows 10
Storage	500 MB Minimum
RAM	512 MB
Other	One Ethernet connection

Installation procedure

To install LRH SW:

1. Run LRH SW setup and click **Next**.
2. Read the LRH SW Software License and accept the agreement.



3. Follow the instructions on the screen. The default location for the c software is *C:\Program Files\Lovato Electric\LRH SW*, change path if needed.
4. If the Select Components step is available, select the components you want to install.
5. Select the **Create a desktop icon** option to add a LRH SW icon on your desktop. A LRH SW group is automatically added to the **Start** menu by the installation procedure.

6. To run the application click the desktop icon or choose **Start** > **All programs** > **LRH SW**.

Trial version

LRH SW is available with a friendly 30 days free trial policy. 30 days after installation a registration form is displayed to enter a license activation key.

Note: Trial version is not supported on virtual machines, only valid licenses can be used.

Licensing

To register the software before the trial period expires, go to **Help** > **Register**.

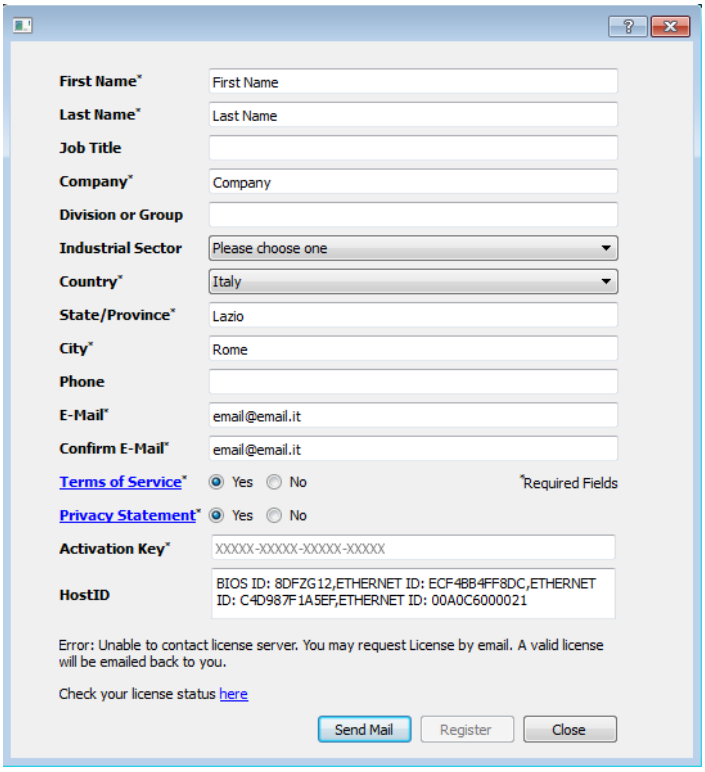
Note: The registration process requires an Internet connection. Ports TCP 80 and 443 are used for activation.

During registration, a license file is downloaded from the licensing server to the computer. License files are saved in following folders depending on OS:

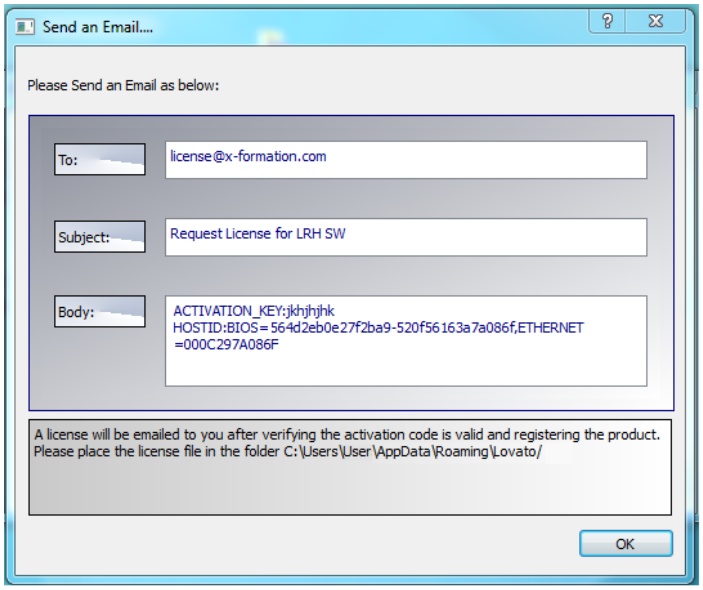
%appdata%\Lovato

Licenses are locked to the **BIOS ID** or to the **Windows product ID** of the computer where the software has been installed.

If LRH SW is not able to reach the licensing server (for example, no Internet connection is available), a button is displayed to activate the license via email.



Pressing the “Send Mail” button the LRH SW will display this form:



This email can be send in a second moment when internet connection will available. You may also activate the licenses and download the licenses file from the web site https://licenses.x-formation.com/licenses. reporting the same data contained in the “Body” of upper form.

Verifying license status

To check the status of your license:

1. Go to: *https://license.x-formation.com/licenses*
2. Enter your activation key and click the **Log In** button.

Installing multiple versions of LRH SW

You may install different instances of LRH SW on the same computer. Each installation has its own settings and can be uninstalled individually.

Three installation scenarios are possible:

Installation scenario	Results
First installation of LRH SW in the system	Software is installed in the specified destination folder
System with only one instance of LRH SW already installed	Current version can be replaced or maintained.
System with multiple instances of LRH SW already installed	Last version installed can be replaced or maintained.

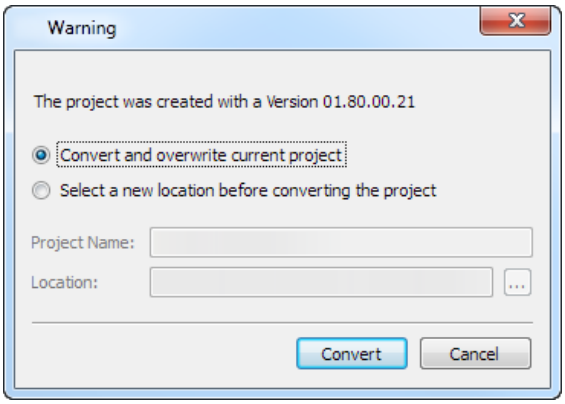
If you try to install a second instance of an already installed version of LRH SW, a warning message is displayed.

Multiple LRH SW installations share a common workspace folder, each sub-folder includes the version number, for example *C:\Program Files\Lovato Electric\LRH SW 2.5*. Each installed version has its ID and can therefore be removed individually.

Each installation is listed separately in the Windows **Start** menu.

Opening older projects

When opening a LRH SW project (.jpr file) created with an older version of the software LRH SW asks to convert the project to the current version:



Option	Description
Convert and overwrite current project	The project is converted without a backup copy of the original version
Select a new location before converting the project	The project is copied inside the specified folder and then converted.

WARNING: Do not edit projects with a version of LRH SW older than the version used to create them. This will damage the project and may cause runtime instability.

Multilanguage for LRH SW

LRH SW is available in multiple languages. All languages are installed by default as part of LRH SW.

The default language is English. To change it go to **Help** > **Change Language**.

Crash reports

A crash report dialog appears whenever LRH SW freezes or crashs.

Important: Always save crash report files since they may contain useful information for technical support.

Note: Crash reports are unavailable in Windows XP.

2 Runtime

LRH SW HMI Runtime is designed to support different platforms and different operating systems.

HMI device basic settings

HMI devices are delivered from factory without Runtime. If no Runtime is installed on the device, see "The Runtime loader" on page 97 for details.

Runtime modes

The LRH SW HMI Runtime is composed of two logic units:

- **Server**: runs communication protocols, collects data, monitors alarms, drives trend buffer sampling.
- **Client**: displays data collected by server.

The server unit is responsible for handling the HMI services such as the communication protocols, performing data acquisition, driving trend buffer sampling activities, monitoring alarms, and so on.

The client unit is the part which is responsible for the visualization process: use the data collected by the server to render it on the display as graphical information.

The server unit works in two operating modes:

- **Configuration mode**: server is idle (for example when no project is loaded on the device or some system files are missing).
- **Operation mode**: server is operating according to the settings defined by the system files and by the loaded application project.

Note: Data on client may be displayed even if no activity is running on the server.

Context menu options

On the HMI device press and hold on an empty area of the screen for a few seconds to display the context menu.

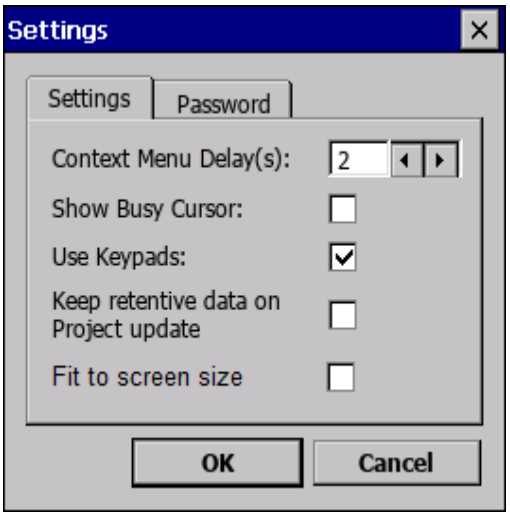
Zoom In/Out

Select view size at runtime

Pan Mode

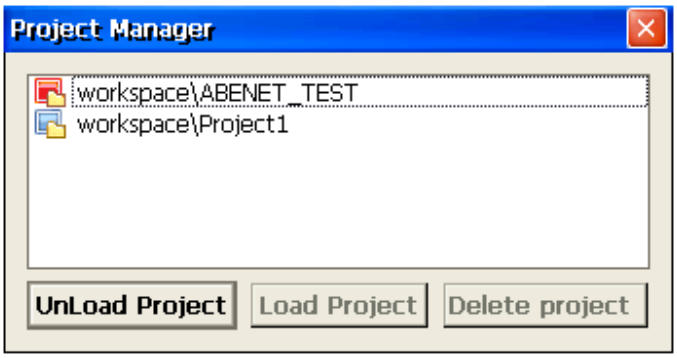
Enables/disables pan mode after a zoom in

Settings



Main parameters	Description
Context Menu Delay (s)	Context menu activation delay. Range: 1–60 seconds.
Show Busy Cursor	Display an hourglass when the system is busy
Use keypads	Display keypads when user touches a data entry field. Set to **disable** when an external USB keyboard is connected to the device.
Keep retentive data on project update	Preserve the content of the retentive data at project download or update.
Fit to screen size	Adapts the view to the screen size
Password	Define password protected operations amongst the following: • Download Project/Runtime • Upload project • Board management (BSP Update) See "Protecting access to HMI devices" on page 527 for details.

Project Manager



This tool allows you to:

- unload the current project
- load another project
- delete a project.

When you load a new project, the current project is automatically unloaded. You must unload a project before you can delete it.

Update

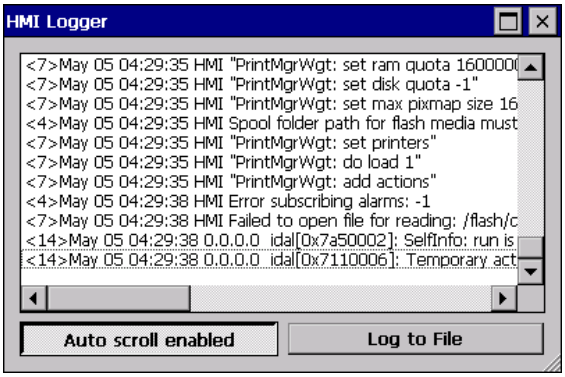
This function loads update packages from an external USB drive. See "Update package" on page 95for details.

Backup

You can create a backup copy of the Runtime and of the project.

Logging

This function displays a log of system operations.



Click **Log to file** to save data: a logger.txt file is saved to the *...\var\log* folder.

This file can be retrieved using an FTP Client and forwarded to technical support.

Note: Once enabled, logging is maintained after power cycles and must be manually disabled.

Show log at boot

This function enables the logger at start up. If the **Log to file** option has been enabled, log files are saved from startup.

Logout

Logs off the current user.

Show system settings

Allow the HMI settings and the management of system components. See "System Settings" on page 543 for details.

Developer tools

Utility functions for debugging at runtime. It is visible only if enabled in the Project Properties (see "Developer tools" on page 76 for details)

About

This function shows information about the Runtime version.

WARNING: Context Menu action has no effect if executed from a dialog page.

Built-in SNTP service

The HMI device features an integrated SNTP that synchronizes the internal real-time clock panel whenever the predefined server is available.

Use HMI device "System Settings" on page 543to configure the service.

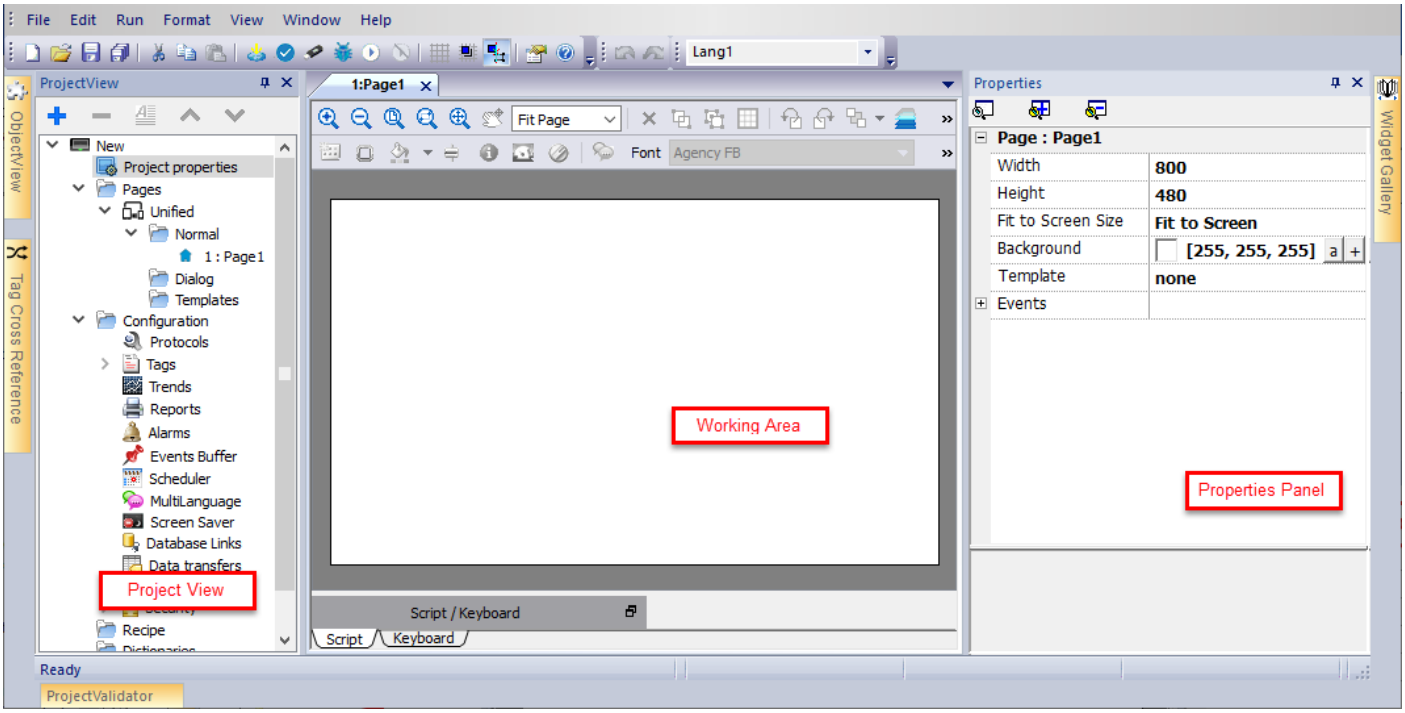
3 My first project

This section describes how to create a simple LRH SW project.

The workspace

Workspace areas

LRH SW workspace is divided into the following main areas:



Area	Description
Project View	Project elements in hierarchical project tree.
Object View	Tree view of widgets organized by page.
Working Area	Space where pages are edited. Tabs at the top of the area show all open pages.
Properties	Properties of selected object.
Widget Gallery	Library of graphic objects and symbols.
Tag cross reference	List of locations where a given tag is referenced.
Project Validator	Area used from the Project Validator to list warning messages related to the project

Note: The workspace layout can be changed at any time, changes are saved and maintained through working sessions.

Resetting the workspace layout

To restore the default layout, use the **File > Reset and Restart** function.

Creating a project

Path: ***File*** *>* ***New Project***

1. In the **Project Wizard** dialog enter a name for the project and the storage location.
2. Click **Next**: the HMI device selection dialog is displayed.
3. Choose one device from the list of the available models.
4. Choose device orientation.
5. Choose the project template to create.
5. Click **Finish** to complete the Wizard.

Portrait rotation exceptions

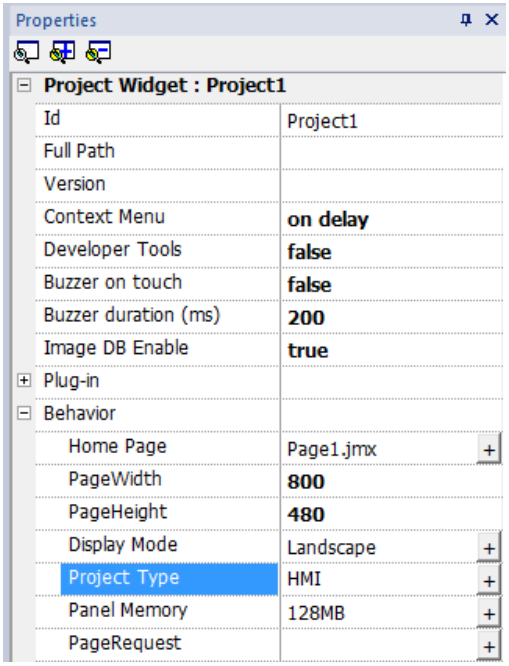
The following elements are not rotated in portrait mode.

Element	Description
Operating system dialogs	System settings and system dialog
ContextMenu and related dialogs	Project Manager, About, Settings, Logging, Backup
Video	IPCamera, MediaPlayer
JavaScript	Alert and Print function
Dialog pages	“Title” of dialog pages
Scheduler	Dialogs for data entry
Macro	ShowMessage, LunchApplication, LunchBrowser
External applications	PDF Reader

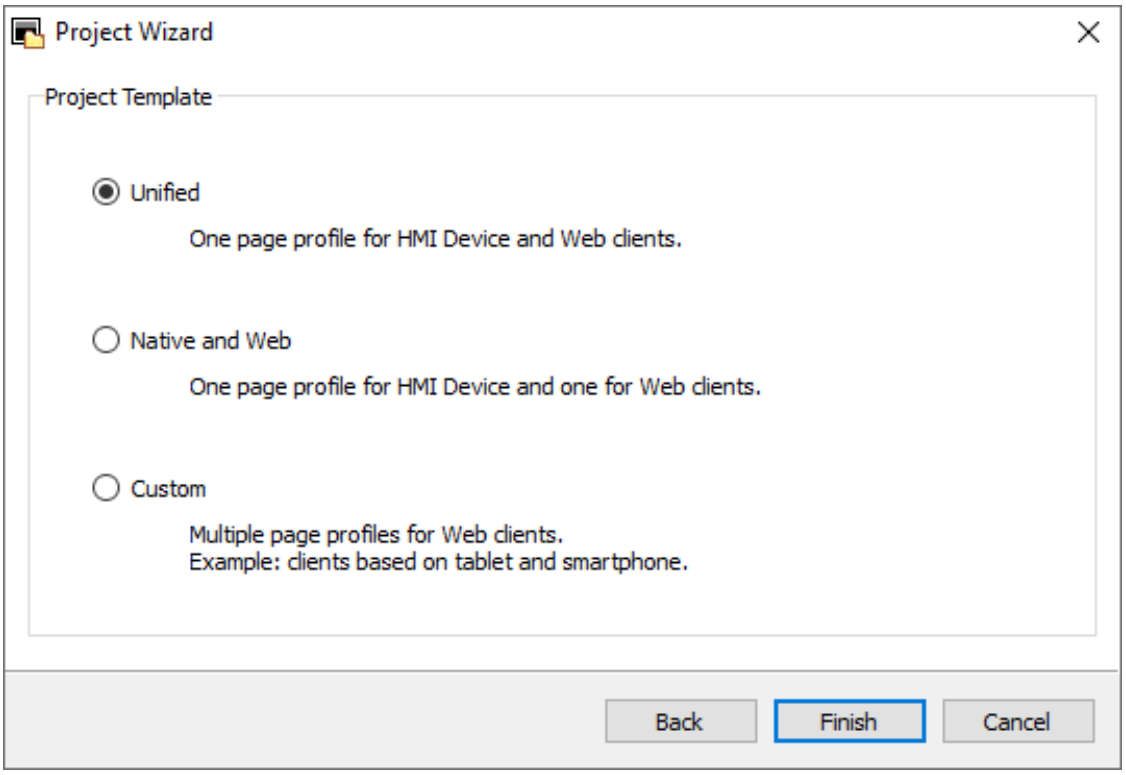
HMI devices based on Linux platform can be rotated from the BSP (see "*Displays*" tab from the "System Settings" on page 543"Linux Devices" on page 544page) without these limitations.

Changing the device model

Once you have developed your project you can still change the device model, from the Project Properties pane. This will not resize the widgets, but will relocate them on the screen. A warning will be displayed if some objects cannot be relocated.



Project Template



The "project template" proposes and then creates, the most common folder structures for the project's pages. Later, you can always modify the structure at any time.

The propose structure are:

Element	Description
Unified	Create a unique folder that will contain all the pages of your project. The same pages could be used on HMI Device, on remote clients and on Web Clients. Use this choice if you want to have the same pages on all platforms.
Native and Web	Create two folders, one to contain the pages of your project that will be used on the HMI device and another one for the pages that will be used on Web clients. Use this choice if you want to have different pages on Web clients.
Custom	Give the possibility to create different folders to contain the pages to use on HMI device, Web client, Table client, and Smartphone client.

Copying, moving, renaming a project

LRH SW projects folder contain all the files of the project: to move, copy or backup a project, move or copy the project folder to the desired location.

To rename a project use the **File > Save Project As** function: this operation might take a few minutes.

WARNING: Do not rename the project folders manually.

Designing a page

Path: ***ProjectView*** > ***Pages***

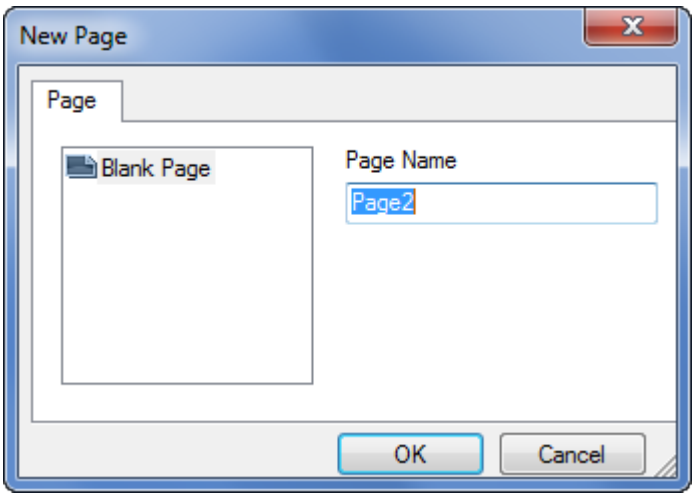
When a project is created, the first page is automatically added and shown in the **Page Editor**.

Adding objects to a page

Drag and drop objects from **Widget Gallery** to the page.

Adding a page

1. Right click the **Pages** node from the project tree and select **Insert new page**.
2. Type a name for the new page.



Importing a page

When importing a page LRH SW will import the page layout and the page widgets without importing the actions and data links attached to widgets. You can choose between two different behavior:

- importing only the pages and the widgets: in this case all actions and data link have to be defined
- importing pages with references to actions and data links: used tags must be present in the project for these elements to work properly

Note: Page import can only be performed between projects made using the same software version. Save the older project as the newer version, then try again.

1. Right click the **Pages** node from the project tree and select **Import page**.
2. Choose the page to be imported from the desired project then click **OK**: a warning message is displayed.
3. Click **Yes** to remove all the links to data and actions. Click **No** to maintain the reference to data links and actions. Tags need to be available in the new project.

Group of pages

You can group similar pages for easier maintenance. Grouping pages does not affect how pages appears at runtime. To create a group of pages:

1. In **ProjectView** right click **Pages** node and select **Create Group**: a new folder is added
2. To move a page to a group, right click a page and select **Groups** > *groupName.*

The Widget Gallery

Path: ***View> Toolbars and Docking Windows*** *>* ***Widget Gallery***

HMI objects required to build an application are available in the **Widget Gallery**. The gallery is divided into several categories, each containing a collection of widgets.

Adding a widget to a page

1. Select the widget from the **Widget Gallery**.
2. Drag and drop it on the page.

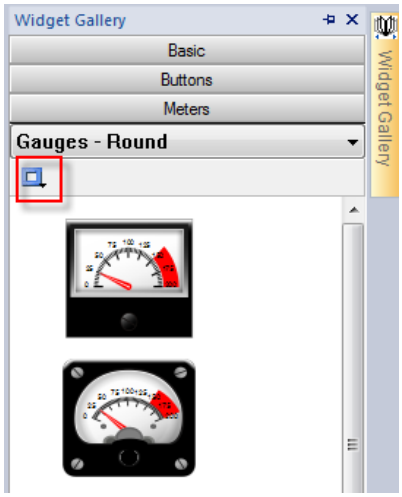
Changing the appearance of a widget

All widgets have properties (**Properties** pane) that can be changed, Some widgets are presented in various styles. You can click the buttons in each category to see available styles.

Example

To set the widget style for round gauges:

1. Click the style button to display the available styles for the widget.

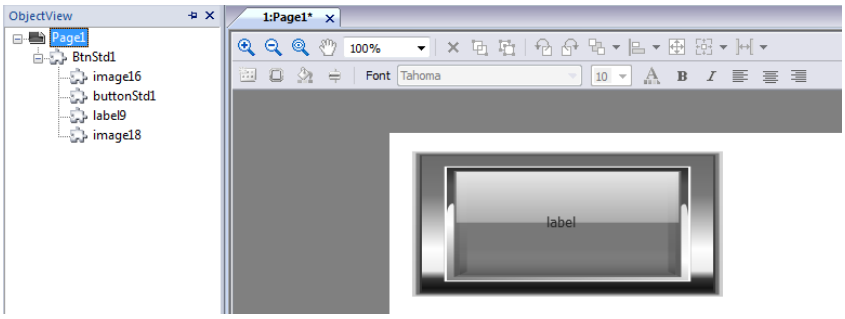


2. Select one of the available styles from the toolbar: depending on the selected widget, different options are available.

Complex widgets

Some widgets are composed of many sub widgets. For example, a button is a complex widget composed by a button widget and a label. The structure of widgets can be seen in the **ObjectView** when the widget is selected.

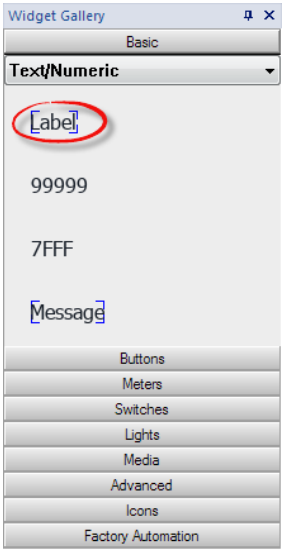
You can select a sub-widget, such as the label in a button, from the **ObjectView** and modify it without ungrouping the whole widget.



Label widget

The label widget gives the possibility to display text and tags values.

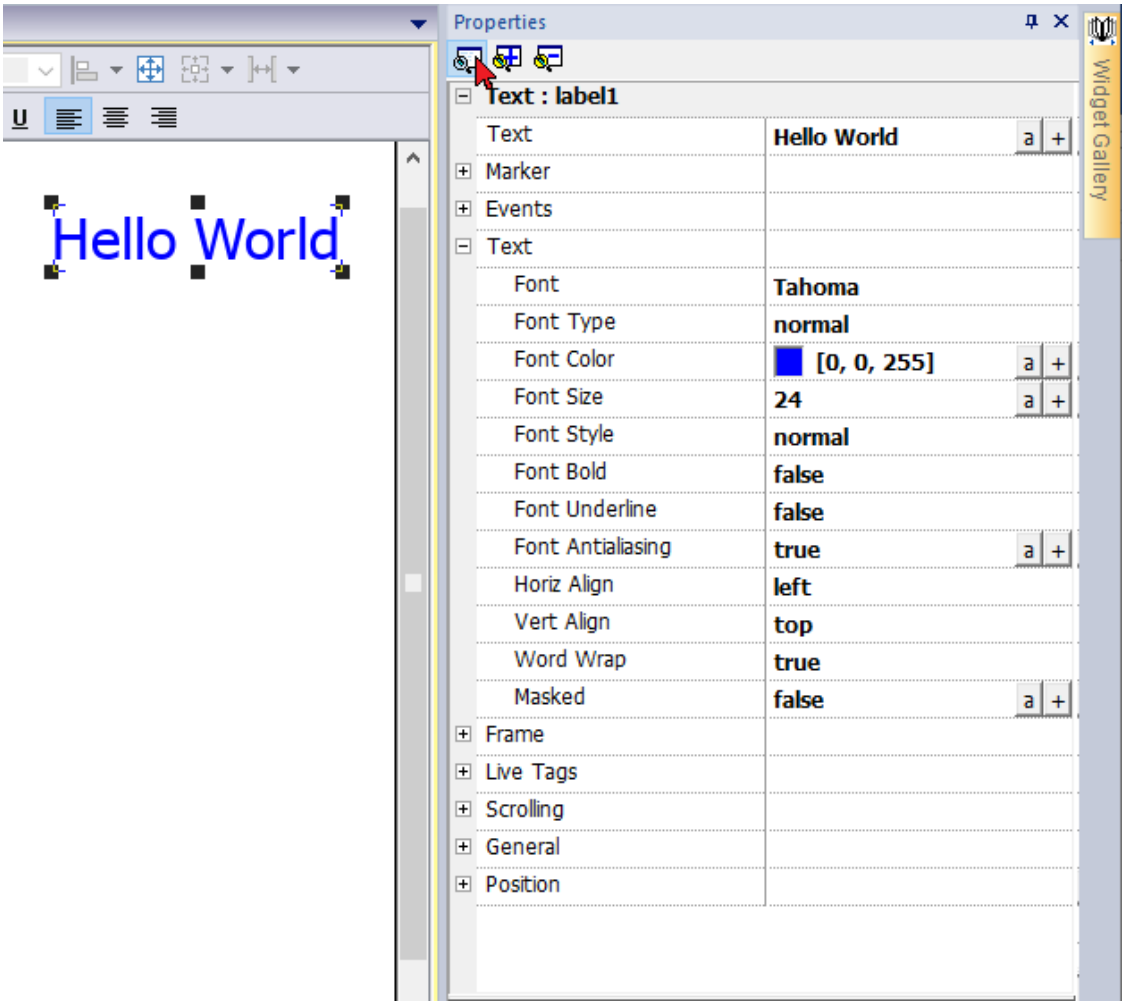
Path: ***View> Toolbars and Docking Windows*** *>* ***Widget Gallery***



Label properties

Drag and drop the widget inside the page and select the widget to open the properties dialog of the widget.

Note that some properties are visible only when the "Show Advanced Properties" button is selected.



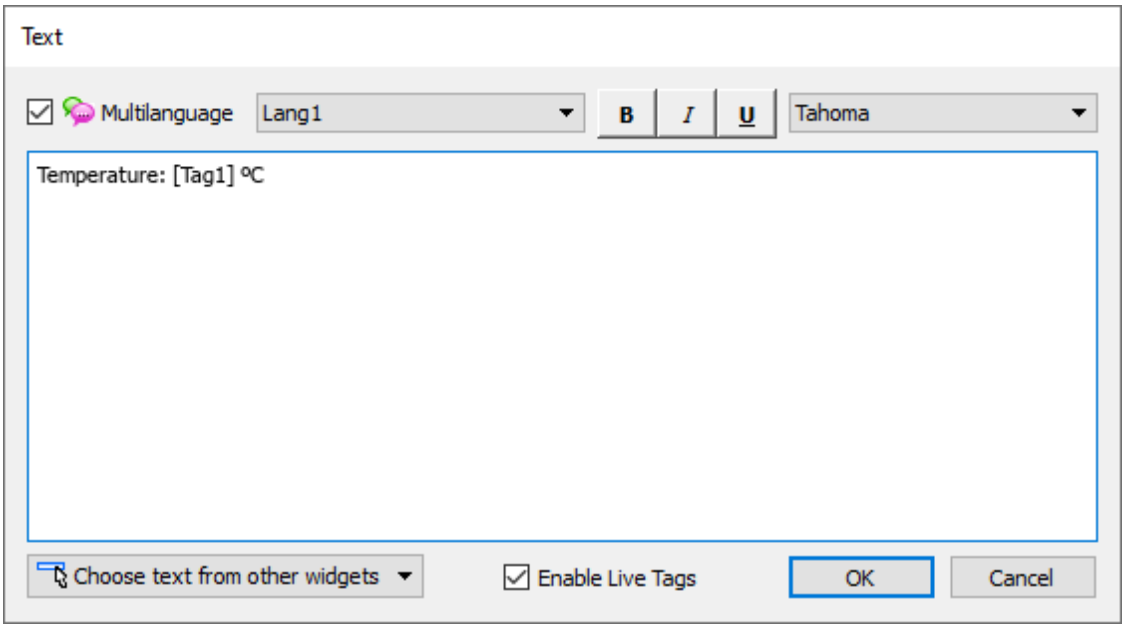
Property	Description
Text	The string to display. String can be static or retrieved from a TAG. See "Attaching widget to tags " on page 33
Marker	Enable a Marker around the widget (It is visible only inside LRH SW)
Events	Action that will be executed if widget contents change. See "Events" on page 49
Text (folder)	Text properties
Frame	Parameters to enable and configure a frame of the widget and/or a color for the background
Live Tags	Enable to use tags values inside the text message. See "Live Tags" on page 24 - Enable Live Tag: Enable live tags placeholder - Dynamic Subscription: When true, only the tags that are visible are retrieved from the communication protocol. When false, all tags are kept continuously updated even they are not visible.
Scrolling	Parameters to enable and configure the scrolling of the text message • None • Slow • Normal • Fast • Custom When the custom mode is selected, the below parameters can be defined: - Scroll type: For each timeout, the text is scrolled of a custom amount of characters or pixels. • Characters • Pixels - Scroll delay: The timeout after which label effectively start to scroll (mSec) - Scroll timer: The timeout which defines each scroll step (mSec) - Scroll dots or - Scroll characters: The number of pixels scrolled for each timer timeout or The number of character scrolled for each timer timeout - Scroll behavior: OnlyOnce Text scrolling stops after the first complete. LoopWithDelay Text scrolling restart after each complete cycle, waiting for the delay.

Property	Description	
		Loop Text scrolling never stop
General	General properties	
	- Id	Widget identifier. You can leave the default value or rename it to have a more appropriate name
	- Visible	When false the widget is not visible
	- Opacity	The opacity-level describes the transparency-level, where 1 is not transparent at all, 0.5 is 50% see-through, and 0 is completely transparent.
	- Blink	The text will blink
	- Lock	When True, the widget cannot be selected and moved from the LRH SW page editor
	- Static Optimization	Normal LRH SW will decide the best optimization mode to use
		Static LRH SW optimize the widget assuming it will never be modified by the runtime
		Dynamic LRH SW will not add additional optimizations
Position	The widget position on the display. See "Widget position on the display" on page 25	

Some properties have a couple of buttons:

- Enter edit mode:
  you can directly type the tag name to use
- Attach to tag:
  the dialog where select the tag to use will be opened

A double clicks over the label widget will open the edit dialog box where you can enter the text to display and set the main text properties.



Live Tags

"Enable Live Tags" is enabled, text between square brackets are managed as tags place holders and will be rendered, from the runtime, using the tag value.

For example, the text label "*Temperature: [Tag1] ºC*" will be rendered as:

Temperature: 18 ºC

where "18" is the value contained inside Tag1

Tags

- [TagName]
  The tag value is read and continuously updated

Use '\' before '[ ]' if you want to show the '[ ]' in the description string, for example: *\[Tag\[1\]\]* will display the string *"[Tag[1]]"*.

Use '\', even when the tag label contains square brackets. For example, to display the live tag value of tag “TAG]3” or “TAG[3]” use:

- TAG\]3 = **[**TAG\]3**]**
- TAG\[3\] = **[**TAG\[3\]**]**

Array Tags

To reference the entire array (all elements will be shown):

- [TagName]
  All array elements will be displayed using a comma separate list.
- [TagName[-1]]
  All array elements will be displayed using a comma separate list.

To reference an element of the array:

- [TagName.Index]
  Example: [MyARRAY.5] will display the sixth element of the MyARRAY
- [TagName[TagIndex]]
  Example: [TagIndex] will display the sixth element of the MyARRAY when TagIndex is 5

Data Formats

Placeholder characters can be used to control how to display the tag value (see "Custom Formats" on page 28)

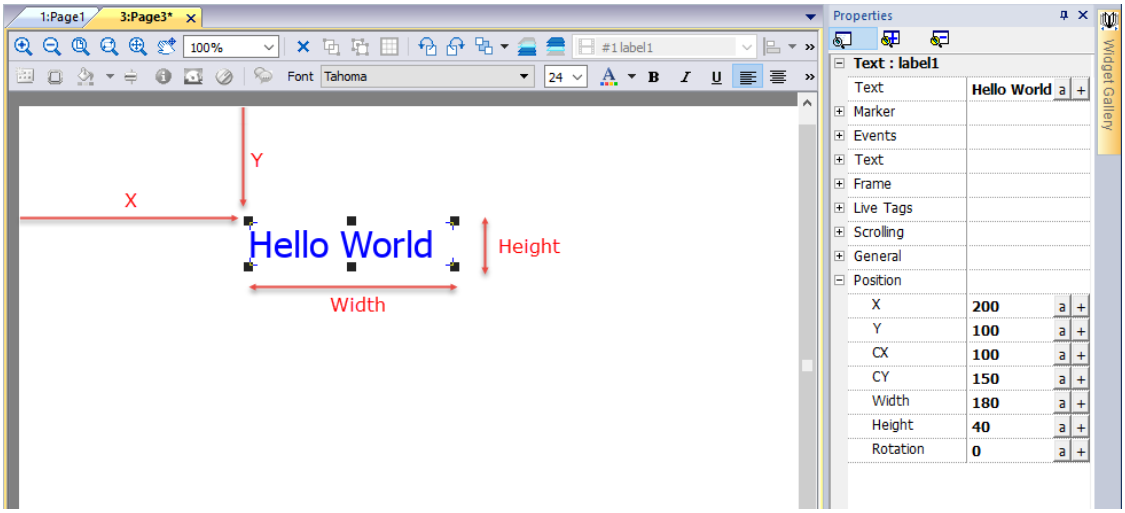
- [TagName|format("###")]

Example:

Live: [fCounter|format("#.00")] - Triggered: [!fCounter|format("#.00")]

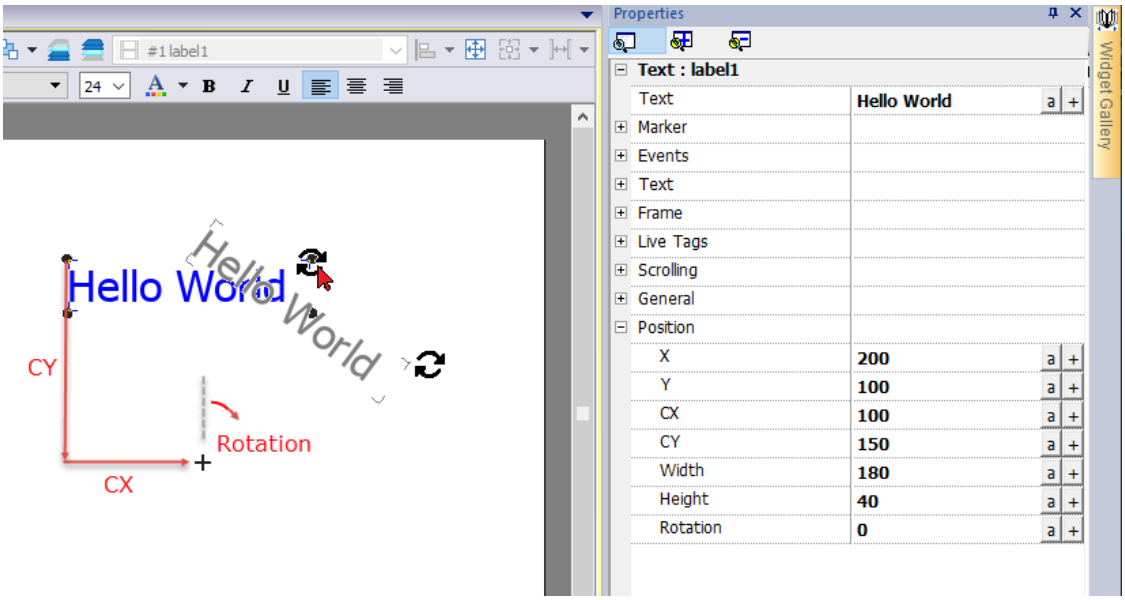
Note that by default, all tags are displayed as an integer. If you want to display a float number, you have to specify how to show the number adding the decimal digits.

Widget position on the display



Rotation

To rotate a widget, click two times the widget. After the first click, the markers will become square, after the second click will become circles. Now click the mouse over a circle marker and drag and drop to rotate the widget. The rotation center is identified by the CX and CY parameters.

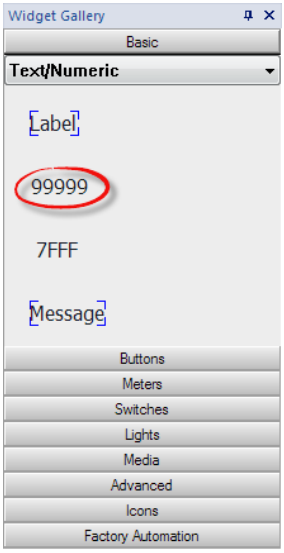


Note that all "Position" properties can be attached to tags and can be modified dynamically at runtime to move the widget.

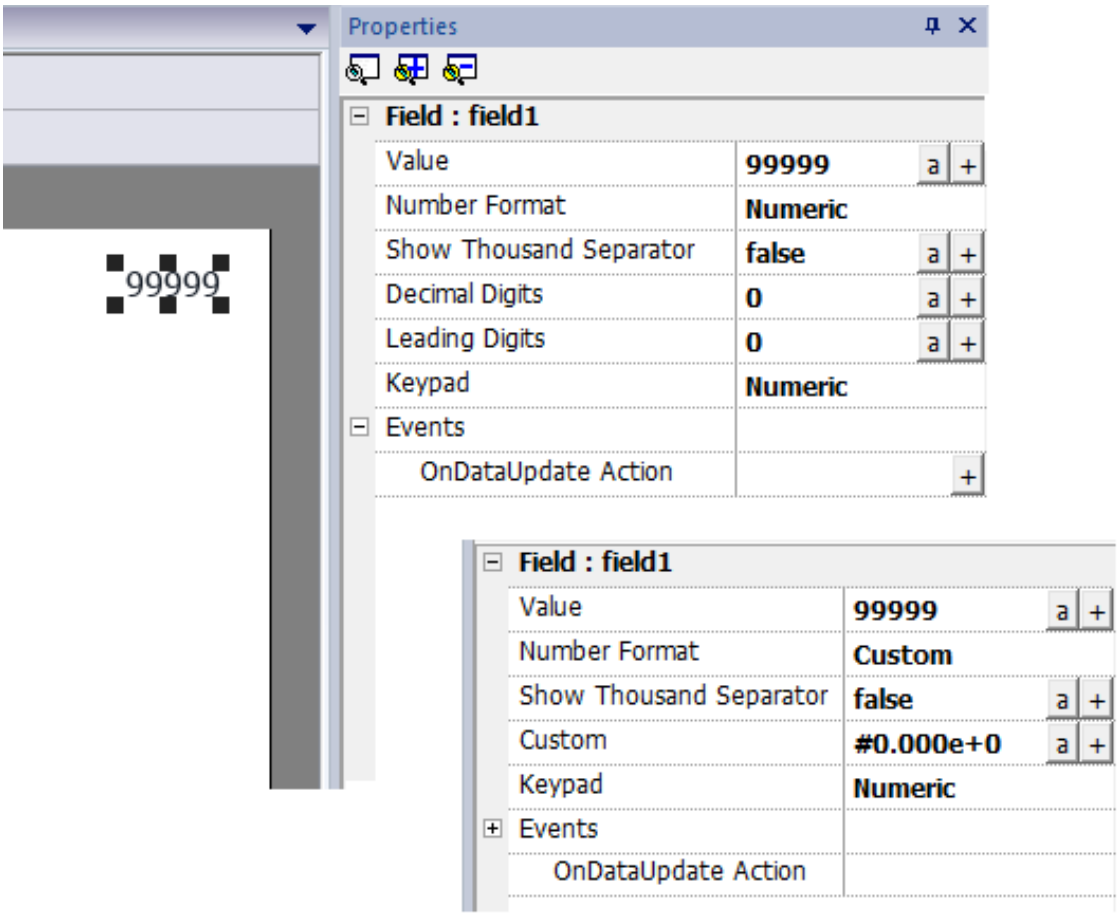
Data field widget

The most common widget is the data field widget that give the possibility to display value of tags. (See "Tag editor" on page 99).

*Path: **View> Toolbars and Docking Windows> Widget Gallery***



Field properties



Property	Description
Value	Tag that contain the information to display
Number Format	Display format • None No restrictions (system decide the format to use) • Numeric Numerical format. Decimal digits and Leading digits can be used to better define the number format • hex Hexadecimal format. Leading digits can be used to better define the number format • HEX The same of "hex" format but using uppercase • scientific Scientific format. Decimal digits can be used to better define the number format • SCIENTIFIC The same of "scientific" format but using uppercase

Property	Description
	• Custom Use the additional "Custom" parameter to better define the format to use (see the below table)
Show Thousand Separator	To show/hidden the thousands separator
Decimal Digits	Number of decimal digits to show (not available on all format types)
Leading Digits	Number of leading digits to show (not available on all format types)
Custom	String that define the number format to use (available only when selected Number Format is CUSTOM)
Keypad	Keypad type to pop up to edit the tab value • None • Alphabetic, Numeric, Etc. Pop up a predefined keypad or a user keypad (see "Keypads" on page 355) • Wheel Keypad will not be displayed . Wheel can be used to increment/decrement the numeric value • Macro Keypad will not be displayed . Keyboard macro can be used to enter keys (see "Keyboard actions" on page 175)
Events	
OnDataUpdate Action	Commands list to execute any time the tag value changes (See "Actions" on page 169 for the available commands)

The character used as thousand separators (point) and the character used as decimal separator (comma) can be modified from the global Project Property. See "Regional Settings" on page 86

Some properties have a couple of buttons:

Enter edit mode:
you can directly type the tag name to use

Attach to tag:
the dialog where select the tag to use will be opened

Custom Formats

In custom property, the allowed chars are “#” “.” “0” “h” “H” “e” “E”

Use the place holder characters to control the display of digits before and after the decimal place. Use the number sign (#) if you want to display only the significant digits in a number. This sign does not allow the display non-significant zeros. Use the numerical character for zero (0) if you want to display non-significant zeros when a number might have fewer digits than have been specified in the format code.

If a number has more digits to the left of the decimal point than there are placeholders in the format code, the extra digits are displayed. However, if a number has more digits to the right of the decimal point than there are placeholders in the format code, the number is rounded off to the same number of decimal places as there are placeholders.

Examples

To display	As	Place Holder
123	0123	000#
1500	5DC	#H
1500	5dc	#h
1500	05DC	000#H
123.456	123.46	#.##
123.456	000123.456000	00000#.000000
12,200,000	1.22E+07	#0.00E+00
12,200,000	12.2E+6	#0.0E+0

Message widget

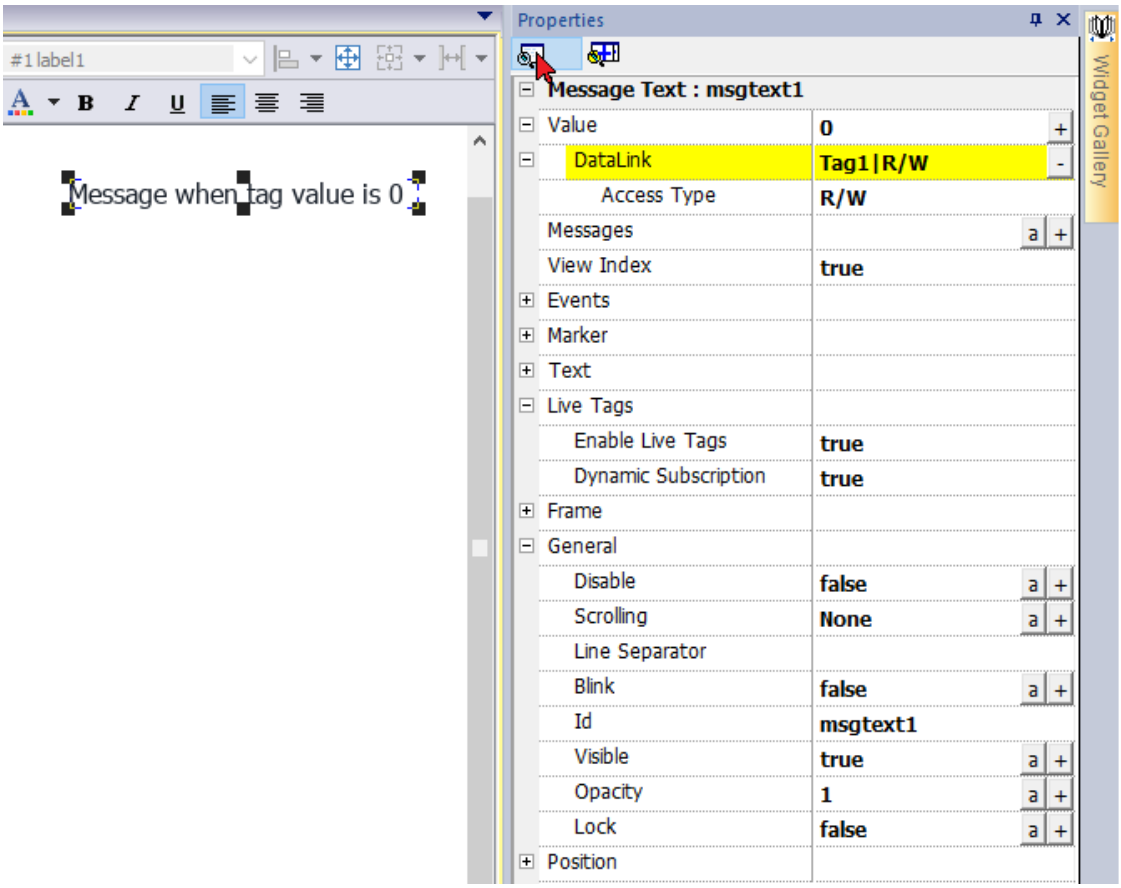
The message widget gives the possibility to display text a message indexed from a tag value.

Path: ***View> Toolbars and Docking Windows*** *>* ***Widget Gallery***

Message properties

Drag and drop the widget inside the page and select the widget to open the properties dialog of the widget.

Note that some properties are visible only when the "Show Advanced Properties" button is selected.



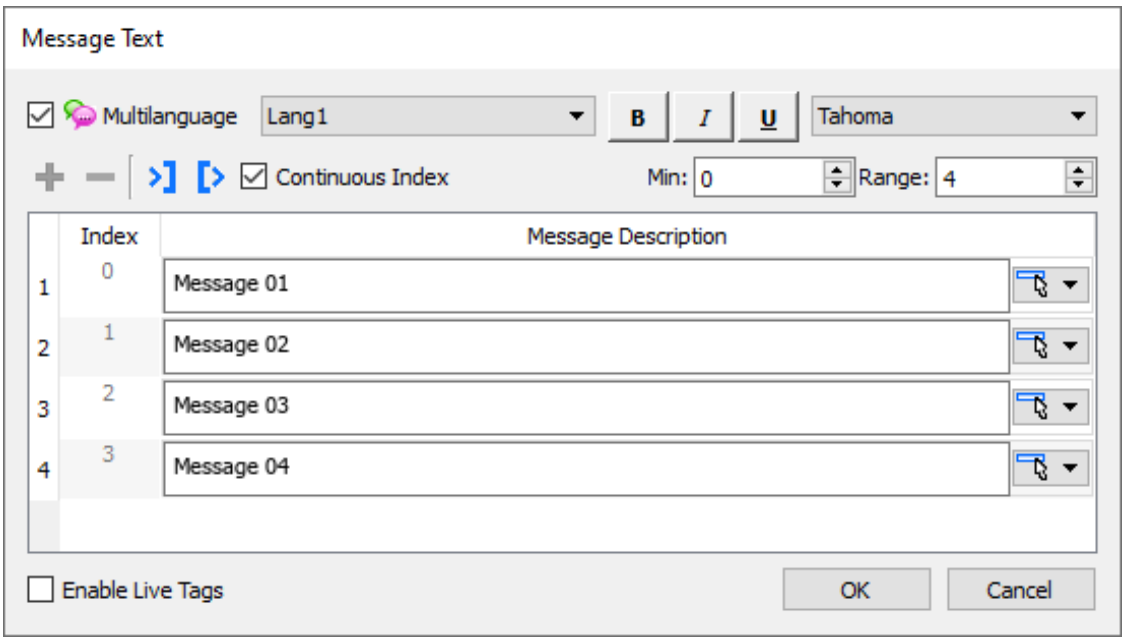
Property	Description
Value	The tag name to used to dynamically select the message to display. See "Attaching widget to tags " on page 33 When the "Access Type" is R/W, the value of the attached tag can be changed by clicking on the message. As a result, the message will be updated to be aligned with the new index value.
Messages	The list of messages to display. Click **+** to open a dialog where you enter messages and the associated index
View Index	If true, when the widget is editable (Access Type = R/W) the selection dialog will also show the index number associated with each message.
Events	Action that will be executed if widget contents change. See "Events" on page 49
Marker	Enable a Marker around the widget (It is visible only inside LRH SW)
Text	Text properties (font, color, size, etc.)

Property	Description
Live Tags	Enable to use tags values inside the text message. See "Message widget" on page 29 - Enable Live Tag: Enable live tags placeholder - Dynamic Subscription: When true, only the tags that are visible are retrieved from the communication protocol. When false, all tags are kept continuously updated even they are not visible.
Frame	Parameters to enable and configure a frame of the widget and/or a color for the background
General	General properties - Disable: Disable user inputs on the widget Attaching a tag at this property is possible to enable/disable the possibility to modify the value at runtime - Scrolling: Parameters to enable and configure the scrolling of the text message When enabled, all messages are linked together and displayed in scrolling mode. - Line Separator: Characters to insert between messages when shown in scrolling mode - Blink: The text will blink - Id: Widget identifier. You can leave the default value or rename it to have a more appropriate name - Visible: When false the widget is not visible - Opacity: The opacity-level describes the transparency-level, where 1 is not transparent at all, 0.5 is 50% see-through, and 0 is completely transparent. - Lock: When True, the widget cannot be selected and moved from the LRH SW page editor
Position	The widget position on the display. See "Widget position on the display" on page 25

Some properties have a couple of buttons:

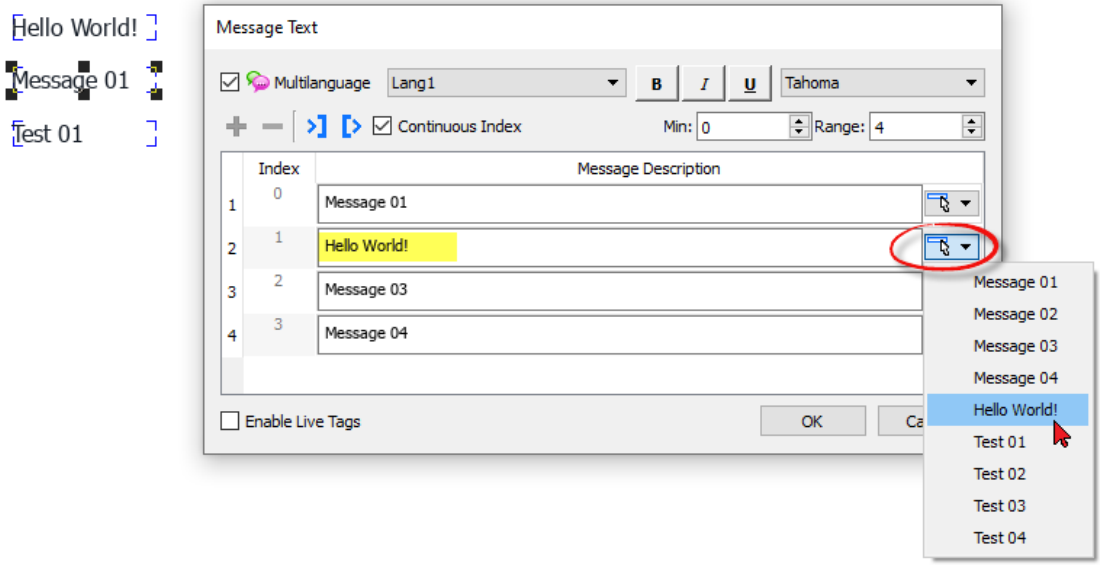
- Enter edit mode:
  you can directly type the tag name to use
- Attach to tag:
  the dialog where select the tag to use will be opened

A double clicks over the label widget will open the edit dialog box where you can enter the text to display and set the main text properties.



Pick Text Button

The "Pick Text" button gives you the possibility to copy text already used from other widgets



Import/Export

The import/export buttons, give you the possibility to import or export the entire messages list inside a .xml file that can be edit/modify using external tools.

Export messages inside an editable .xml file

Import messages from a .xml file

If you need to use the same message widget in different places, to save maintenance time you can create and duplicate a custom widget. When a custom widget is configured with "Only Logic" or "Full" inheritance mode, the modify (e.g. add, remove or change messages list) of a single widget will be propagated to all widgets. See "Creating a custom widget" on page 452

Attaching widget to tags

To control a widget and animate it through live data it is possible to bind a specific property to different data sources. For example it is possible to bind the gauge **Value** property to a probe temperature tag, or the **Display** property to a recipe data

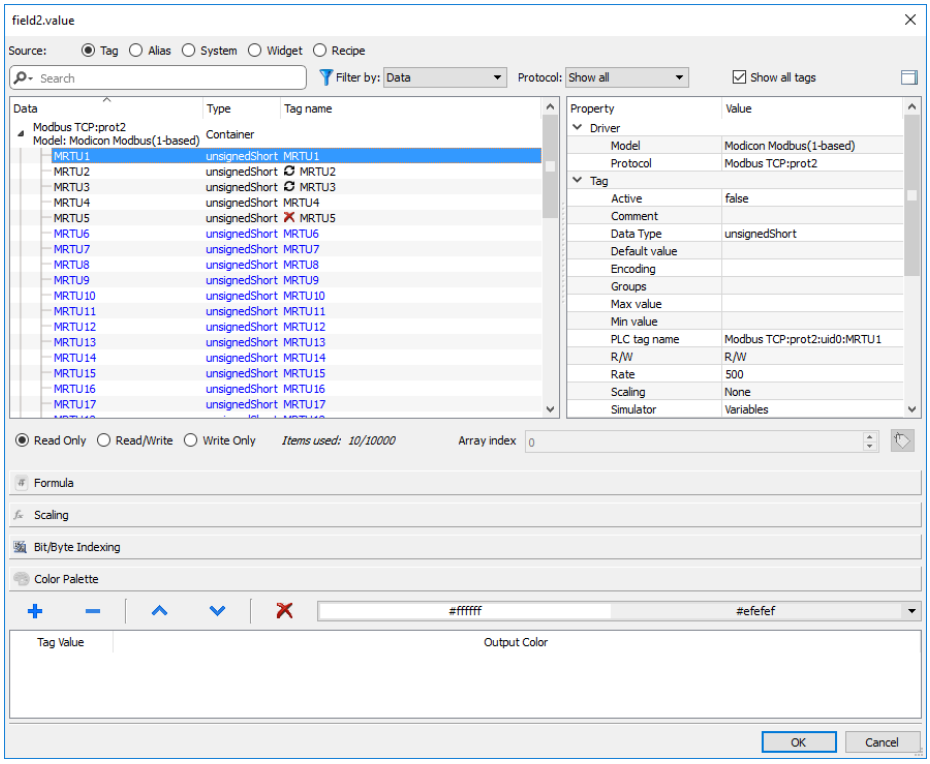
Data sources

Elements to which an object property can be attached:

Data source	Description
Tag	Tag defined in the Tag Editor
Alias	Indexed tag address
System	Predefined system tags (see "System Variables (Attach To)" on page 125)
Widget	Connect to a widget property (for example, value of a slider widget)
Recipe	Data from the Recipe Manager (see "Recipes" on page 251)

Attaching a property to a tag

1. Click **+** in the **Properties** pane.
2. In **Source** choose the data source, in the list choose a protocol and the tag. Use the **Search** box to filter tags.



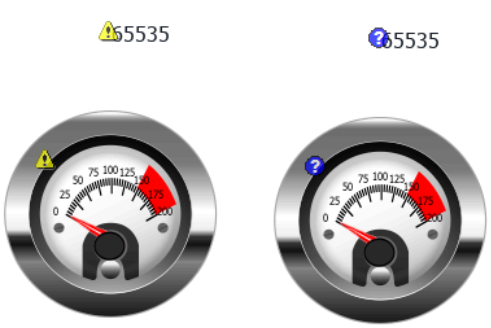
3. Set the access type (for example **Read Only**). The **Array Index** field appears when the selected tag is an array to identify the element of the array to use. The indirect index mode, through an additional tag, is supported.
4. Click **OK** to confirm.

The icons adjacent to the tag name highlight when a definition does not match the tag definition in the dictionary, or when missing. If the **Show all tags** is selected, all the dictionary tags are shown also if not imported within the application. A double-click will import the tags from the dictionary.

See ""Attach to" parameters" on page 39 for details.

Communication Error

Two icons may appear close to widgets that have an attached tag.



- : communication error
- : data not yet available (slow communication protocol)

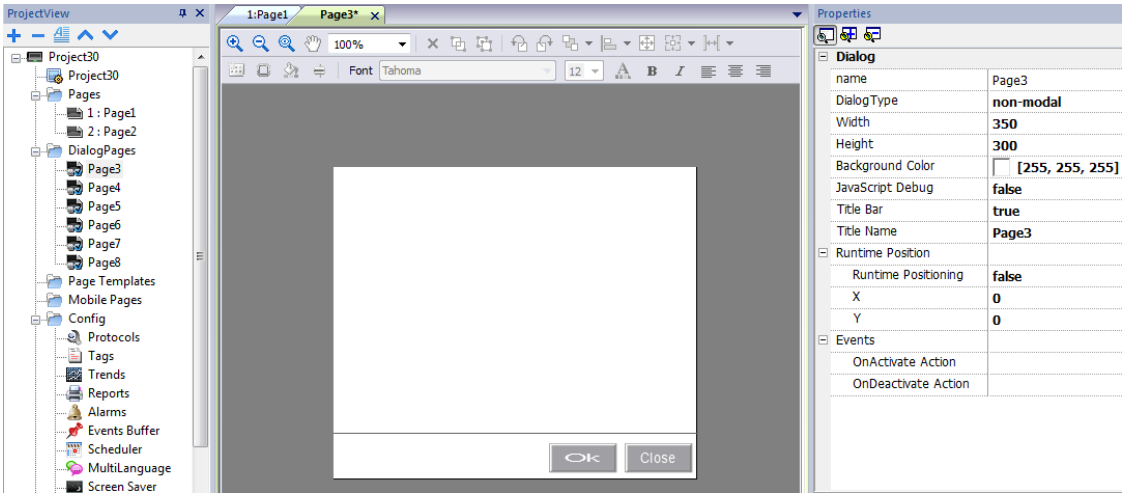
Dialog pages

Path: ***ProjectView> Dialogs***

Dialog pages are opened at runtime on top of the current page on project request. They are used to notify alarms, errors or to require user action.

Main dialog properties

Property	Description
Dialog Type	**modal** = user cannot return to main project window/page until dialog is closed. **non-modal** = user can continue to use main project window (or other non- modal dialogs) while a dialog is shown on top of it.
Title Bar	**true** = dialog title displayed **false** = no dialog title displayed
Title Name	Dialog title. Only if **Title Bar**=true.
Runtime Position	Dialog fixed position **false** = Dialog will be placed centered on the screen **true** = Dialog will be placed with upper-left corner at position X and Y



Maximum number of dialogs

Maximum number of open dialogs is defined in "Functional specifications and compatibility" on page 539.

When the maximum number of open dialogs is reached, the oldest dialog is closed to open the new one.

4 Programming concepts

Programming for LRH SW is based on a few basic concepts and behaviors.

Data types

When creating a tag you have to specify its properties. Data type are specific to LRH SW, memory type are specific to the selected protocol. Choose the value according to the internal representation you need for the selected controller address.

Note: arrays type use the same data type followed by "[ ]" (i.e.: boolean [ ])

Data Type	Memory Space	Limits
boolean	1-bit data	0 ... 1
byte	8-bit data	-128 ... 127
short	16-bit data	-32768 ... 32767
int	32-bit data	-2.1e9 ... 2.1e9
int64	64-bit data	-9.2e18 ... 9.2e18
unsignedByte	8-bit data	0 ... 255
unsignedShort	16-bit data	0 ... 65535
unsignedInt	32-bit data	0 ... 4.2e9
uint64	64-bit data	0 ... 1.8e19
float	IEEE single-precision 32-bit floating point type	1.17e-38 ... 3.4e38
double	IEEE double-precision 64-bit floating point type	2.2e-308 ... 1.79e308
string	Array of elements containing character code defined by selected encoding	

System Time

Format of System Time inside the HMI Device is the Unix time (also known as Epoch time). It is the number of seconds that have elapsed since the Unix epoch, that is the time 00:00:00 UTC on 1 January 1970.

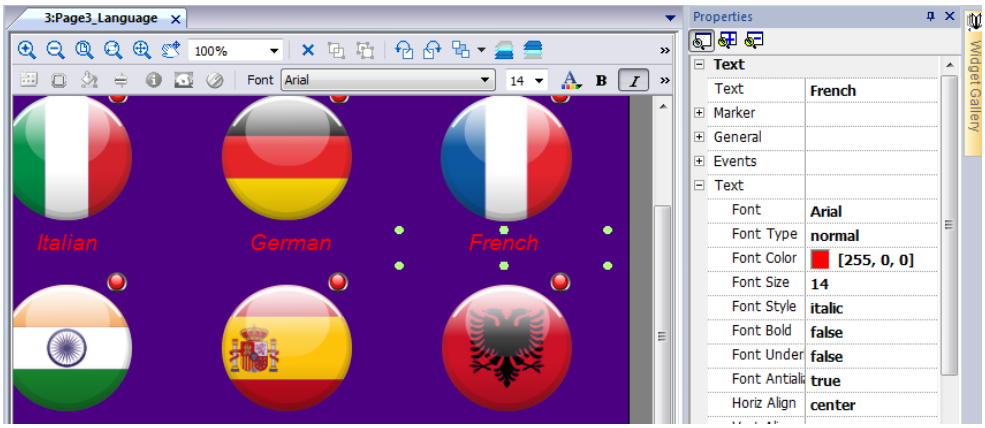
Example:

Tag Value	System Time	ISO 8601
0	01/01/1970 – 01:00:00	1970-01-01T00:00:00+00:00
1	01/01/1970 – 01:00:01	1970-01-01T00:00:01+00:00
60	01/01/1970 – 01:01:00	1970-01-01T00:01:00+00:00

"Attach to" parameters

Object properties

In LRH SW the properties of an object placed on a page can be set at programming time or configured to be dynamic. To change a property at programming time use the page toolbar or the property pane. Select the object first to see its properties displayed.

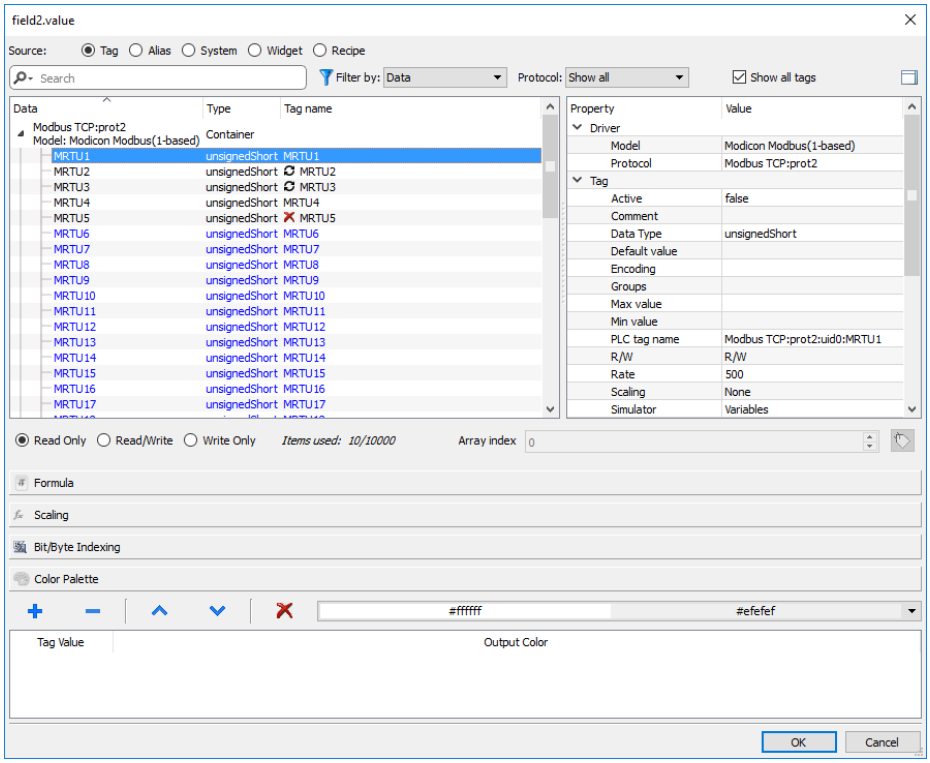


The page toolbar shows only the most common object properties, while the property pane show all the properties in a basic or advanced view.

To change a property value dynamically you can attach it to tags or variables.

Attaching a property to a tag

1. Click **+** in the **Properties** pane.
2. In **Source** choose the data source, in the list choose a protocol and the tag. Use the **Search** box to filter tags.



3. Set the access type (for example **Read Only**). The **Array Index** field appears when the selected tag is an array to identify the element of the array to use. The indirect index mode, through an additional tag, is supported.
4. Click **OK** to confirm.

The icons adjacent to the tag name highlight when a definition does not match the tag definition in the dictionary, or when missing. If the **Show all tags** is selected, all the dictionary tags are shown also if not imported within the application. A double-click will import the tags from the dictionary.

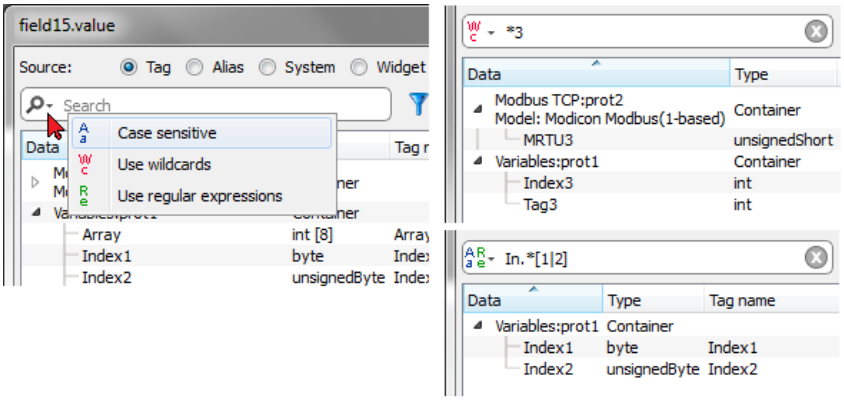
Data sources

Elements to which an object property can be attached:

Data source	Description
Tag	Tag defined in the Tag Editor
Alias	Indexed tag address
System	Predefined system tags (see "System Variables (Attach To)" on page 125)
Widget	Connect to a widget property (for example, value of a slider widget)
Recipe	Data from the Recipe Manager (see "Recipes" on page 251)

Advanced search

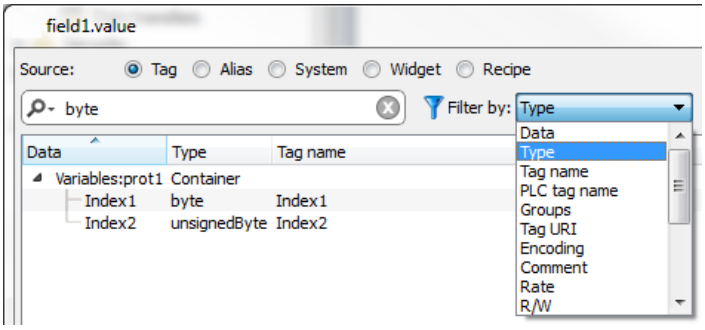
Various syntax options can be applied to search box:



Main options	Function
Wildcards	Search using simple wildcards matching . Character '?': matches any single character. Character ' *': matches zero or more of any characters." […]": sets of characters can be represented in square brackets.
Regular Expression	Describes character pattern. See https://en.wikipedia.org/wiki/Regular_expression for additional details regarding regular expressions.

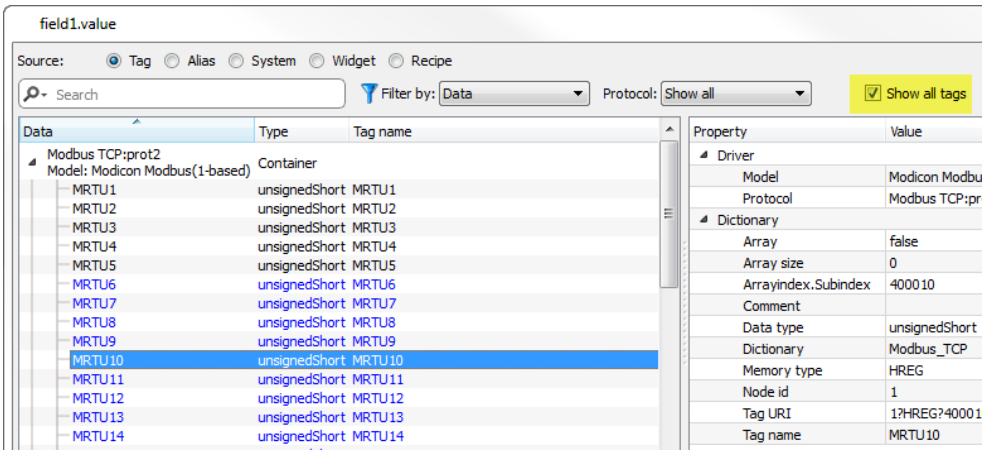
Filtering tags

Choose various tag filter criteria:

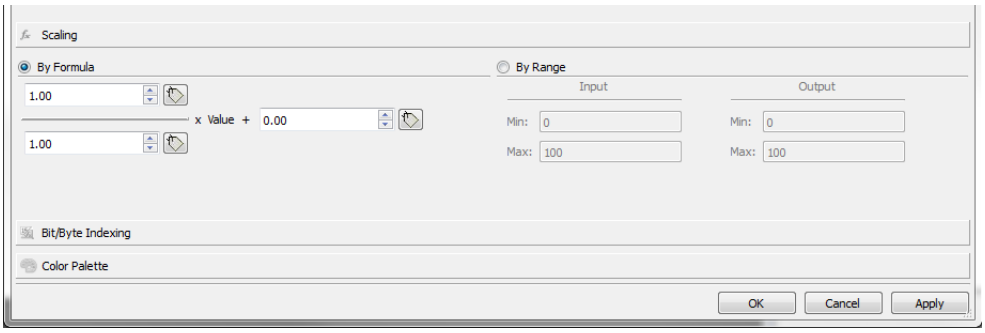


Showing dictionary tags

When **Show all tags** is checked, tags that belong to one dictionary but have not been imported yet, appear in blue color. You can select and double-click a tag to import it into the project.



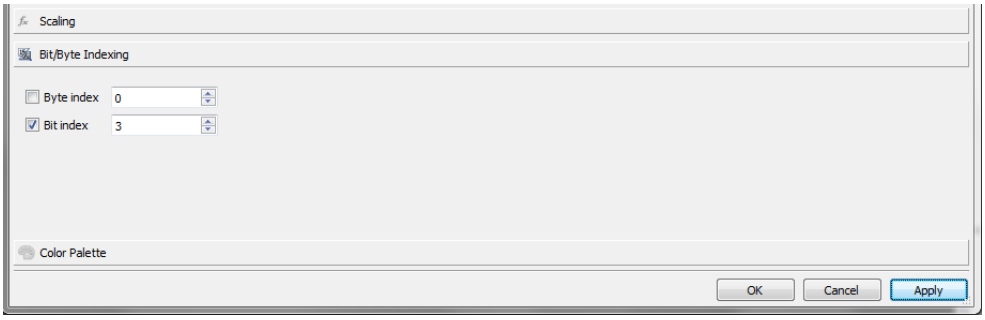
Converting tag value



Scaling tab converts the tag value. In **By Range** section set the input and output range: the system will automatically calculate the scaling factors.

Extract tag bit/byte based on index

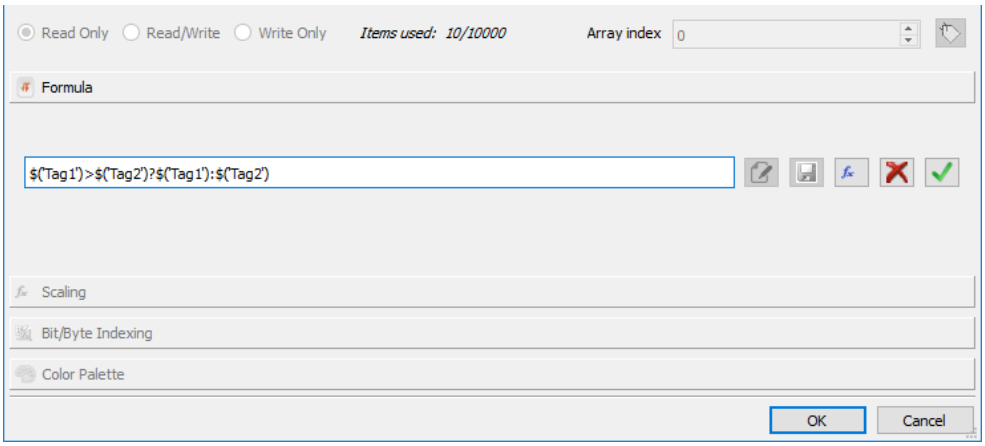
Allows extracting a single bit or byte content from a word depending on the specified bit or byte number



Use a formula to calculate the value to use

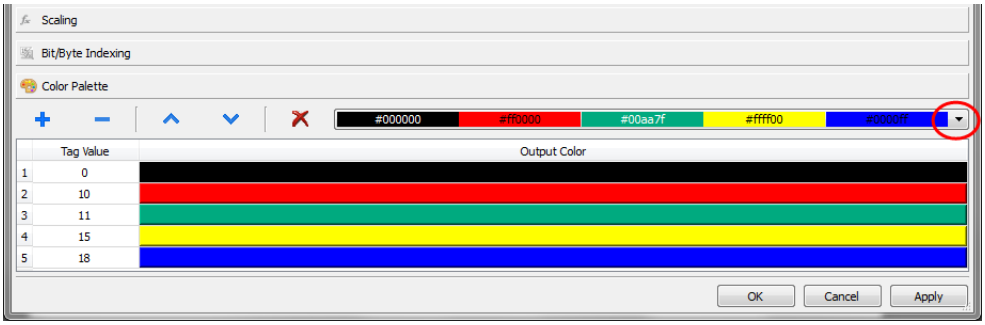
Allow to use a formula to calculate the value to use. See "Formula" on page 44 chapter for additional details.

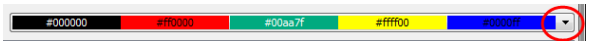
Note that using a formula the datalink will be ReadOnly



Mapping tag values to color

Allows to mapping numeric or string tag values to colors. For example, this option can be used to change the color of a button.



Section	Function
	From the toolbar add/remove or move up/down the colors lines. The tag value is editable and you can modify the sequence values.
	Last defined color combination is saved automatically and can be retrieved from the color toolbar.

Tag value could be a range of values separate by a comma, examples:

- 5, 10-15, 20
- A, AB, C

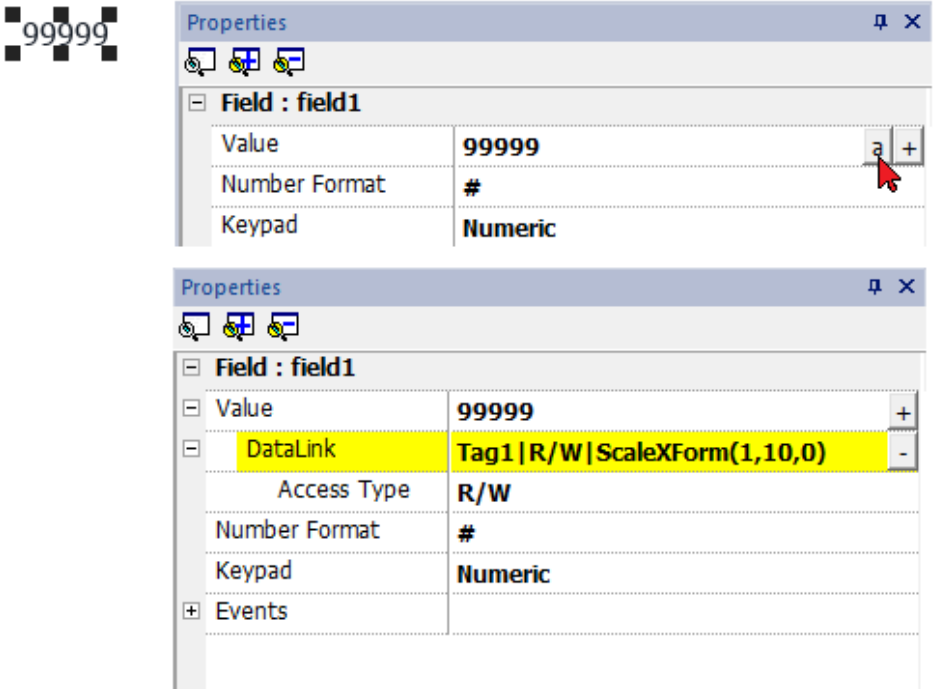
It is responsibility of the application's developer define all items correctly to cover all possible application's values, we could have unexpected color when the value is not defined inside the defined colors palette.

Note that the mapping tag value to color will return a string data type (e.g. "#FF0000")

Datalink Serialization

Instead of use the above "Attach to…" dialog box, datalinks can be entered, or modified, manually.

Click **a** button in the **Properties** pane and enter the text that describe the datalink



The data link format is:

```
Tagname [index] | [Atribute] | [XForm] | [XForm] | ...
```

on in case of formula:

```
= <formula>
```

Example:

- arrayTag[2]
- Tag[0|index]
- Alarm triggered:_SysPropMgr
- Tag|R/W|ScaleXForm(1,10,0)
- Tag|R/W|ScaleXForm(1,10,0)|ByteIndexXForm(1)|ColorPaletteCustomXForm(0#00aa7f,1#ff0000)
- =$('Tag1')>$('Tag2')?$('Tag1'):$('Tag2')
- =$Contains($('Tag4'),$('Tag3'))
- =$Pow(2,$('Tag2'))

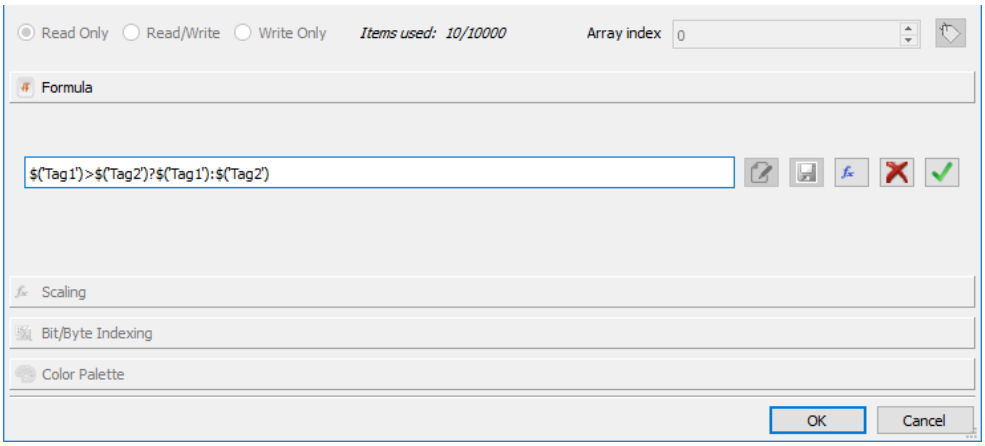
Formula

A formula is an expression made of:

- Operators: can be the basic mathematics operations, logic operators, compare operators or basic string operators.
- Operands: can be literals (numbers and strings used as constants) and references to tags.

Round brackets are supported as priority operators. The operator $ will be used to call functions and, in particular, to referring to a tag (see below for examples).

The attach to dialog allow to use a formula to calculate the value to return.



Commands

 Enter edit mode

 Save the entered formula inside the formulas' library to have the possibility to reuse the same formula inside other places of the project.

 Open the formulas' library to select an already defined formula.

 Removing the entered formula

 Confirming the entered formula

When you are in edit mode you can simple edit the formula and double click tags or functions from the library to add them inside the formula.

Example of formulas are:

- $('Tag1')+$('Tag2')
- $('Tag1')&$('Tag2')
- $('Tag1')>$('Tag2')?$('Tag1'):$('Tag2')
- $Pow(2,$('Tag2'))
- $Contains($('Tag1'),$('Tag2'))

Syntax for formula

Basic Operations	
'Text'	String literal
NUMBER	Number literal, e.g. 169857 or 13.547
String(…)	Cast to string (note there is not $)
Number(…)	Cast to number (note there is not $)

Basic Operations	
$FuncName(param1, param2, …)	General function call. (Both default and user ones)
$('TagName')	Tag, or widget property, or recipe, etc.  **Note that tag name must be string literal**
$('TagName')[index]	Element of a array tag.  **Note that tag name must be string literal**
exp1 ? exp2 : exp3	Ternary expression. If exp1 is true, then is taken exp2, otherwise is taken exp3. This is like using if/then/else statement

Math Operators	
+	Addition
-	Subtraction
*****	Multiplication
/	Division
%	Module

Bitwise Operators	
&	Sets each bit to 1 if both bits are 1
|	Sets each bit to 1 if one of two bits is 1
~	Inverts all the bits
^	Sets each bit to 1 if only one of two bits is 1
<<	Shifts left by pushing zeros in from the right and let the leftmost bits fall off
>>	Shifts right by pushing copies of the leftmost bit in from the left, and let the rightmost bits fall off
>>>	Shifts right by pushing zeros in from the left, and let the rightmost bits fall off

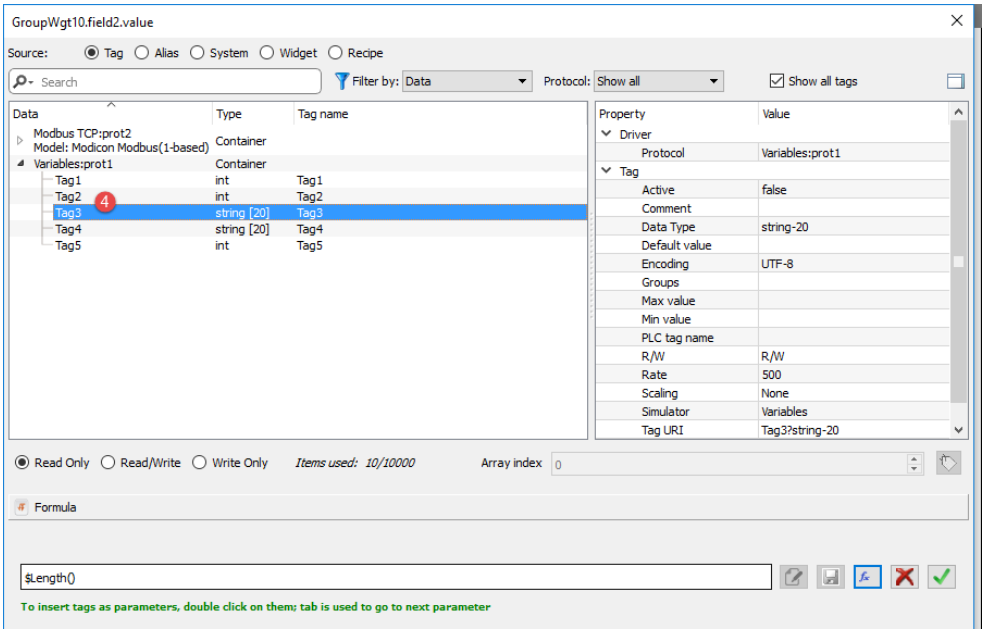
Logical Operators	
&&	AND
||	OR
!	NOT

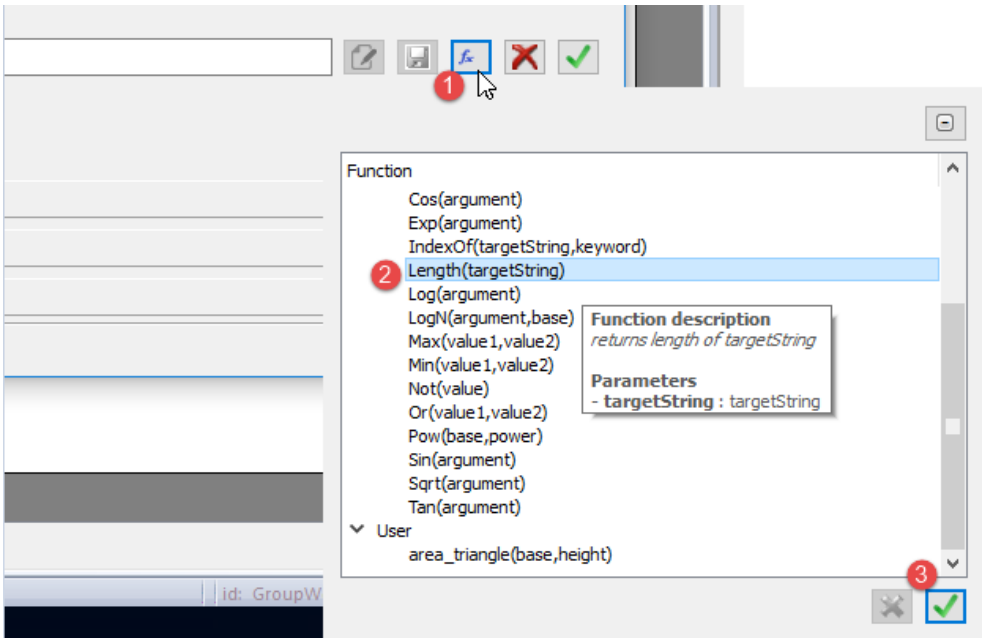
Compare Operators	
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
==	Equal to
!=	Not equal to

Use predefined formula from the library

To insert a project or a predefined formula:

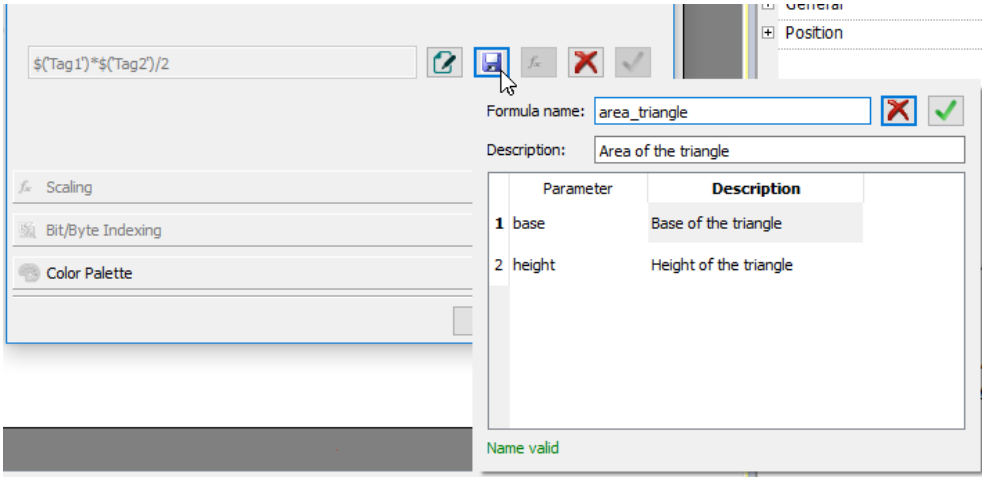
1. Open the formulas' library
2. Select the formula to use
3. Confirm the selected formula
4. Enter the arguments required from the selected formula



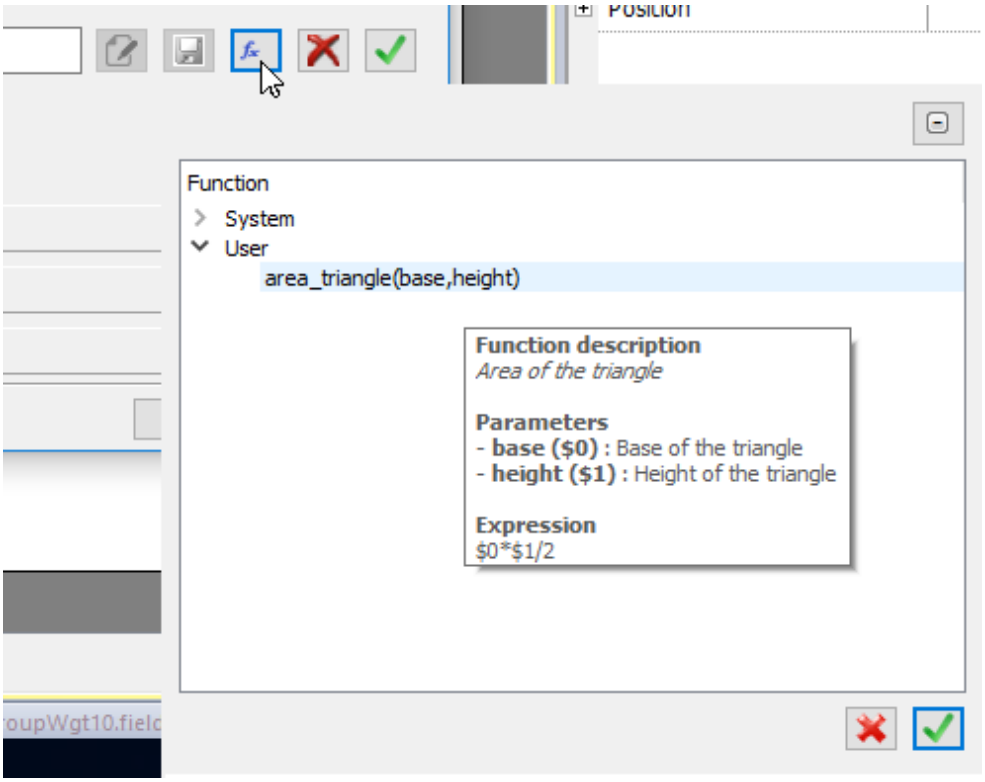


Add user formulas into the library

After entering a new formula, using SAVE button is possible to store the new formula inside the project folder to make it available from the formulas' library.



A user formula could be retrieved from the formulas' library as for the other predefined formulas.



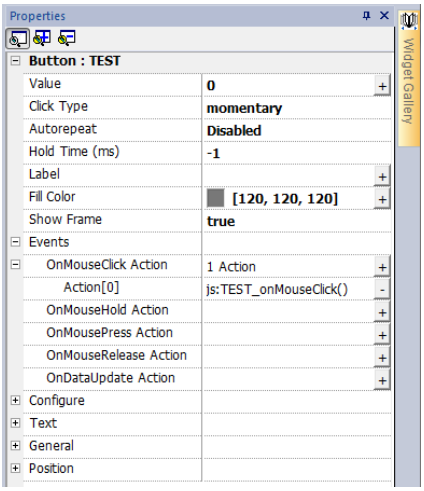
Events

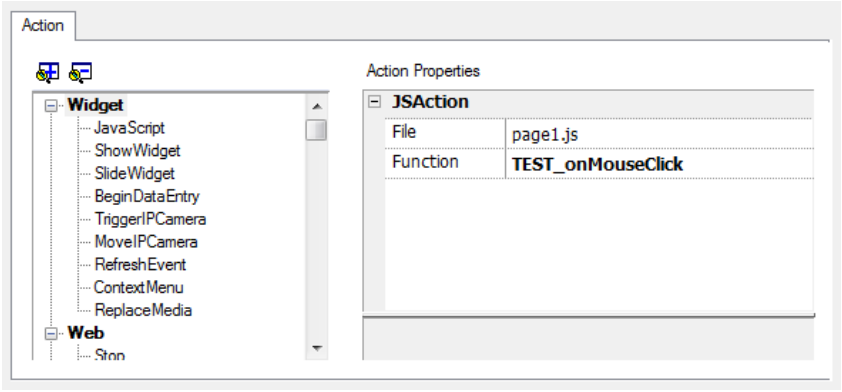
Events are used to trigger actions at project level and can be associated to:

- buttons / touch (click, press, release)
- mouse wheel
- external input devices like keyboards and mouse (click, press, hold, release, wheel)
- data changes (OnDataUpdate)
- switch of pages (OnActivate, OnDeactivate)
- alarms
- scheduler

You can attach one or more actions to an event, so that they will be executed whenever the event occurs.

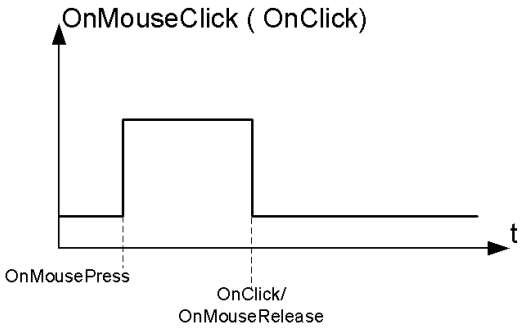
This example shows a JavaScript action activated by pressing a button.





OnClick / OnMouseClick

Triggers the event when the button/key is pressed and released quickly.

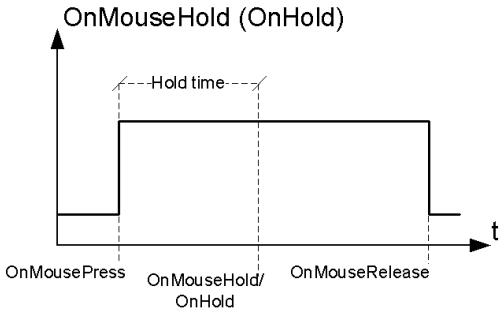


OnHold/OnMouseHold

Triggers the event when the button/key is pressed and held pressed for a certain time set as **Hold Time** in the widget properties. Actions programmed for this event will be executed only after the hold time has expired.

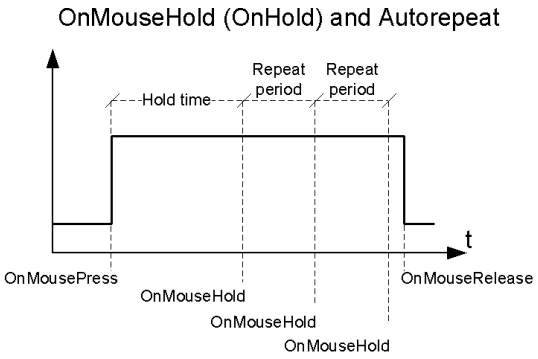
The default **Hold Time** is configured in Project properties but can be redefined for each button/key. See "Project properties" on page 73.

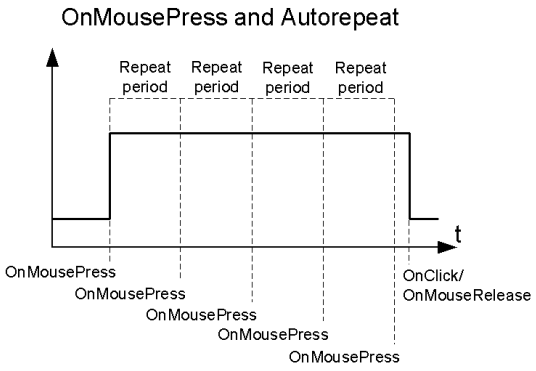
Note: If **Hold Time** is set to -1 for the widget, the project **Hold Time** value will be used.



Autorepeat

Enables auto repeat for a press or hold event of button or key. **Autorepeat Time** is specified in the Project properties but can also be redefined for each button or key





OnWheel

Triggers the event when a wheel (for example a USB mouse wheel) value changes. A wheel usually is used to increase/decrease values in a text box or attached to a tag.

OnActivate

Triggers the event when a page is loaded. The event starts before widgets in the page are initialized.

OnDataUpdate

Triggers the event when the tag value changes. The update moment depend on the time needed by the protocol to finish the update process. For example the **OnDataUpdate** event can be triggered or not, depending on whether data becomes available from protocol respectively after or before widgets being initialized for the first time. In particular, page change notifications are more likely to happen with slow protocols and remote clients.

Note: The value read during **OnActivate** can be the same obtained from a subsequent **OnDataUpdate** event, since **OnDataUpdate** notifications are sent asynchronously.

Widgets positioning

You can position widgets in the page using two methods:

- Snap to Grid
- Snap to Object

To display the grid, on the **View** menu, click **Show Grid**.

Snap to Grid

*Path: **View**> **Snap to Grid***

When you move or re-size an object, its top left corner will align with the nearest intersection of lines in the grid, even if the grid is not visible.

Setting grid properties

*Path: **View**> **Properties***

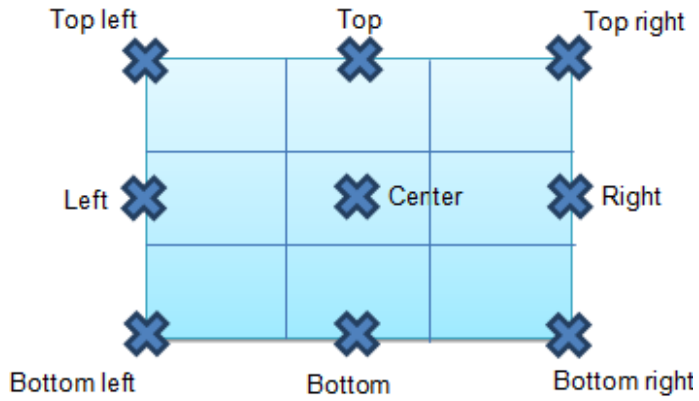
Parameter	Description
Spacing X	Space in pixel between two lines/dots on the X axis
Spacing Y	Space in pixel between two lines/dots on the Y axis
Type	Grid type (dot or line)
Color	Grid color

Snap to Object

*Path: **View**> **Snap to Object***

When you move an object, it will align with other objects on the page.

When you select an object, one of the following hot points is selected as the source of the snap point, depending on the area you pressed: top, top left, top right, bottom, bottom left, bottom right, left, right, center:

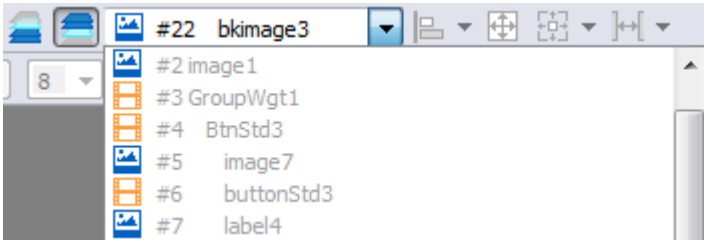


An algorithm finds a matching hot point among the near widgets hot points matching either the x or the y coordinates of the source snap point. For line widgets, the source snap points are the terminal points of the line.

Managing overlapping widgets

When one or more widgets on the page overlap, you can manage their order so that one is displayed on top of the other.

The order of the widget on the page is shown in the combo box. A widget with greater z-order number is in front of an element with a lower z-order number. A picture icon identifies static objects, a movie frame icon identifies dynamic objects.



Important: Correct ordering of widgets is essential for runtime performance since overlapping dynamic widgets can invalidate static optimization and reduce performance of HMI applications.

Hiding/showing widget on z-order

To hide widgets above a selected widget:

- On the toolbar click and select a widget: all widgets above this one are hidden

To hide widgets below a selected widget:

- On the toolbar click and select a widget: all widgets below this one are hidden

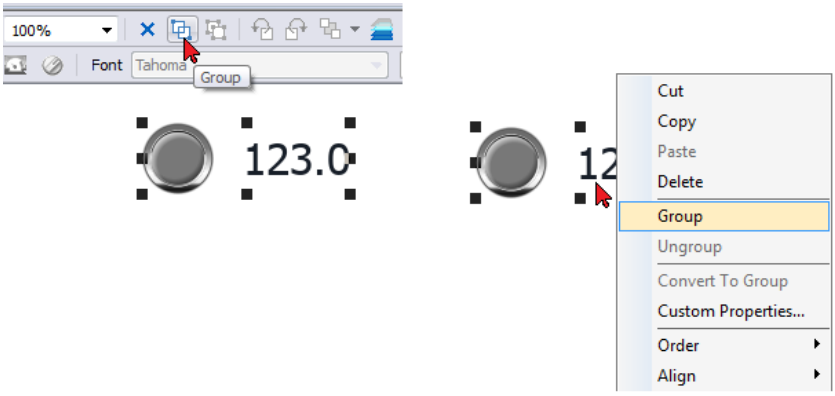
The toolbar allows to:

- hide widgets stacked above and/or below selected widgets
- work on different widgets using the combo box which lists all the widgets in their z-order.

Grouping widgets

To group widgets:

1. Select all the widgets to group.
2. Right-click and then click **Group**.

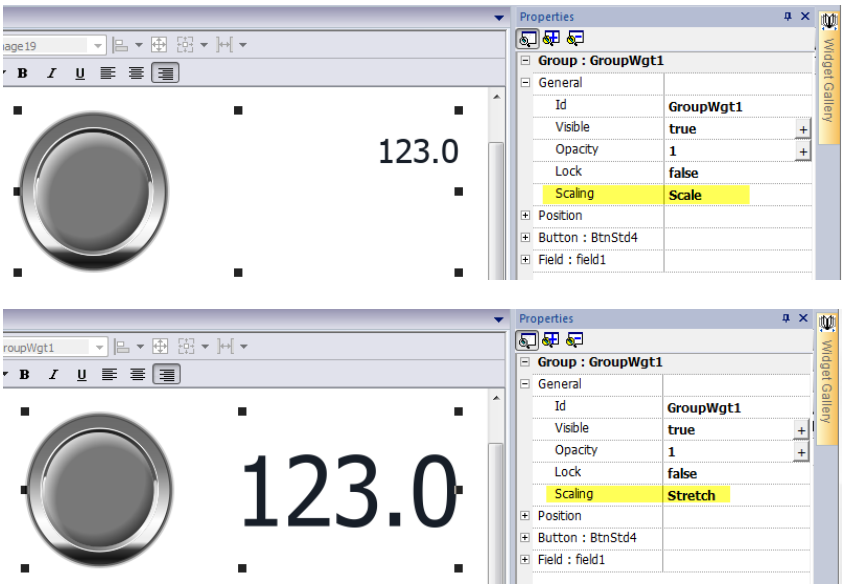


Tip: Double click to enter the group editing mode. In group mode only the group widgets are editable and selectable. All other widgets are partially hidden

Resizing grouped widgets

You can define how object reacts when re-sized. Use the **Scaling** property in **General** section:

- **Scale**: object and text are not re-sized proportionally
- **Stretch**: object and text are re-sized proportionally



Grid Layout Group

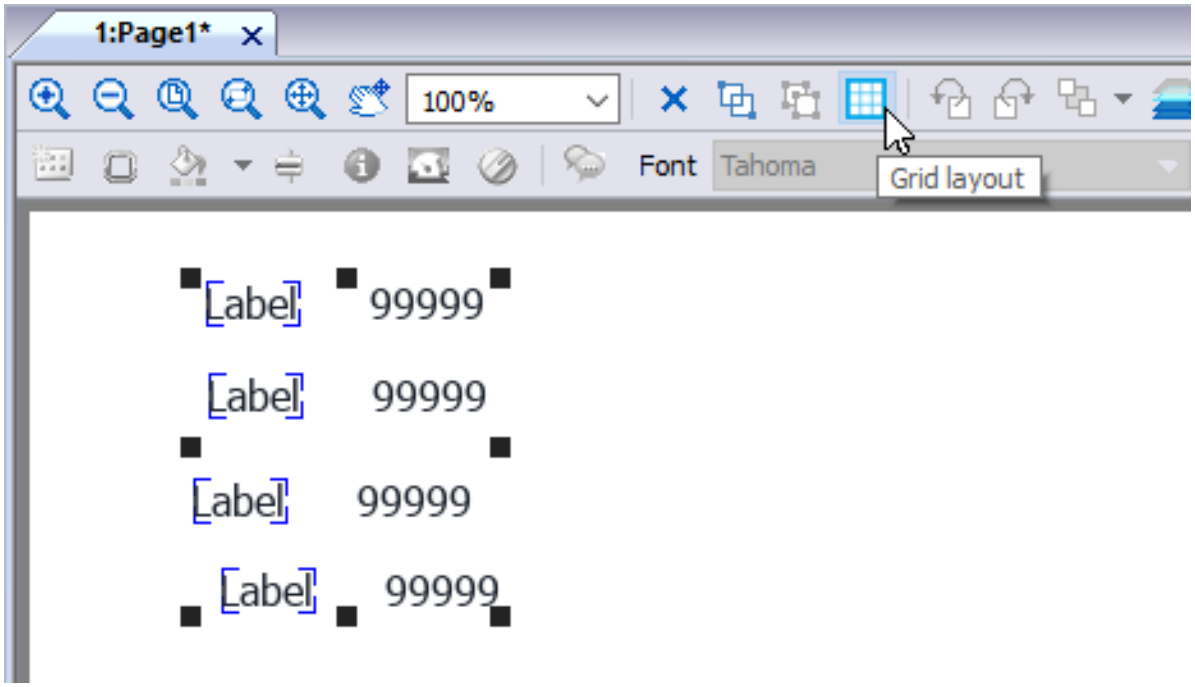
The grid layout add the possibility to configure the spatial relationships among the widgets of the group.

To create a grid layout:

- Enable the "*Grid Layout*" parameter of the group of widgets.

or

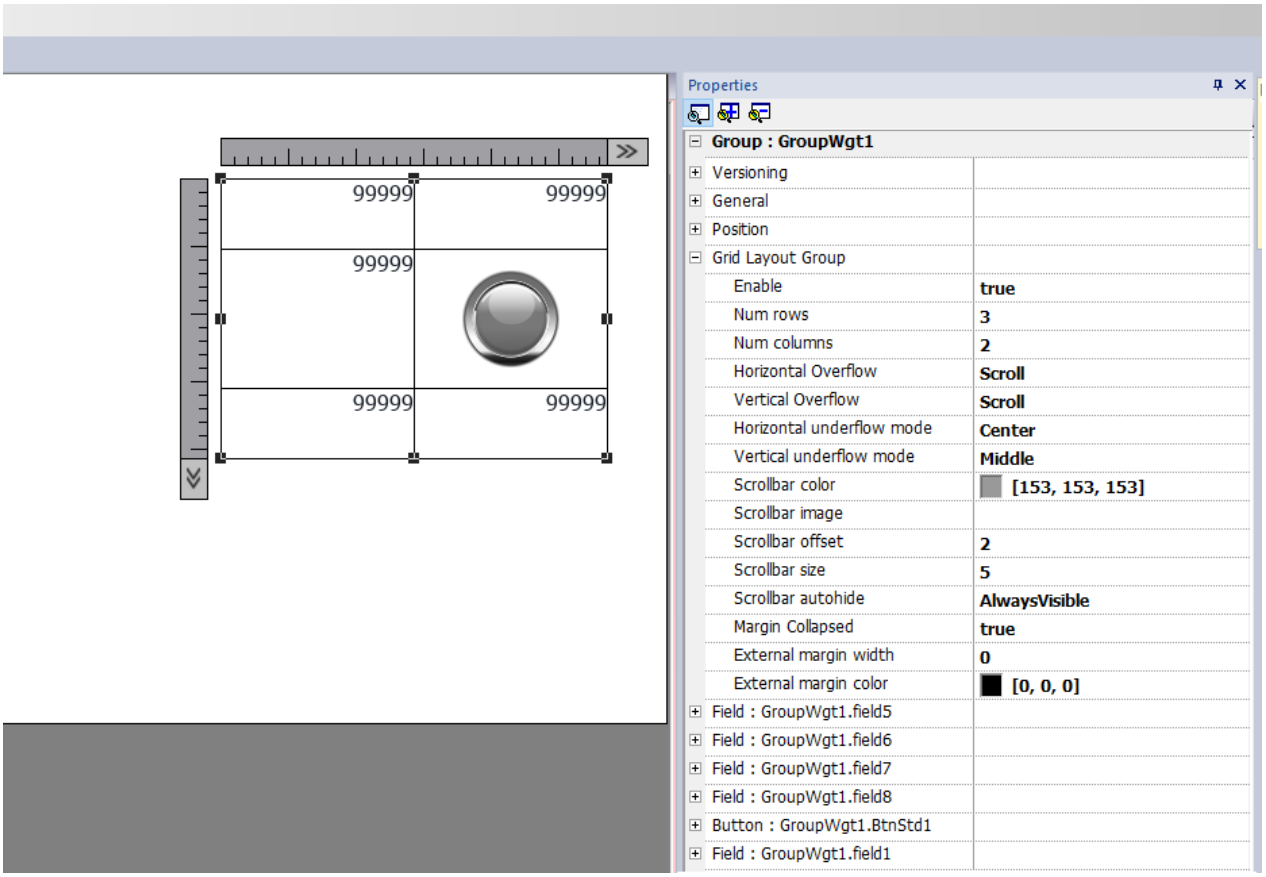
- Select the widgets that will be inside the table and click the “Grid Layout” button on page toolbar. The selected widgets will be aligned and collected inside a group with the grid layout property enabled.



There are several elements associated with the grid layout that can be configured:

- Grid properties
- Rows, Columns Properties
- Cells Properties

Grid Properties

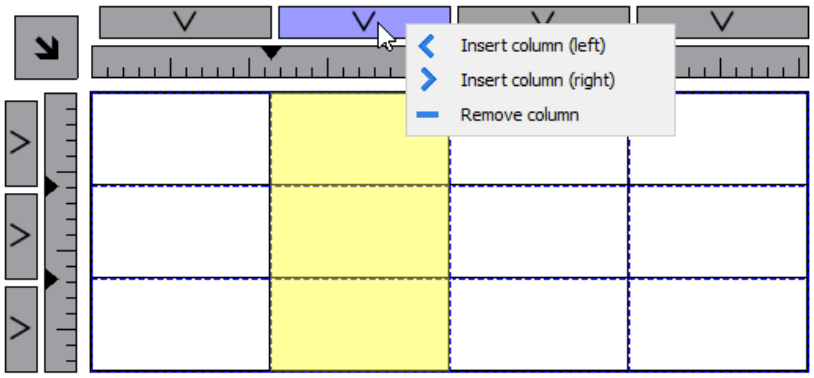


Parameter	Description
Enable	Enable the grid layout. A grid will be generate around the widgets of the group
Num rows **Num columns**	Number of rows and columns of the grids. Rows and columns can be removed only if their cells are empty .
Horizontal overflow **Vertical overflow**	This parameter define the behavior of the grid when it is too small to contain all rows and columns. • Hidden Rows and columns that do not fit into the grid are not displayed • Visible The grid can not be made smaller than the minimum size required to contain all defined rows and columns • Scroll When the grid is too small to hold all the defined rows and columns, the scroll bars can be used to shift the content of the grid.

Parameter	Description
Horizontal underflow **Vertical underflow**	This parameter defines the behavior of the grid when it is larger than the size defined for the rows and columns • Blocked The grid can not be made larger than the maximum size of rows and columns • Left, Center, Right - Top, Middle, Bottom Defines the position of the widgets when cells are bigger than the maximum defined sizes
Scrollbar color **Scrollbar image** **Scrollbar offset** **Scrollbat size** **Scrollbar autohide**	Parameters to define look and position of the scroll bars
Margin collapsed	Collapse all left-right and top-botton margin using the parameters of the stroke with greater width.
External margin width **External margin color**	External margin parameters

Add or remove rows or columns

To add or remove rows or columns, double click over the grid to enter in edit mode and right click over column or row selector to open the context menu.



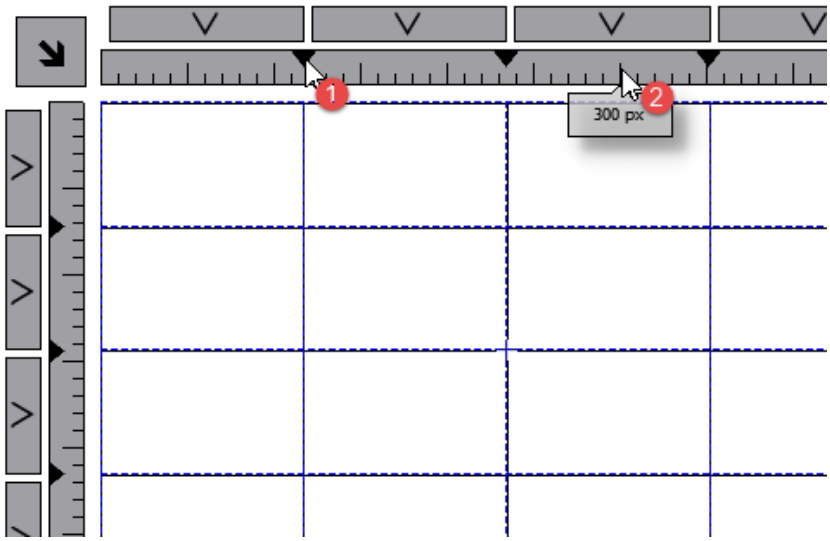
Merge or split rows or columns

To merge or split rows or columns, double click over the grid to enter in edit mode and move the cursor over the ribbons:

- Double click the black triangle to merge the two adjacent rows or columns (1)

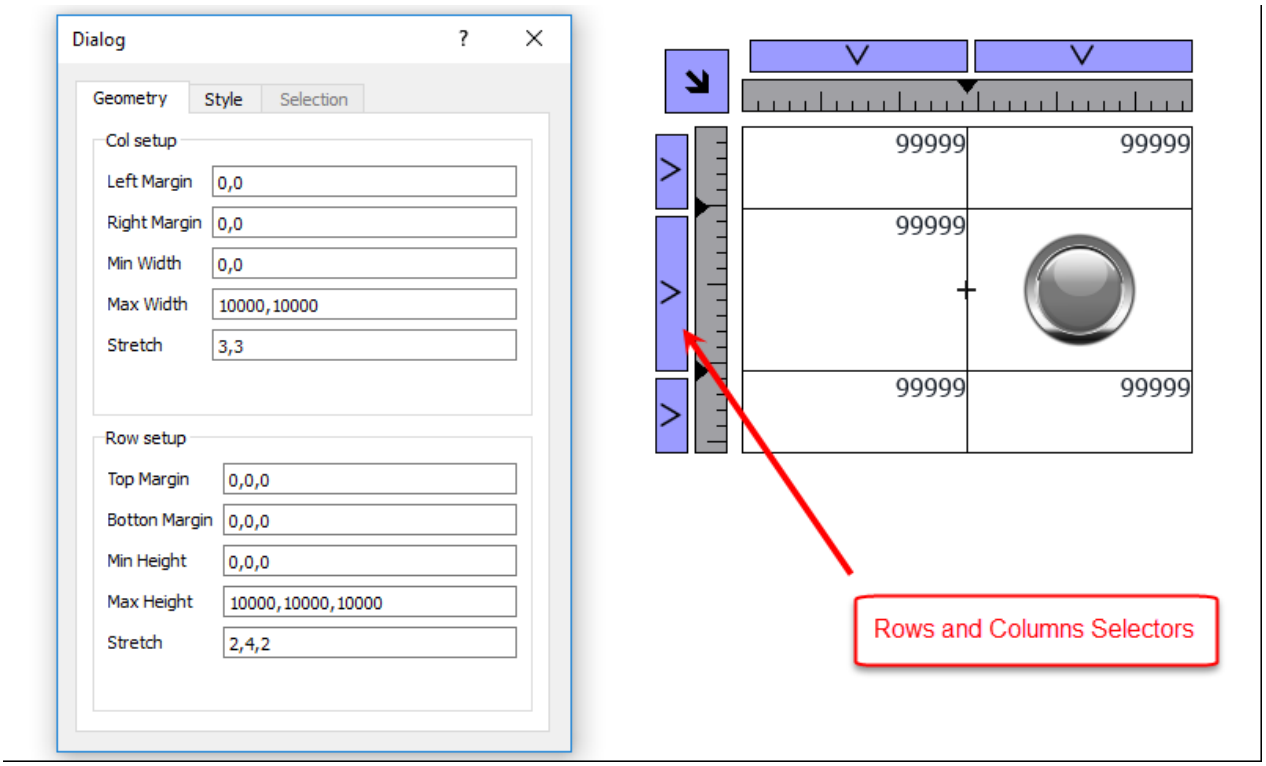
  Note that merge is possible only with an empty row or column.

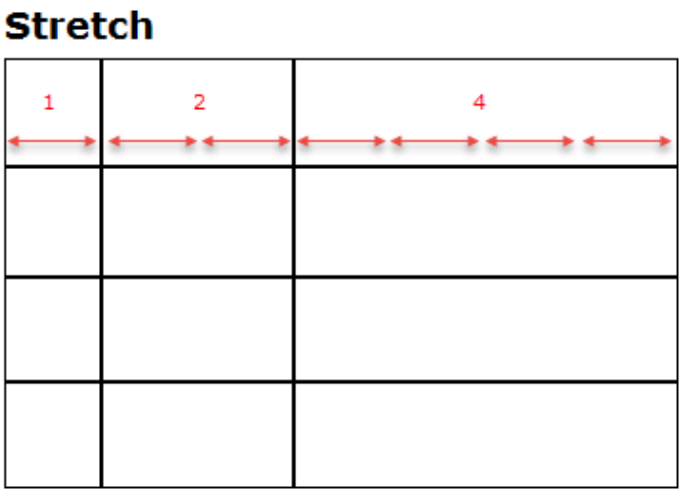
- Double click on ribbon to split the selected row or column (2)

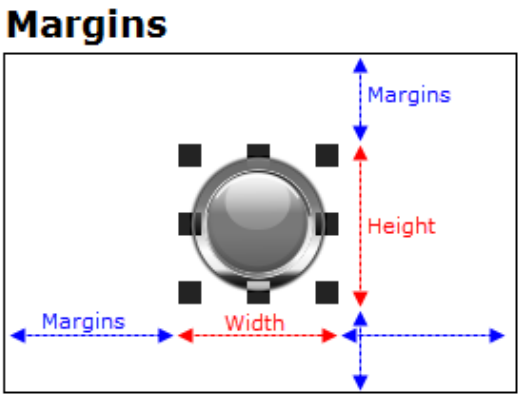


Rows, Columns Properties

Row and columns properties are available inside a pop up dialog after clicking on the row and column selectors, that are visible after double clicking the group of widgets.







Geometry parameters

Parameter	Description
Left margin **Right margin**	Distance of the widget from the border of the cell
Min width **Max width**	Min/Max width that widget can assume when the cell is stretched
Stretch	Defines the relationship between the widths of the columns that will be maintained if the grid is stretched
Top margin **Bottom margin**	Distance of the widget from the border of the cell
Min height **Max heighty**	Min/Max height that widget can assume when the cell is stretched
Stretch	Defines the relationship between the heights of the rows that will be maintained if the grid is stretched

Style parameters

Parameter	Description
Left stroke width **Right stroke width** **Top stroke width** **Bottom stroke width**	Strokes width
Left stroke color **Right stroke color** **Top stroke color** **Bottom stroke color**	Strokes color
Background color	Row background color

The list of values that are separated by a comma, are related to rows and columns. Example, the first value is for row 0, the second value for row 1, and so on.

Color format could be #rrggbb or #rrggbbaa, where "*aa*" is the alpha value which defines the opacity of the color.

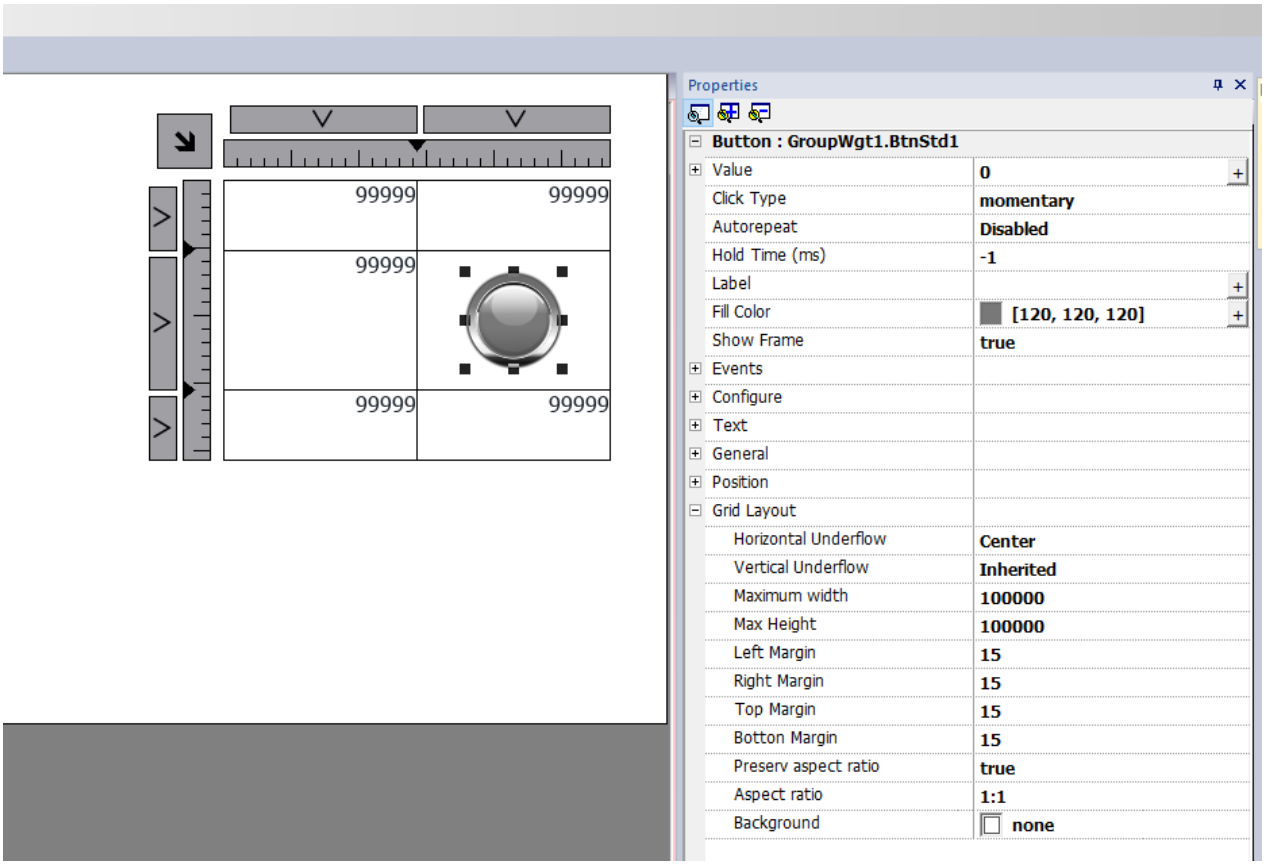
Selection parameters

The selection parameters is available only when the grid is used inside a Table Widget (see "Table widget" on page 431 for details)

Parameter	Description
Forground color **Background color** **Stroke color**	Colors that the row assume when it is selected The list of colors is related with row templates. First color is for row template 0, second color is for row template 1, and so on.

Cells Properties

Properties of a single cell are available inside the properties panel when a cell is selected. To select a cell: first double click the widget group, then click the cell to select.



Parameter	Description
Horizontal underflow **Vertical underflow**	This parameter defines the behavior of the widget when the cell is larger than the size defined for widget. • Inherited Inherits the value used for the row or column • Left, Center, Right - Top, Middle, Bottom Defines the position of the widgets when cells are bigger than the maximum defined sizes
Max width **Max height**	Overwrite global grid parameters
Left margin **Right margin** **Top margin** **Bottom margin**	Overwrite global grid parameters Additional pixels that are added to the total margin.
Preserve aspect ratio	Preserve aspect ration of the widget
Aspect ratio	Available only when "*Preserve aspect ratio*" is true
Background	Background color of cell

Printing report

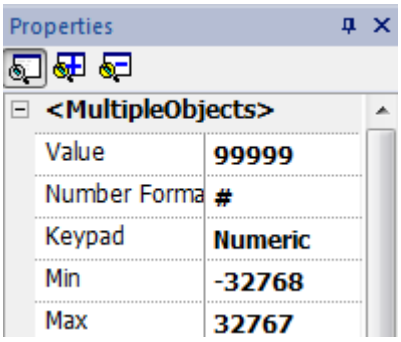
Note the grid layout is available even inside the print report gallery.

Changing multiple widgets properties

You can set the properties of more widgets of the same type all at once.

To change properties:

1. Select widgets.
2. Set common properties from **Properties** pane.
3. When multiple widgets are selected, the Properties pane title changes to **<MultipleObjects>**: all changes will be applied to all selected widgets.



Note: Not all properties can be modified for multiple widgets simultaneously and must therefore be modified individually.

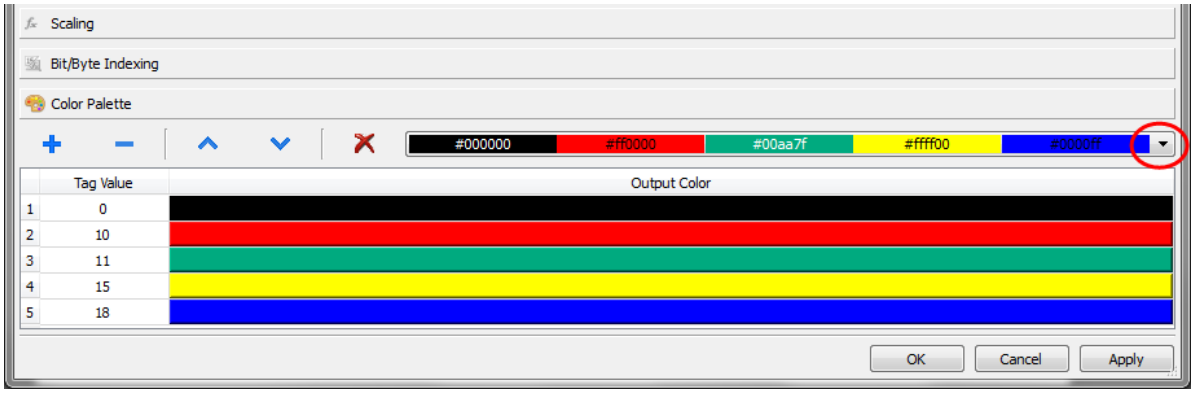
Changing fill color property according to tag values

LRH SW allows to change the color property of a widget dynamically, based on tag values in two ways:

- Using ColorPalette
- Connecting the Color property to a String type tag

Changing color property using ColorPalette

1. Create the tag (internal or PLC) that you want to refer to for color management. The tag can be of any data type. On the basis of the value of this tag, the color will change.
2. Attach this tag to the **Fill Color** property of an object (for example, a button).
3. In the same dialog select the **ColorPalette** tab and add the colors that will be used for the object according to the tag value.



Note: The last used colors' tables are saved and can be reused selecting them from the colors list box on the toolbar.

Changing color property connecting Color property to a String type tag

1. Create the tag (internal or PLC) that you want to refer to for color management. On the basis of the value of this tag, the color will change. The tag must be of String type and the **Arraysize** property of the tag must be big enough to contain the string formatted as explained here.
2. Attach this tag to the **Fill Color** property of an object (for example, a button).
3. Write in the **String** tag the RGB color code of the required color. Use one of these formats:

- **#XXYYZZ**, Where XX, YY and ZZ are the RGB components of the needed color expressed in Hexadecimal format (range 00–FF).
- **rgb(XXX,YYY,ZZZ)**, where XXX, YYY and ZZZ are the RGB components of the needed colors expressed in Decimal format (range 0–255).

Note: This feature can be applied to all the objects available in the Widget gallery that have a color property. The runtime change of the color is possible only thanks to the properties of the SVGs that are composing the object. This feature can not be applied to other image formats such as JPEG or BMP files.

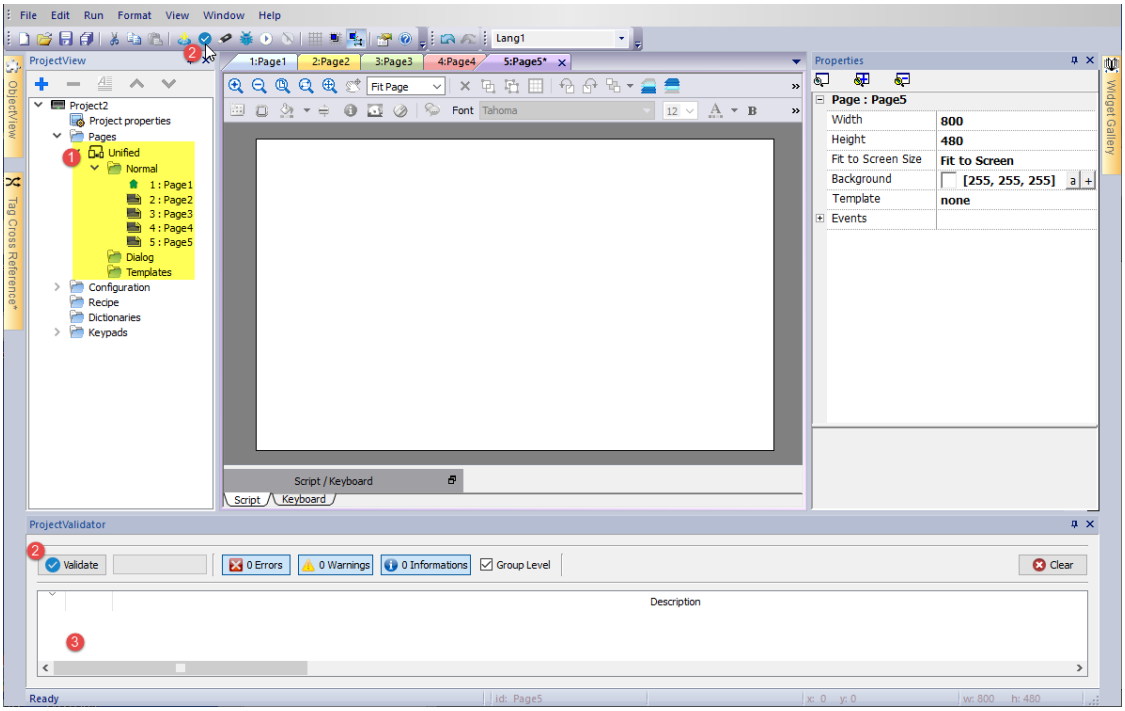
5 Pages

This section describes how pages are organized. You can have the same pages shown inside all clients (default mode) or you can customize the pages to better adapt them to each different client.

Unified pages

Starting from LRH SW v4.0 there is no longer a need to create the pages for HMI device and Web client differently. The same pages can be rendered indifferently on the HMI device or on Web clients. Since some properties or some widgets could be not supported on Web client, some pages could be render differently. The "Project Validator" tool can be used to check if some pages contain widgets that will be rendered differently into Web client.

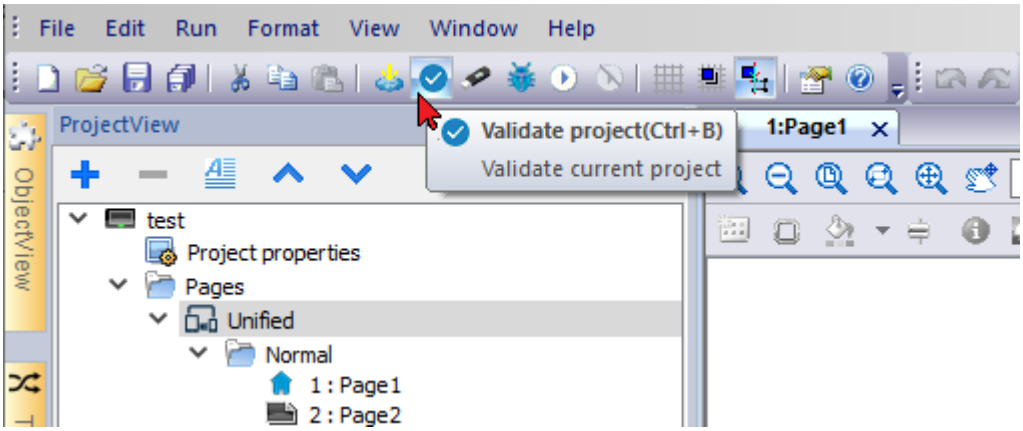
1. Pages
2. Project Validator button
3. Project Validator output messages



Project Validator

The "Project Validator" tool check and list the widgets that will be rendered differently into Web client. User can double click each warning message reported from the Project Validator to open the pages that contain the reported widgets to take the appropriate action. However, user action is not mandatory, the project can be downloaded anyway and the unsupported property will not be managed from the Web Client.

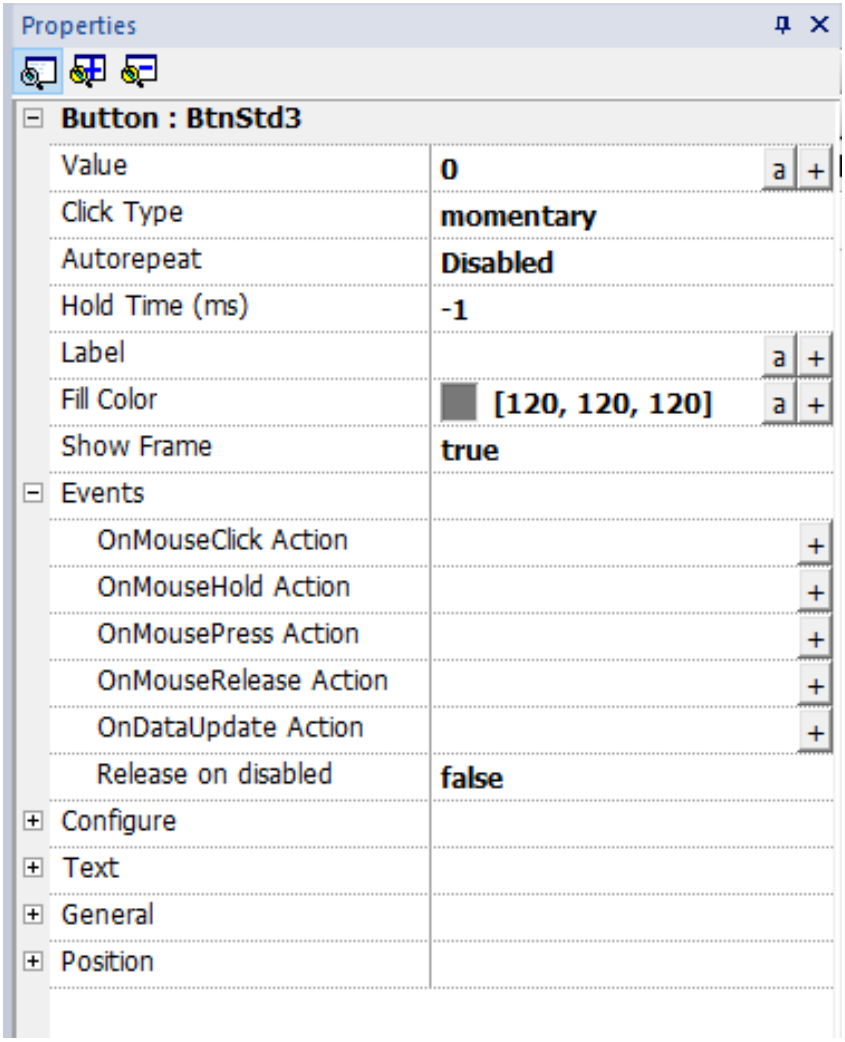
Note that the current version of the Project Validator checks the widget's web compatibility. It is not checking the entire project (e.g. missing tags or Javascript errors)



Example

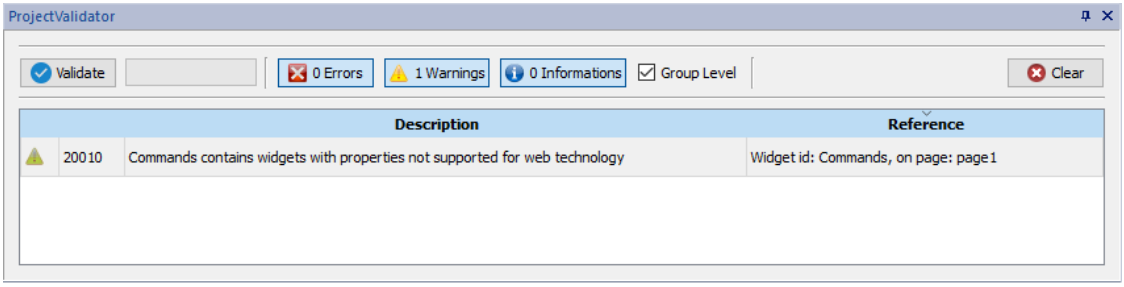
The "Release on disabled" property is not supported on the Web client.

- When the property is set to "False", HMI device and Web client will work in the same way and project validator will not report any message.
- When the property is set to "True", the Project Validator will report the warning message. In the case that the project will download to the HMI device, the Web client simply will not manage the "Release on disabled" property.

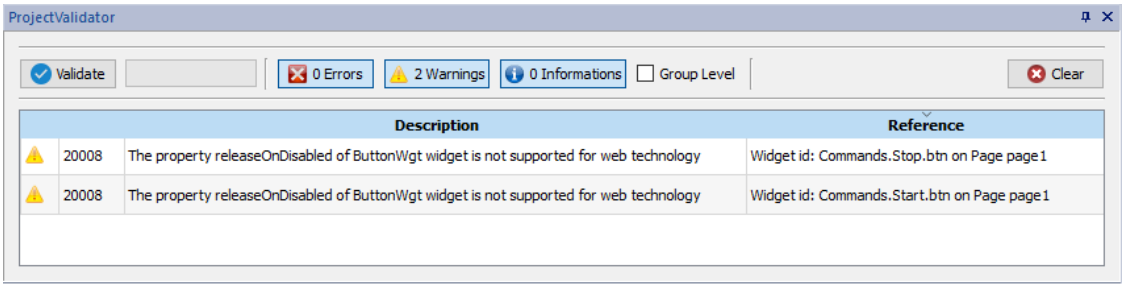


Group Level

When the "Group Level" is checked, the Project Validator will report the group name that contains one or more widgets with the unsupported properties. A double click will select the grouped widget.



When the "Group Level" is not checked, the Project Validator will report the list of the not supported properties. A double click will select the widget that have the unsupported property.



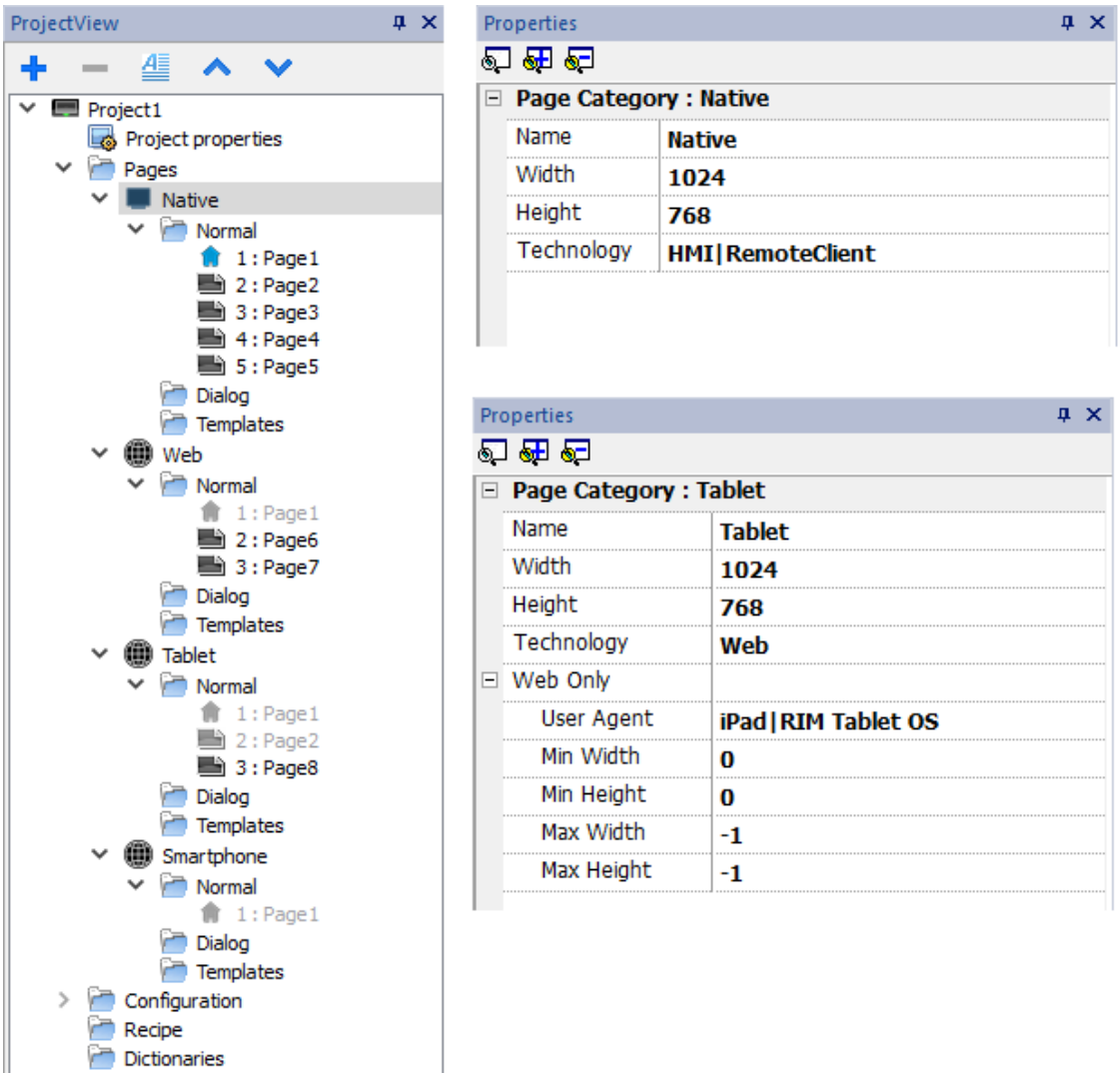
Differentiated pages

If a project needs to have different pages for the HMI device, web client, tablet client, etc., there is the possibility to add different folder to contain the pages to use on the different clients. Right click on the page folder to add a new category of pages. For each category, you have to define the below properties where Technology, User Agent and Min/Max are filter parameters to define the web clients that belong to the category.

Property	Description
Name	The category name
Width, Height	The default size used when create a new page

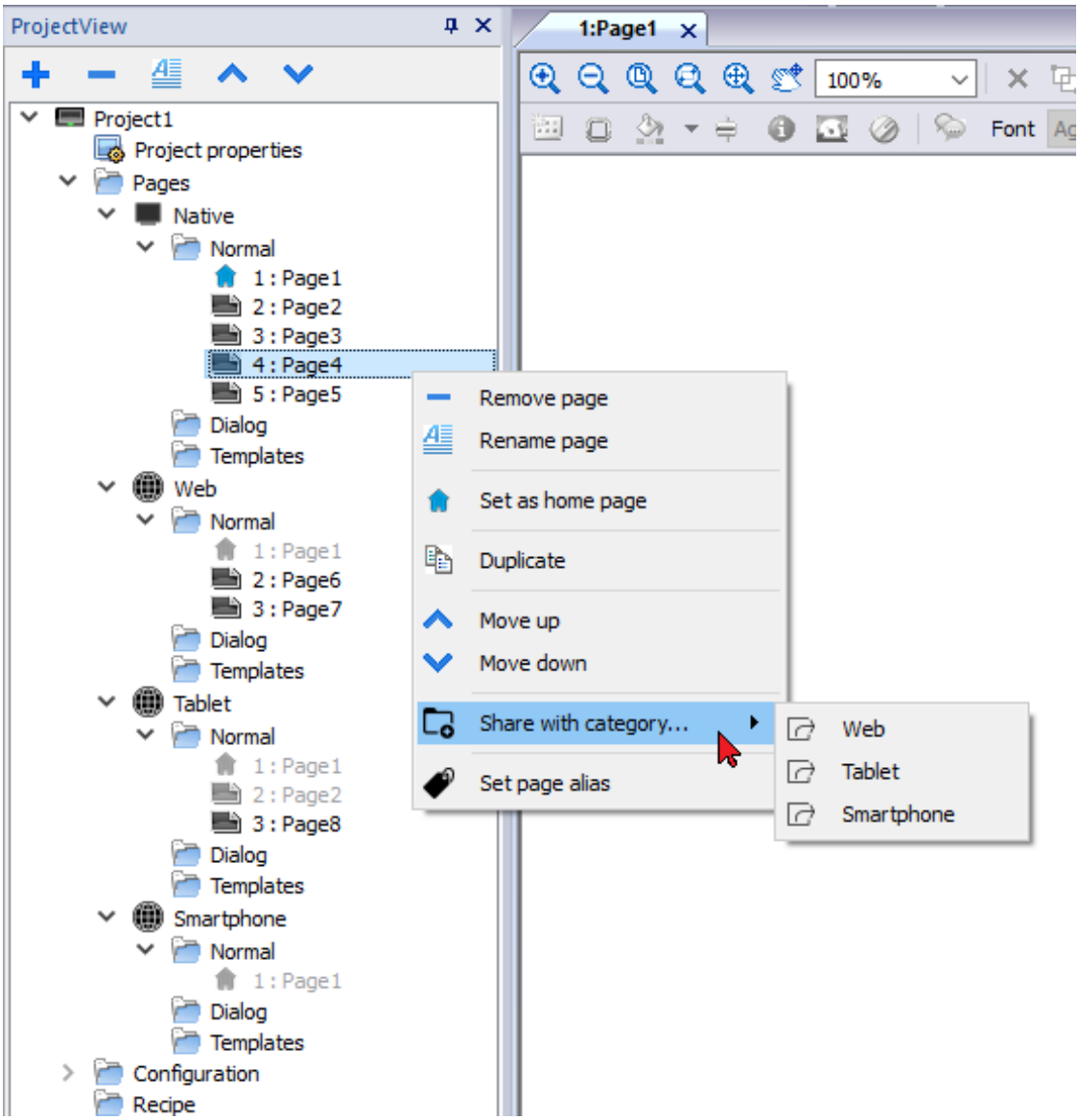
Property	Description
Technology	Identify the clients that can use these pages. It can be a combination of: • Local HMI Device • Remote LRH SW Client • Web Clients (PC, Tablet, Smart Phones, etc.)
User Agent	It is a regular expression that identifies the web browsers that can display the pages of the category. The user-agent of the web client has to match with this parameter. Example: .* Anything (all web clients) Android Only Android web clients Android|iPhone Only Android or iPhone web clients
Min Width **Min Height** **Max Width** **Max Height**	Defines the size of the display of the Web browser that has to show the pages of this category. The default, Min=0 and Max=-1, is meaning any size.

If the definition of a Web client belongs more than one category, are choices the pages that are available inside the closest category.



Shared pages

Pages can be shared between the categories. Shared pages are highlighted in gray color and can be opened indifferently from each category.



Home Page

From the context menu of the page is possible to define the Home page of the category. The Home Page is the first page that is displayed in the browser type defined in the category and defines the starting point for your web project. The pages you can access from the home page depend on how other pages are linked in the project.

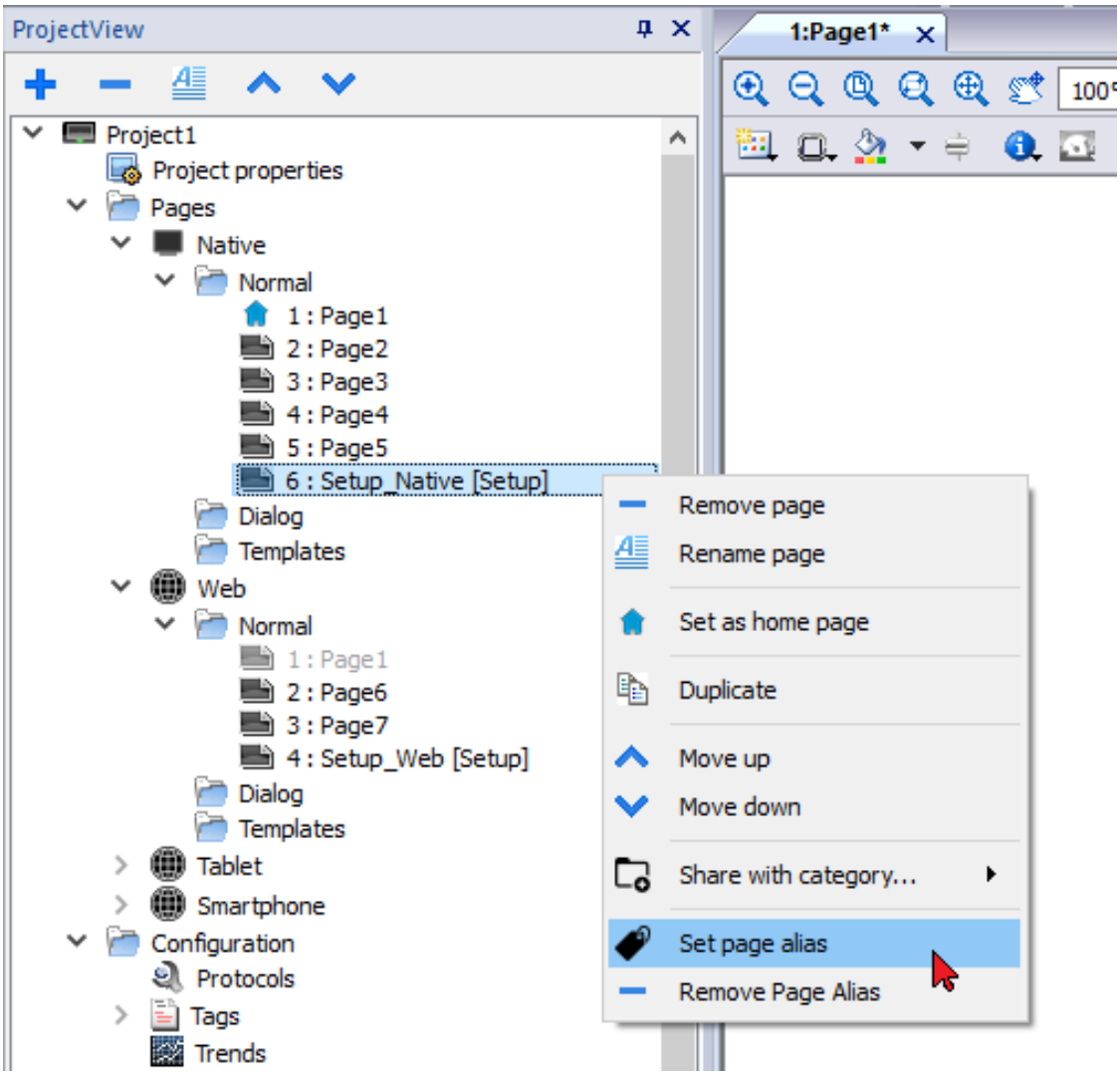
Alias pages

Using pages shared between categories could be useful the alias page parameter to load the appropriate customized page.

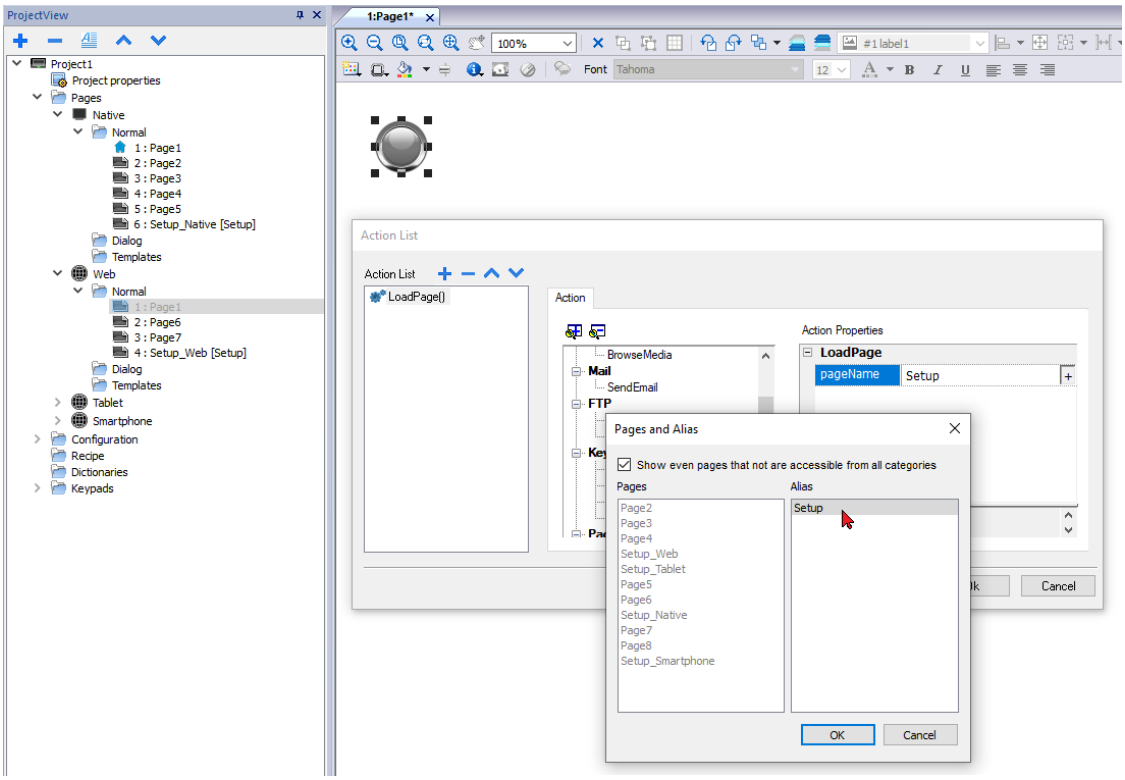
For example, you can have a shared "Page1" common to all categories. Page1 will be shown on both the HMI device and on Web Client, but from this page, you need to add a macro to load a customize setup page. This means a macro that load the page "Setup_Native" on HMI device or a different page "Setup_Web" on a web client.

To load a different page depending on the client used, you can add the same alias to both "Setup_Native" and "Setup_Web" pages and use the alias name in the LoadPage macro.

Set the alias page:



Use the alias page:

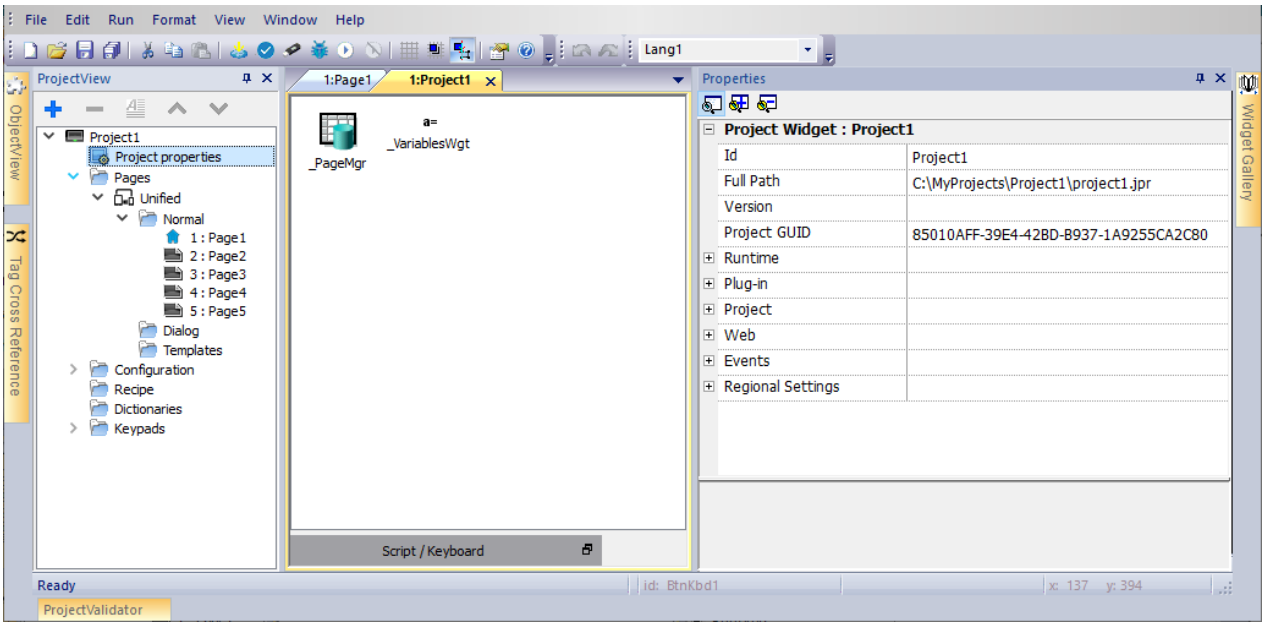

ProjectView
Project1
Project properties
Pages
Native
Normal
1 : Page1
2 : Page2
3 : Page3
4 : Page4
5 : Page5
6 : Setup_Native [Setup]
Dialog
Templates
Web
Normal
1 : Page1
2 : Page6
3 : Page7
4 : Setup_Web [Setup]
Dialog
Templates
Tablet
Smartphone
Configuration
Recipe
Dictionaries
Keypads
1:Page1*
100%
Font
Tahoma
Action List
Action List
LoadPage()
Action
BrowseMedia
Mail
SendEmail
FTP
Action Properties
LoadPage
pageName
Setup
Cancel
Pages and Alias
Show even pages that not are accessible from all categories
Pages
Alias
Page2
Page3
Page4
Setup_Web
Setup_Tablet
Page5
Page6
Setup_Native
Page7
Page8
Setup_Smartphone
Setup
OK
Cancel

6 Project properties

Project properties contain settings for the project.

Path: ***ProjectView****> double-click* ***Project properties****>* ***Properties*** *pane*

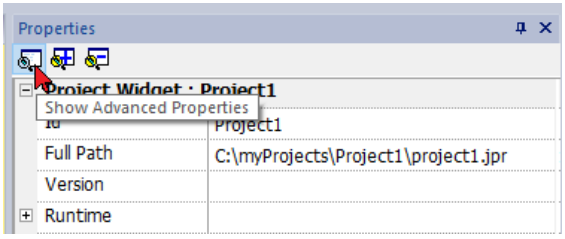
The project **Properties** pane contains a list of project level user-configurable data.



Basic and advanced properties

Some properties are displayed only in advanced mode. To view all project properties:

- Click **Show Advanced Properties** button to expand the property view in the **Properties** pane.



Available properties

Property	Description
Id	Project name (read only)
Full Path	Project path (read only)
Project GUID	Project unique identifier (read only)

Property	Description
Version	The Version field is available for users to report the project version
+ Runtime	Properties related with the application runtime. See "Runtime" belowfor details
+ Plug-In	Optional modules. See "Plug-in" on page 79for details
+ Project	Properties related with the project. See "Project" on page 80 for details
+ Web	Properties related with the web interface. See "Web" on page 84for details
+ Events	Global events. See "Events" on page 85for details
+ Regional Settings	Definition of date format, list separator, thousand and decimal symbol of number. See "Regional Settings" on page 86 for details

Project ID, Project GUI and Project Version are available from system variables. See "Default variables" on page 141 for details.

Runtime

Path: **ProjectView**> double-click **Project properties**> **Properties** pane

Property	Description
Context Menu	Define how context menu should appear in the HMI project. **on delay** = context menu appears touching/pressing and holding for a few seconds an empty area of the runtime screen, or via **Context menu** action **on action** = context menu appears only via **Context menu** action. See "Widget actions" on page 213 for details.
Developer Tool	Enable/disables a collection of runtime debugging utility tools.
Buzzer on Touch	Enables buzzer when touching a widget on HMI device screen. Supported widgets: • buttons • hotspots • needles • fields • external keys • combo boxes • tables items • control list items  *On Windows CE panels, available for from v1.76 ARM / 2.79 MIPS.*

Property	Description
Buzzer duration	Default 200 ms
Keyboard	Enables the use of keyboard macros at runtime when using external keyboards.
JavaScript Debug	Enables the JavaScript debugger at runtime for the current project.
Allow JS Remote Debugger	Enables JavaScript remote debugger for current project.  Remote debugging not supported on LRH SW Client.
Image DB enable	Activates an engine used by the Runtime to optimize project performance.  **WARNING: This property should only be disabled by technical support for debugging purposes since this might reduce performance at runtime.**
FreeType Font Rendering	Switches to FreeType the font rendering used by LRH SW and runtime.  The main reason for using the FreeType is that we need the same engine in all devices to avoid different rendering, in particular if static optimization is involved.
Communication icon delay (ms)	Delay before display the communication error icon (default is 0 mSec) The special value -1 is meaning always disabled
Fast Boot	When fast boot is enabled and the User Interface is started before the background server • Default: User Interface is loaded after the background server is ready to use • Fast UI: User Interface is loaded before loading the background server

Fast Boot

When fast boot is enabled, the HMI device will provide the welcome screen as fast as possible after the power up. In this mode, only the minimum necessary features are loaded before starting the User Interface. Loading of protocols, events, trends, alarms, actions are postponed after loading the User Interface.

There are two flags to set:

- The “Fast Boot” flag available inside the advanced project properties
- The “Fast Boot” flag available inside the Services page of the BSP System Settings tool (see "System Settings" on page 547)

When fast boot is enabled and the User Interface is started before the background server the JavaScript event project.onServerReady can be used to get server synchronization.

Example:

```
if (!project.serverIsReady) {
       // Set the callback to wait for server ready
       project.onServerReady = onServerReady;
```

```
} else {
        // Server is ready, call it now
        onServerReady();
}

function onServerReady()
{
        project.setTag("Tag1", 1);
        project.showMessage("Server is ready, tags can be used: " + project.getTag("Tag1")
)
}
```

This is an advance feature available only on Linux platforms

Developer tools

Collection of runtime debugging functions that can be enabled or disabled.

1. In **Properties** pane, set **Developer Tools** to **true**.
2. Download the project.
3. Open context menu.
4. Select **Developer tools**.

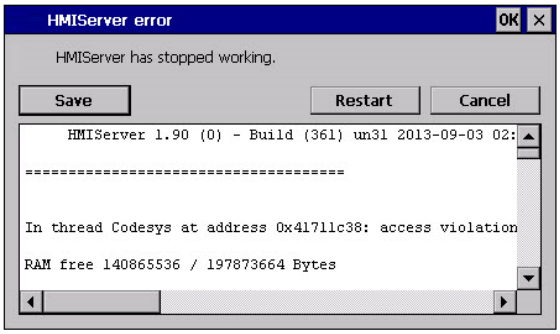
Developer tools list

Tool	Description
Show/Hide all	Shows a dialog containing information about device status like CPU load, memory usage, event queues.
CPU statistics	Shows information on CPU load. See "CPU Statistics" on the facing page.
Memory statistics	Shows information about system RAM . A negative value indicates that free memory is decreasing.
Event queues	Shows information on event queues (size, maximum achieved size, number of processed events, last and maximum processing time). Timing statistics are only available for non-UI queue.
Timelog summary	Show page loading time.
Embed window	Allows embedding in runtime the scene or leave the developer tool window as a standalone window (dialog).
Reset queue stats	Resets statistical information on event queues.
Disable watchdog	Disable the watchdog function and prevents system restart in case of freeze or crash of services.
Ignore exceptions	Disables crash report function, exceptions are not saved in the crash report window.
Profiling	Measures the time spent for loading/rendering the active page. See "Profiling" on the facing page

Watchdog

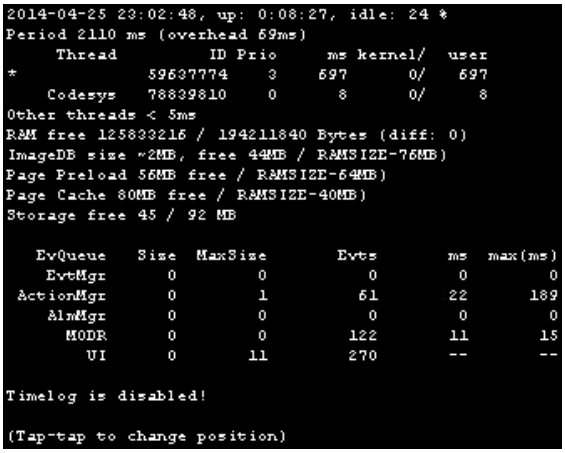
This feature allows you to disable the watchdog. This way you can avoid system restart in case of a runtime crash and have the time to save the crash report or check system status information (for example,memory available, CPU load, events queue size and so on).

The crash report dialog is displayed automatically in case of a system freeze or crash allowing users to save a log file of crash.



Important: Save this file for technical support.

CPU Statistics

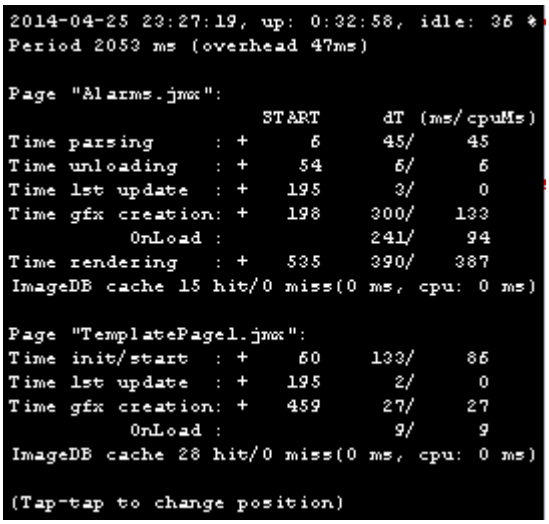


On the top row the current machine time is shown along with the total device uptime.

CPU statistics are collected with a frequency of 2000 milliseconds. The actual period and the overhead required to collect and visualize statistics are displayed as well. The more the actual period is far from the nominal 2000 milliseconds the higher is the system load. CPU consumption of threads is listed reporting the name of the thread (if available, main thread is marked with a *), the thread ID, the thread priority and CPU time spent during the 2000 milliseconds period, divided in user and kernel time.

Profiling

Profiling allows you to check time spent for loading/rendering the active page. Profiling will start from the next page load and will be active only for the first painting of the page to the screen (the configuration is retained).



Profiling option	Description
Enable timelog	Enable timelog capture. Timing will be visible inside the “Timelog summary” window.
Save timelog to file	Saves a report of profile details and the time spent loading a project and its pages into a timelog.txt file. This file can be exported and shared for further analysis. **Important: The execution of this function may reduce page change performance.**
Overlay OnLoad times **Overlay Rendering times**	This view allows displaying time spent on single widgets and is available only for the rendering and OnLoad steps. The view gives an immediate feeling of where time is spent. Red zones represent the most time critical zones. Detailed widget times are visualized by a tooltip window. In case of out-of-the-scene widgets some arrows allow to navigate to these areas and hovering on them the tooltip will show the area summary
Select overlay color	Select the overlay color to use

Timelog data

Data	Description
Time parsing	Time spent parsing current page. Depends on page complexity/number of widgets.
Time gfx creation	Time spent for image rendering. Mainly related to the *Onload* method.
Time rendering	Time spent rendering the page.
Time unloading	Time spent unloading the page, if current page depends from another page.

Times are provided in couples: wall time/CPU time. Wall time is the absolute time required by this part which can be higher than the actual CPU time required since higher priority threads are also running (for instance protocols). The start time column refers to the page load start time. It can be used to track the actual time required to load a page, since partial times

only refer to the most time critical functions and do not include other times that often contribute significantly to the total time.

For example, the actual total wall time required to load a page is rendering (which is the last step) start time + rendering wall time.

FreeType font rendering

New projects use the FreeType font engine as default. Projects created with older versions of LRH SW could use an older font engine also after project conversion to avoid any backward compatibility issue.

Switch to FreeType whenever possible for better page rendering.

Once you have switched to the new font rendering, save the project and verify that all texts are displayed correctly in all project pages.

Font rendering issues

When switching to the FreeType font engine a project created with the older font engine, you may experience the following problems:

- text requires more/less pixels for rendering thus changing text layout
- widgets are resized to accommodate text
- better rendering can be obtained using antialiasing (antialiasing is a text widget property)

Plug-in

You can choose which software modules are downloaded to the runtime with the project. Software plug-in has been designed to reduce memory requirements for the HMI application in HMI devices where storage is limited.

Path: **ProjectView**> double-click **Project properties**> **Properties** pane

Property	Description
WebKit	Module required by WebBrowser widget
TextEditor	Module required by TextEditor widget
PDF Reader	PDF Reader
VNC Server	VNC Server

Note: Not all software plug-in modules are compatible with all HMI device platform.

Once enabled, software plug-in become part of the runtime. Use LRH SW to install it using one of the following procedures:

- install Runtime/update Runtime
- update package

To remove plug-ins from runtime use one of the following functions in System Mode:

- format flash
- restore factory settings

Important: The system cannot detect automatically which software plug-ins are required by the HMI application, make sure you select them all in the Project Properties.

Project

These properties define various elements of page behavior.

Path: **ProjectView**> double-click **Project properties**> **Properties** pane

Property	Description
Home Page	The first page loaded at runtime (after log-in page if security is enabled in project). When security is enabled, you can specify a different homepage for each groups of users. In this case this setting is ignored. See "User management and passwords" on page 325 for details.
PageWidth **PageHeight**	Defines the default size in pixel of an HMI page. Default is the display resolution of the HMI device model selected when creating the project.
Display Mode	Defines HMI device orientation.
Project Type	Defines HMI device type for the project. According to the model, some project features and properties are automatically adjusted. **WARNING: Starting from v2, the LRH SW HMI Runtime will check if the selected project type is matching with the HMI device model and will advise with a message when the selected type is not matching: "HMI Type mismatch. Convert project and download again."**
Panel Memory	Size of the available internal panel memory.
PageRequest **CurrentPage** **SyncOptions**	You can synchronize pages shown on the LRH SW HMI Runtime and LRH SW Client from a controller such as a PLC. Attached tag must contain an integer value within the range of the available project pages and must be available at least as a Read resource. See the "Web" on page 84 for the Web Browser support
Hold Time **Autorepeat Time**	Defines the values for hold time and auto repeat time for buttons and external keyboards. Note: These properties can be redefined for each button or key in their widget property table.
Hide Project Loading at boot	When hidden, the splash screen stay on the screen until the application is ready to run.
Target Zoom Factor	It is the zoom factor of the HMI device that will be applied when project is loaded at runtime.

<table>
<tr><th>Property</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><td>Range</td><td>0.3–2.9</td></tr>
<tr><td>Fit to screen</td><td>-1 = Fit to screen size
  Fit to screen maintains the aspect ratio. It find the scaling factor, i.e. scale for width and height, then take the smallest.</td></tr>
<tr><td>Default value</td><td>1 = no zoom</td></tr>
</table>
</td></tr>
<tr><td>Background color option</td><td>When the defined page is smaller of the entire display area, colorize the area that is not covered from the page (for example when page is Zoom Out)
<table>
<tr><td>None</td><td>Old mode, color is white (default)</td></tr>
<tr><td>Selected color</td><td>Color to use</td></tr>
<tr><td>Page background</td><td>Auto adjust color based on background of template or of page</td></tr>
</table>
</td></tr>
<tr><td>Signature</td><td>Algorithm to use to signing
sha256
sha1</td></tr>
<tr><td>Gesture Passthru Enabled</td><td>Enable the possibility to pass gesture events to underlying widgets after a configurable delay. User has to keep pressed the finger and then execute the gesture.</td></tr>
<tr><td>Gesture Passthru Delay (ms)</td><td>When enabled, the gesture events are passed to underlying widgets after this delay (see "Gesture events pass thru" on page 415 for details)</td></tr>
<tr><td>Gesture Multitouch</td><td>Enable multi touch gestures
false
true (default)  This property give the possibility to disable the multi touch gestures. This could be useful to avoid problems with old projects that were not designed to manage the multi touch gestures.</td></tr>
</table>

Property	Description
On Access Denied	When user try to use a widget that is locked from the security configuration to read-only (e.g. a field or a button), a padlock icon is shown for a couple of seconds to highlight that the widget is not accessible. • None • Show Icon
ComboBox View Mode	Select the visualization mode of all the Combo Box widgets of the project (see "Combo Box widget “full screen” mode with images" on page 404 for details) • Context Classic view with drop-down menus • Full screen Enhanced view with configurable texts and images that will pop up in the middle of the screen for easy scroll and selection.

PageRequest, CurrentPage and SyncOptions

It is possible to have LRH SW HMI Runtime exchange devices information on the page shown by the HMI. You can synchronize pages shown on the HMI device and on LRH SW Client or to control an HMI project from a controller such as a PLC.

The following properties can be customized:

Property	Description
PageRequest	Page to be shown on the HMI device and on LRH SW Client. Attached tag must contain an integer value within the range of the available project pages and must be available at least as a Read resource.
CurrentPage	Page number displayed on the HMI device or on LRH SW Client or on both. Attached tag must be available at least as a Write resource and must have integer data type.
SyncOptions	Synchronization of project pages with the value contained into the **CurrentPage** property. Options can be: • **disable**: page number value is ignored, • **local**: page number displayed on HMI, • **remote** : page number displayed on LRH SW Client. • **local + remote**: page number displayed on HMI and on LRH SW Client, if different pages are displayed the last page loaded is considered.

Example: forced page change from controller/PLC to HMI device and LRH SW Client

Set properties as follows:

PageRequest	attached to tag "A"
CurrentPage	empty
SyncOptions	disable

Set value of tag "A" to display the requested page on HMI device and LRH SW Client.

Example: forced page change from controller/PLC to HMI and LRH SW Client. Read current page loaded on HMI

Set properties as follows:

PageRequest	attached to tag "A"
CurrentPage	attached to a tag "B" as read/write
SyncOptions	local

Set value of tag "A" to display the requested page on HMI device and LRH SW Client. Tag "B" will contain the number of page currently shown by the device.

Example: forced page change from controller/PLC to HMI device and LRH SW Client. Read current page loaded on LRH SW Client.

Set properties as follows:

PageRequest	attached to tag "A"
CurrentPage	attached to a tag "B" as read/write
SyncOptions	remote

Set value of tag "A" to display the requested page on HMI and LRH SW Client. Tag "B" will contain the number of page currently shown by LRH SW Client.

Example: forced page change from controller/PLC to HMI device and LRH SW Client. Force LRH SW Client page synchronization with HMI device (not vice versa).

Set properties as follows:

PageRequest	attached to a tag "A" as Read/Write
CurrentPage	attached to the same tag "A" as per **PageRequest**
SyncOptions	local

Set value of tag "A" to display the requested page on HMI and LRH SW Client. Change page on HMI to display the same page on LRH SW Client.

Example: forced page change from controller/PLC to HMI device and LRH SW Client. Force HMI page synchronization with LRH SW Client (not vice-versa).

Set properties as follows:

PageRequest	attached to a tag "A" as read/write
CurrentPage	attached to the same tag "A" as per **PageRequest**
SyncOptions	remote

Change value of tag "A" to display the requested page on HMI and LRH SW Client. Change page on LRH SW Client to display the same page on HMI.

Example: synchronize displayed page between HMI device and on LRH SW Client

Set properties as follows:

PageRequest	attached to a tag "A" as read/write
CurrentPage	attached to the same tag "A" as per **PageRequest**
SyncOptions	local+remote

Changing page on HMI device, same page will be shown on LRH SW Client and vice-versa.

Web

Path: **ProjectView**> double-click **Project properties**> **Properties** pane

Property	Description
Web Inactivity Timeout	Defines a timeout for LRH SW client. When the timeout expires without any activity the current user is logged out. **Range**: 1–86400 s (form 1 s to 24 h) **Default value**: 600 s **Values**: 0 = disabled
Web Icon	The favorite icon associate at the web pages
Refresh Time	Defines the refresh time for the communication between the runtime and LRH SW clients. **Range**: 500–10000 ms **Default value**: 3000 ms
Browser Optimization	**true**: Web engine optimization enable (default) **false**: Web engine optimization disable (useful for old browsers that not support the web engine optimization)
Enable Global JavaScript for remote	Define if the JavaScrip code defined inside the Project Properties, general triggered from Alarms and Schedulers events, have to run only on local HMI device or even on remote

Property	Description
	clients. **None** — Will not be executed on remote clients (run only inside the local HMI device) **Client** — Will be executed on LRH SW Client **Web** — Will be executed on Web client **Both** — Will be executed on both LRH SW Client and Web clients
Max Bandwidth (Kbs)	Limit for maximum data sent by server (useful for old slow browsers). Set to 0 to use all the available bandwidth (default)
Web clients connection mode	**Auto** — The connection mode is selected by the client (default) **SSE** — Force the Server-Sent Events mode **Long Polling** — Force the Long-polling mode
WebPageRequest	You can synchronize pages shown on the LRH SW Clients from a controller such as a PLC. Page to be shown on the LRH SW Client. Attached tag must contain an integer value within the range of the available project pages and must be available at least as a Read resource.
Web Communication icon delay (ms)	Delay before display the communication error icon (default is 0 mSec) The special value -1 is meaning always disabled

The project.getClientType() can be used to retrieve the running client type. See "Project object" on page 487 inside JavaScript chapter for additional details.

Events

Path: **ProjectView**> double-click **Project properties**> **Properties** pane

Property	Description
OnWheel	Used only in conjunction with wheel input devices. Normally the wheel is used to increase/decrease the value of a tag without an external keyboard device. Attach this property to a change of wheel event and use an action like **BiStep** to increase/decrease a tag value. **The project's OnWheel Action is executed only when the OnWheel Action will not overwritten from the loaded page.**

Regional Settings

Path: **ProjectView**> double-click **Project properties**> **Properties** pane

Property	Description
Short date format	The date format to use when user select SHORT-DATE in the date format of the widget
Long date format	The date format to use when user select LONG-DATE in the date format of the widget
List separator	List separator character to use inside the dumped files.
Decimal symbol	Character to use in numeric widgets to separate the integer part from the fractional part (it is visible only when user configure the widget to show the fractional part)
Thousand symbol	Character to use in numeric widgets to separate the thousands (it is visible only when user configure the widget to show the thousand character)

You can use placeholders to freely define the Time and Date format (see "Time and Date placeholders" on page 413)

7 The HMI simulator

HMI simulator allows you testing projects before downloading it to the HMI device. It may be used to test the project when no HMI device is available and to speed up development and debugging activities.

The HMI simulator supports:

- online simulation - in communication with real devices (only for protocols with Ethernet or RS-232 communication),
- offline simulation - simulating tag behavior

The data simulation method is set in the **Simulator** column of the Tag Editor.

Data simulation methods

Set tag simulation behavior in the **Simulator** field of Tag Editor.

Method	Description
Variables	Data is stored in a simulator variable. This variable holds the value of the tag so you can read and write the value.
SawTooth	A count value is incremented from **Offset** to **Amplitude + Offset** value with a **Period** of 60..3600 seconds. When the counter reaches **Amplitude + Offset**, the value is reset to **Offset** and the counter restarts.
Sine Wave	A sine wave value is generated and written to the tag value. **Min, Max** and **Period** values can be defined for each tag.
Triangle Wave	A triangle wave value is generated and written to the tag value. **Min, Max** and **Period** values can be defined for each tag.
Square Wave	A square wave value is generated and written to the tag value. **Min, Max** and **Period** values can be defined for each tag.

See "Adding tags" on page 101 for details.

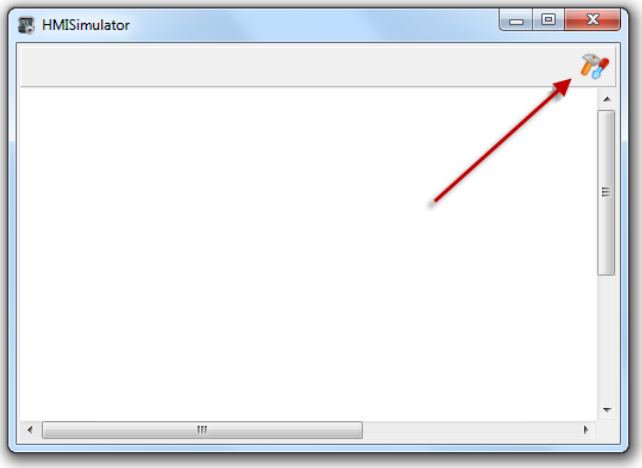
Simulator settings

The Simulator works by default with simulated protocols. It can also work with real protocols (Ethernet or serial protocols)

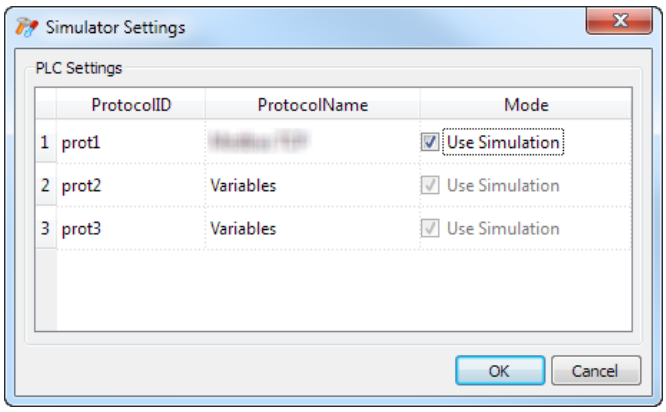
Note: For protocols not supporting communication with external devices, such as the Variables protocol, this option is always disabled.

Changing simulated protocols

1. Click the simulator **Settings** icon.



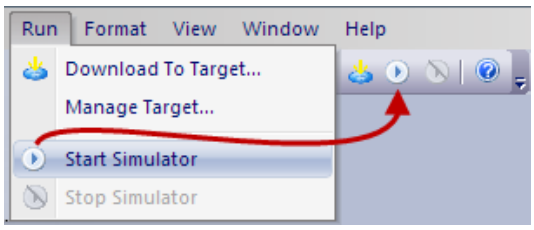
2. Select **Use Simulation** to use simulated protocols, otherwise real protocols will be used for communication with external devices.



Launching and stopping the simulator

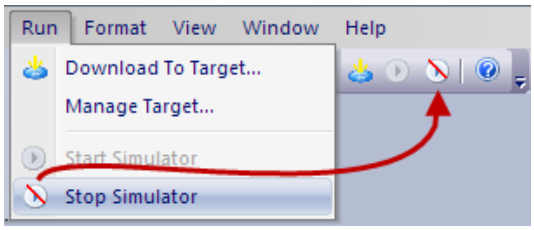
To launch the simulator:

1. On the **Run** menu, click **Start Simulator:** the Simulator runs on the computer in the same way as the server would run on the HMI device.



To stop the simulator:

1. On the **Run** menu, click **Stop Simulator** or on the simulated page double-click the **Exit** button.



8 Transferring the project to HMI device

To transfer the LRH SW project to the target HMI device you can use:

- function **Run > Download to Target**
- function **Run > Update Package** with the use of a USB device

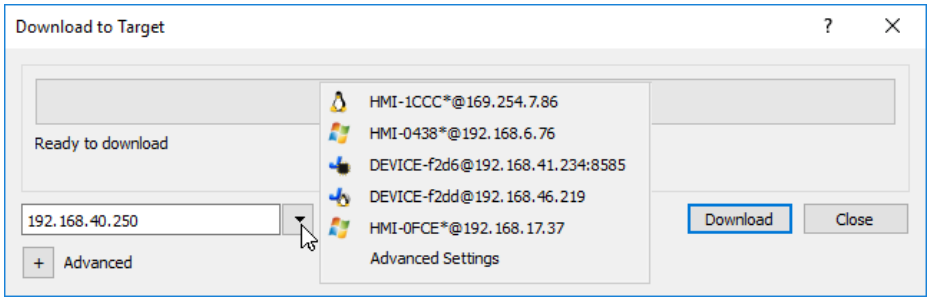
Download to HMI device

Path: ***Run*> *Download to Target***

This function transfers project and LRH SW HMI Runtime via Ethernet .

Note: The HMI device must have a valid IP address. See "HMI device basic settings" on page 8 for details on how to assign an IP address.

1. Click the discovery button: a list of the detected IP addresses is displayed.
2. Select the HMI device IP address.

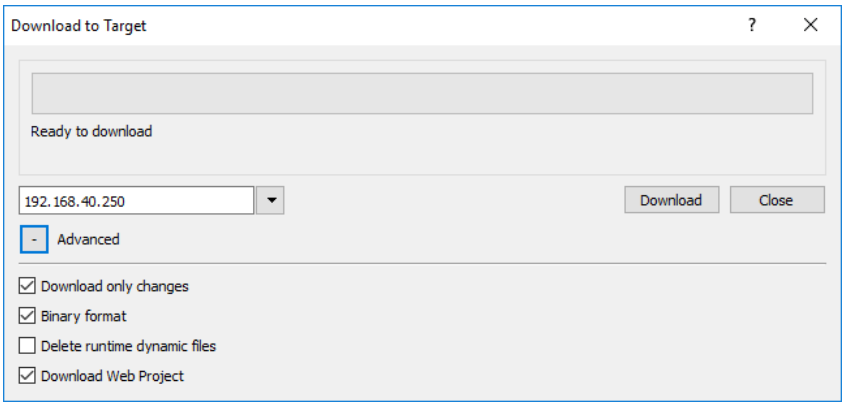


You can even enter the IP address manually or, if available, the host name provided by a DNS server. Using a service tool like Bonjour, HMI devices can be discovered using their hostname (e.g HMI-0d37.local). Bonjour is a trademark of Apple inc.

3. Click **Download**: LRH SW will switch the HMI device to Configuration Mode and transfer the files.

When the download operation is completed, the HMI device automatically switched back to Operation Mode and the project is started.

Advanced options



Option	Description
Download only changes	Transfers to the HMI device only the modified project files.
Binary format	Download files using binary format.

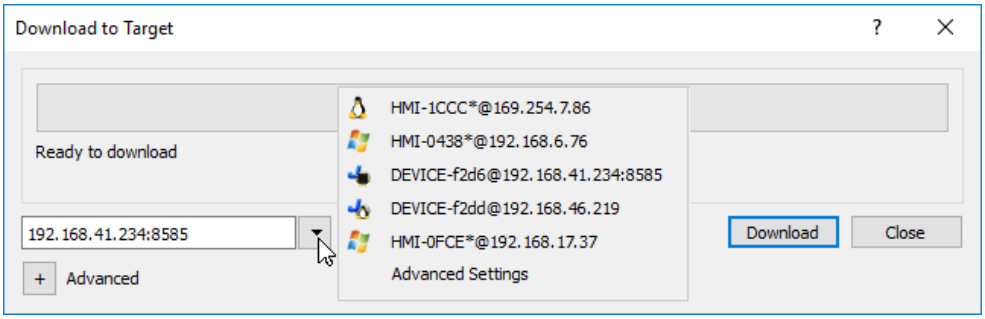
Option	Description
Delete runtime dynamic files	Modified configuration of recipes, users, schedulers, etc. done at runtime will be deleted and overwritten by the configuration defined in the project. ***CAUTION: This operation cannot be undone, deleted dynamic files cannot be restored.*** ***CAUTION: Dynamic files are not deleted if stored on external devices (USB or SD Cards).***
Download Web Project	Download the LRH SW pages to HMI device.

When transferring a project, LRH SW uses a combination of HTTP and FTP connections:

- HTTP connection - issues the commands to switch to transfer mode or to unload running project,
- FTP session - transfers the files to the flash memory in the HMI device.

Advanced Settings

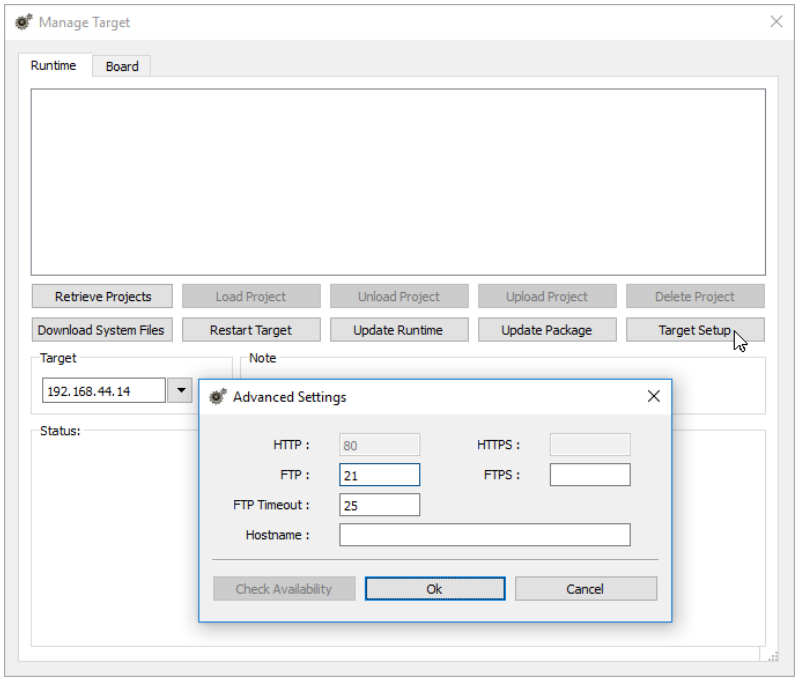
Using the “Advanced Settings” option, you can define the ports to use, but generally, you do not need to enter this information because HMI devices will provide the ports to use inside the panesl list.



Changing HMI device connection settings

Path: ***Run> Manage Target***

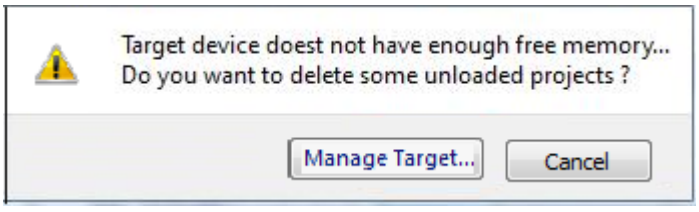
1. Click **Target Setup**: the **Advanced Settings** dialog is displayed. Default port for HTTP connections on the HMI device is port 80.



2. Set correct HTTP, FTP or HTTPS, FTPS ports for the HMI device. (These are the ports used by the system to connect to the HMI device and may need to be modified when default ports are used by other services or applications or if the local network requires specific settings.)
3. Specify **Hostname** to easily identify each device in a network where multiple devices are available. The default hostname is “HMI” for all devices.
4. Click **Download System Files**. At the next download the new ports will be used in the HMI device and new hostname will appear in the drop-down list

Managing big projects

For successful download the project size should be at least 2 MB smaller than the available memory. If not, you run out of flash memory in the HMI device and a warning message is displayed.



To free more memory:

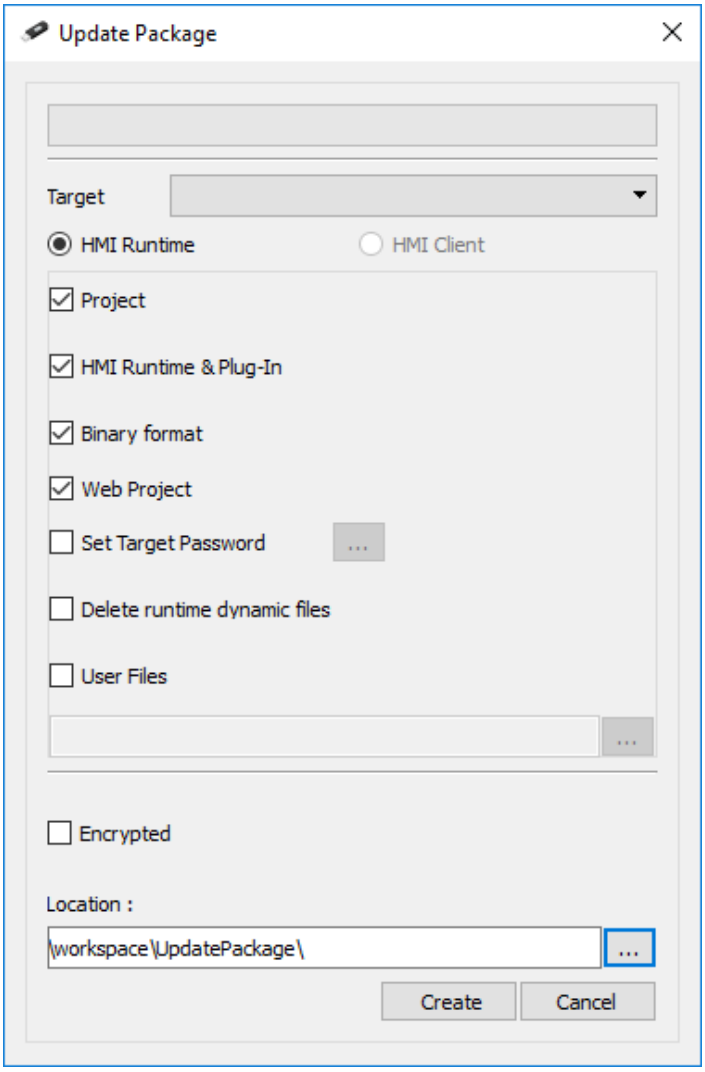
1. Click **Manage Target**.
2. Delete the projects you no longer need t to make more memory available.

Update package

The Update Package create a UpdatePackage.zip file to install or update the application inside the HMI device using an USB memory key.

Creating an update package

Path: ***Run*** *>* ***Update Package***



Option	Description
Target	HMI device type. Selected automatically if the project is open.
Application Selector	Select the application to insert inside the UpdatePackage.zip • HMI Runtime • HMI Client (Available only on Linux devices)
Project	Adds open project to update package.

Option	Description
HMI Runtime & Plug-In	HMI Runtime is added to the update package. If the project is open the required plugins are also added to update package.
Binary Format	Download files using binary format.
Web Project	Download the LRH SW pages to HMI device.
Set Target Password	Sets password to perform critical tasks (for example, project download/upload , board management) See "Protecting access to HMI devices" on page 527.
Delete runtime dynamic files	When checked, all dynamic files will be deleted and the FRAM will be clean up.
User Files	Selects files to be copied to the QTHM folder of HMI device. Max size 5 MB
Encrypted	Enables encryption of update package so that it can only be unzipped by the HMI Runtime.
Location	Location of update package.

Important: When create a package with the HMI Runtime application, always include both project and the runtime. If you need to use an old project with the latest Runtime version, convert the project first. See "Installing the application" on page 2 for details.

Example of user's file location

Computer:

C:\Users\Username\Desktop\myFolder

- *subFolder1/file1*
- *subFolder1/file2*
- *file3*
- *file4*

Linux devices:

/mnt/data/hmi/qthmi

- *subFolder1/file1*
- *subFolder1/file2*
- *file3*
- *file4*

Loading an update package

*Path: from the context menu > **Update***

1. Assuming you have stored the package in the root folder of a USB drive, remove the drive from the computer, plug it in the HMI device, display the context menu by holding your finger for a few seconds on the screen and select **Update**.
2. The system will check for the presence of the update package in the USB drive root and ask confirmation to proceed with the update.

3. Select **Auto select best match** and click **Next**: the procedure is completed automatically. Alternatively use the browser button to select the file to use.

The Runtime loader

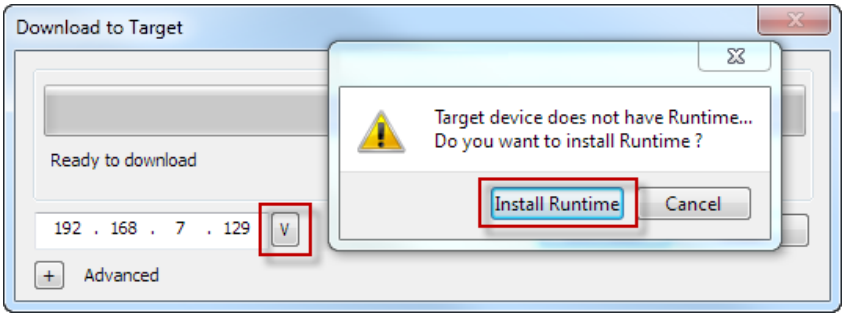
HMI devices are delivered from factory without Runtime.

When you power up the device for the first time, the Runtime Loader window is displayed (see "System Settings" on page 543for details)

The Runtime Loader presence depends on the device Operating System and may not be available on all the units. Old versions of HMI devices may not include the Runtime Loader. Contact technical support if you need further information.

Installing Runtime from LRH SW

When you download a project the Runtime is automatically installed if needed.



See "Transferring the project to HMI device" on page 91 for details.

1. Click **Install Runtime**: the procedure is run automatically.

Installing Runtime from a USB drive

1. Prepare the Update Package as described in "Update package" on page 95
2. Plug the USB drive in the device and follow the instructions for the type of device (see "System Settings" on page 543for details)

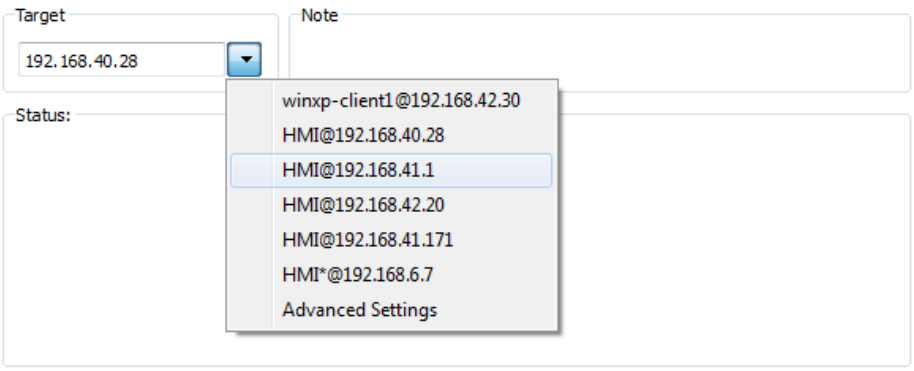
Note: Old versions of HMI devices may not support automatic installation of Runtime. Contact technical support for more information.

Upload projects

Path: ***Run*** *>* ***Manage Target***

You can copy a project from the Runtime to the computer where LRH SW is running.

1. In the **Runtime** tab, select the IP address of the device from the drop-down list **Target**.



2. Click **Retrieve Projects**: a list of all the projects available is displayed.
3. Select project to upload
4. Click **Upload Project**

Upload could be password protected. See "Protecting access to HMI devices" on page 527 for details.

5. If required, enter password. The upload process starts.

A copy of the project is saved in:

C:\Users\username\Documents\LRH SW\workspace\Uploaded\RuntimeIPAddress\workspace\ProjectName

Note: If the upload operation fails, check firewall settings the computer where LRH SW is running.

9 Tag editor

A tag is a friendly name used to identify the memory location of a device. Tags can be read or write from an external device through communication protocols.

From the Tags Editor, you can configure the protocols and the list of tags to use.

Communication protocols

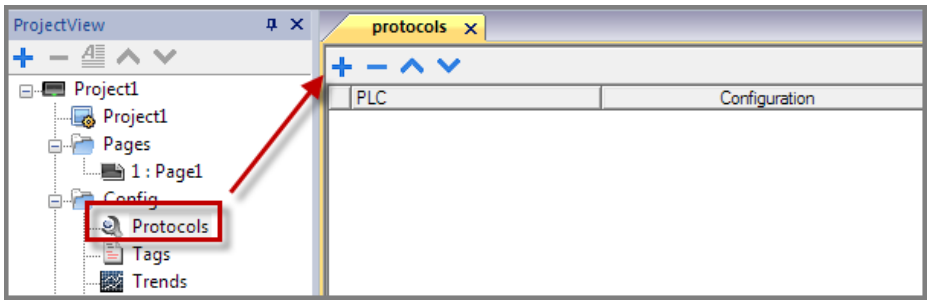
Path: ***ProjectView*** *>* ***Config*** *>* ***Protocols***

Device communication drivers are configured in the **Protocol Editor**.You can add up to the maximum number of protocols as specified in Table of functions and limits. Variable and System Variables are not counted as protocols.

Note: you can run different Ethernet protocols over the same physical Ethernet port, but you cannot run different serial protocols using the same serial port. Some serial protocols support access to multiple controllers, but this option is set within the protocol itself which is still counted as one protocol.

Adding a protocol

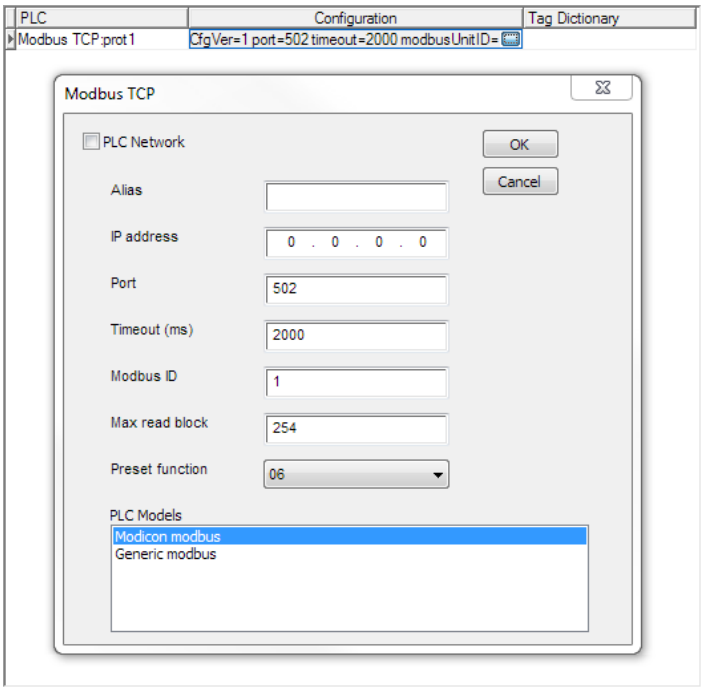
1. Click **+**.



2. Select the protocol from the **PLC** list and enter the required values.

Changing protocol settings

To change configuration parameters, click the browse button in the **Configuration** column.



Protocol parameters

Click **Show Advanced Properties** icon to see all parameters.

Parameter	Description
Dictionaries	Tags imported for the protocol. See "Importing tags" on page 104 for details.
Enable Offline AlgorithmOffline Retry Timeout	See "Automatic offline node detection" on page 291 for details.
Version	Protocol version available in LRH SW for selected HMI device.

Adding tags

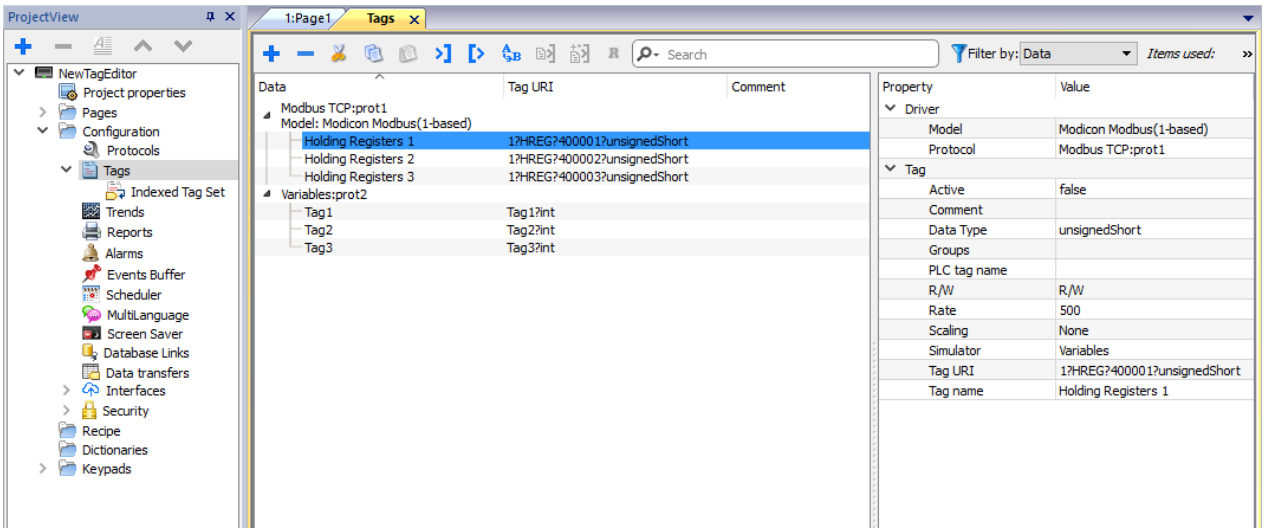
LRH SW uses tag names to access all device data. All fields and reference locations in the device need to be assigned a tag name to be used in the HMI project.

Tag Editor can be used to create and manage tags. After the tags have been defined, they can be used in the project by attaching them to widgets' properties.

See ""Attach to" parameters" on page 39 for details.

Tag editor

*Path: **ProjectView** > **Tags***



Adding a tag

1. Click **+** and enter the required data.
2. Select the Address from the communication protocol address dialog
3. Click on the fields that are inside the property dialog if something is to change (e.g. tag name)

Note that if a tag is selected, the add tag command **+** will create a new tag using the property of the selected tag.

Tag properties

Some properties depend from the protocol used. See specific protocol documentation for details.

Property	Description
Active	Update mode. • **false** = tags are read from controller only when required by the HMI device. • **true** = tags are continuously read even if not required by the displayed page. **Important: Leave this value set to false for higher communication performance.**
Description	Tag description
Encoding	Encoding type for string data type (UTF-8, Latin1, UTF-2 and UTF-16)
Groups	Group names associated to a tag
PLC tag name	Original PLC tag name, used to match tags used by HMI application (Tag Name) and tags exported from PLC
R/W	R/W tag attribute (R/W, R or W). Note: The content of Write Only tags is always written and never read. When communication is not active, the content of these tags may not be available in widgets.
Rate	Tag refresh time. Default: 500ms. When the refresh rate is set to “Manual”, the HMI device will not read the tag from the remote device automatically in background. Tag is read and refreshed into the database only by explicitly required from the “ForceReadTag” action or using the forceRefresh option into the JavaScript getTag(). **WARNING: Tags refresh rate is the maximum refresh rate. Actual refresh rate depends on: communication type (serial, fieldbus, Ethernet), protocol, amount of data exchanged.**
Scaling	Conversion applied to tag before database storage. • **By Formula** = defined as a linear transformation. • **By Range** = defined as a range conversion. • **Fixed Point** = fixed point scaling
Simulator	Tag behavior during simulation. Several profiles are available.
Tag address	Controller memory address. To edit click on the right side of the column to get the dialog box where you can enter the address information.
Tag name	Unique tag name at project level. Primary key to identify information in the runtime tag database. **WARNING: Duplicate tag names are not allowed.**

Note that is allowed to select multiple tags in Tag Editor and to change the same property to all (e.g. to change refresh time in 10 tags to 500 without change it in all tags one by one).

Managing tag names

Tag names must be unique at project level. If the same tags, from the same symbol file have to be used for two different controllers, use the “Alias” feature to add a prefix to the imported tags and make them unique at project level.

Note: Not all protocols support the “Alias” feature.

Managing tag groups

Tags used in each page are identified as part of a group, so that requests made by the communication protocol to the connected controller(s) can be processed faster: only the tags included in the displayed page are polled from the controller.

Scaling

Using the tag scaling function it is possible to resize the tag values that will be visible from the HMI application.

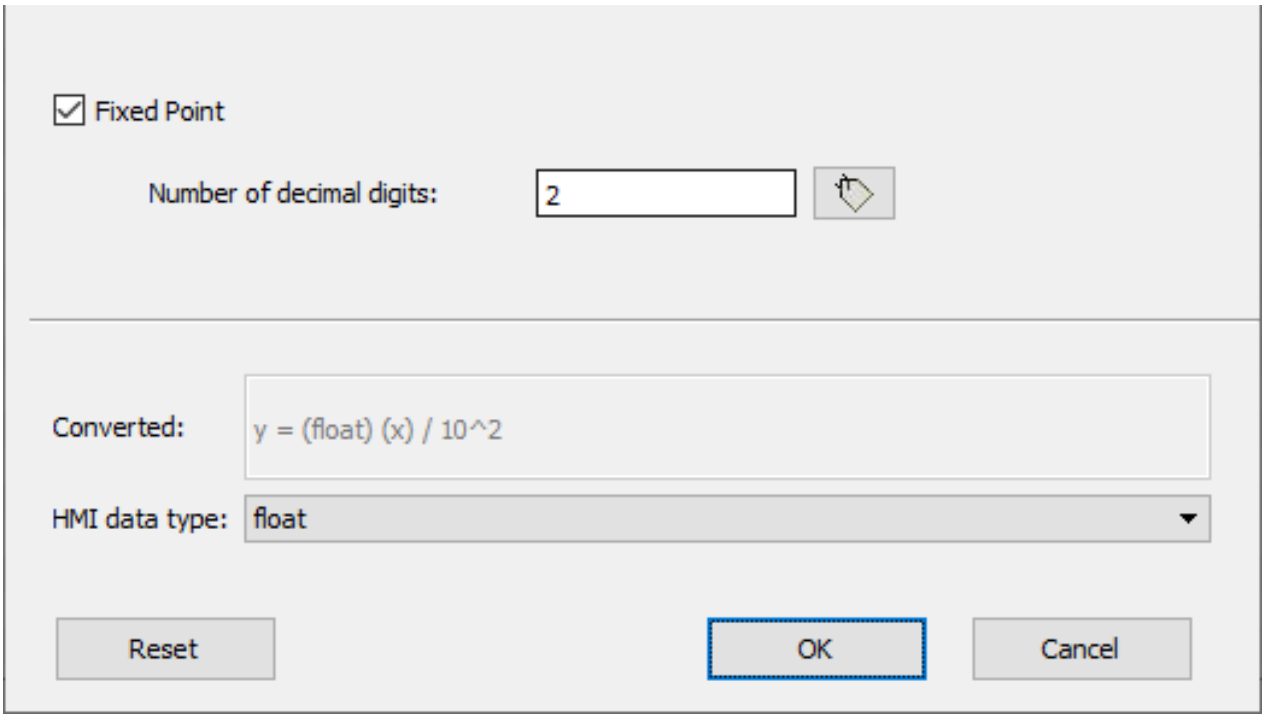
There is the possibility to configure

- Linear transformation, using the "By Formula" or the "By Range" mode
- Fixed Point transformation

Generally, the data type used inside the HMI is the same data type inherited from the PLC device. When a transformation is used, considerate the possibility to change the HMI's data type to not lose precision.

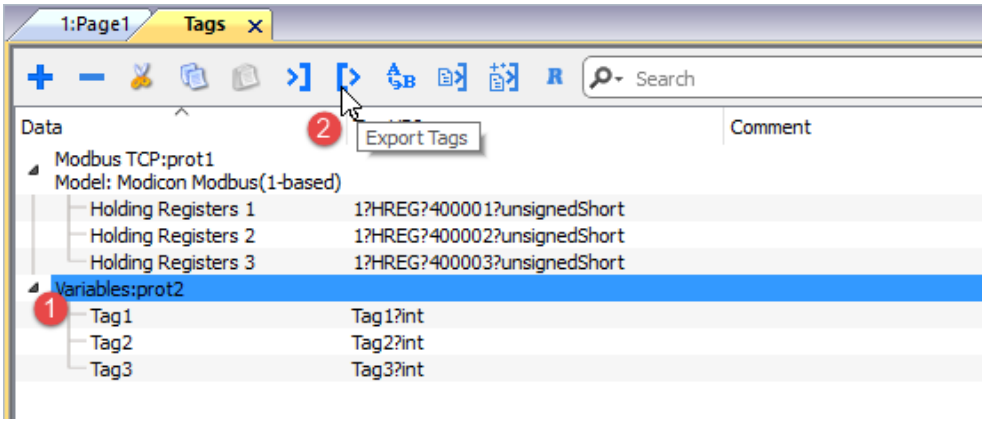
Example

If your PLC manages value with two decimal digits using an integer in fixed point, you can configure the scaling transformation as the below picture where the value read from the PLC will be divided by 100 and stored inside a float data type. E.g. PLC integer value 12345 will become the float value 123.45 inside the HMI device.



Exporting tags

Path: ***ProjectView*** *>* ***Tags***



1. Select the protocol for the tags you want to export.
2. Click the **Export Tags** button: all the tags configurations for the selected protocols are exported into an .xml file.

You can edit the resulting .xml file using third part tools (for example, Microsoft Excel) and then re-import the modified file (see "Importing tags" below for details).

Importing tags

Introduction

Some protocols allow you to import tags stored in a comma separated file (.csv or other formats).

Importing is a two step process:

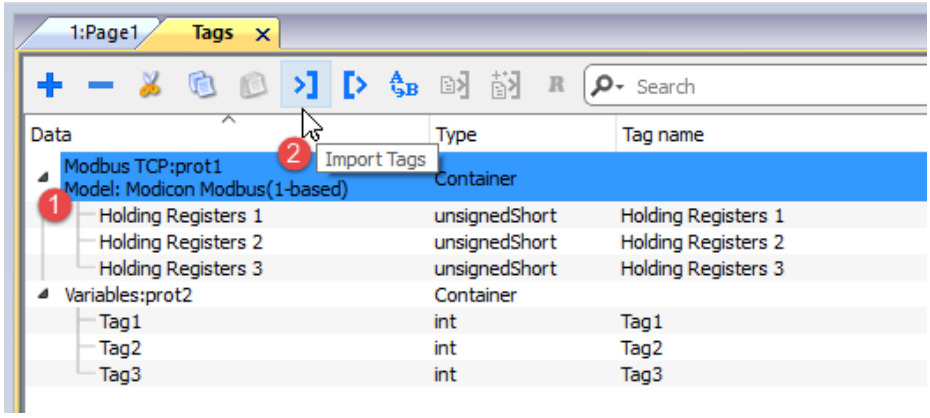
1. Import of the tag definition into a dictionary
2. Import tags from the dictionary to the project

WARNING: Special characters in tag names, such as "&" character, that can cause communication errors will be substituted with the underscore "_" character when imported. See "Limitations in Unicode support" on page 299

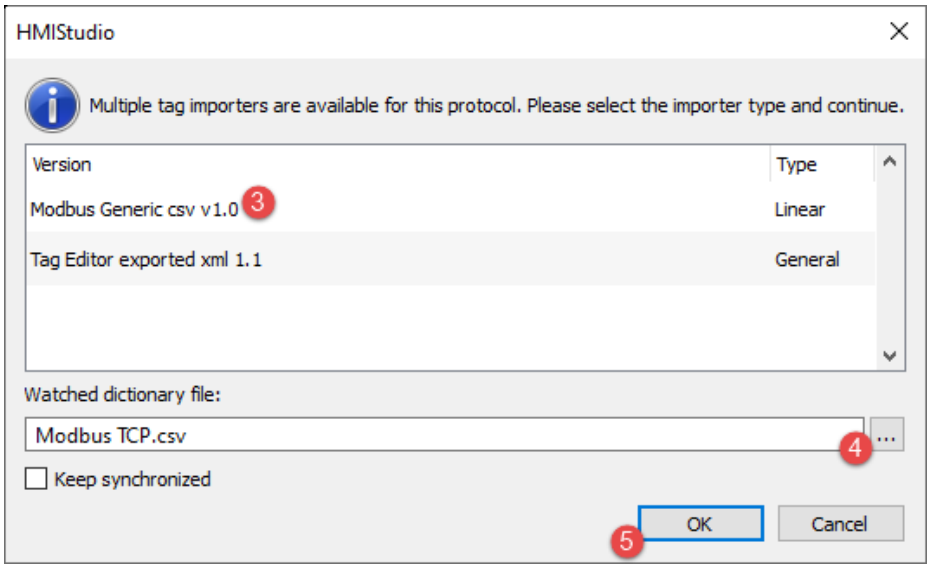
Importing tags

To import tags from an external file:

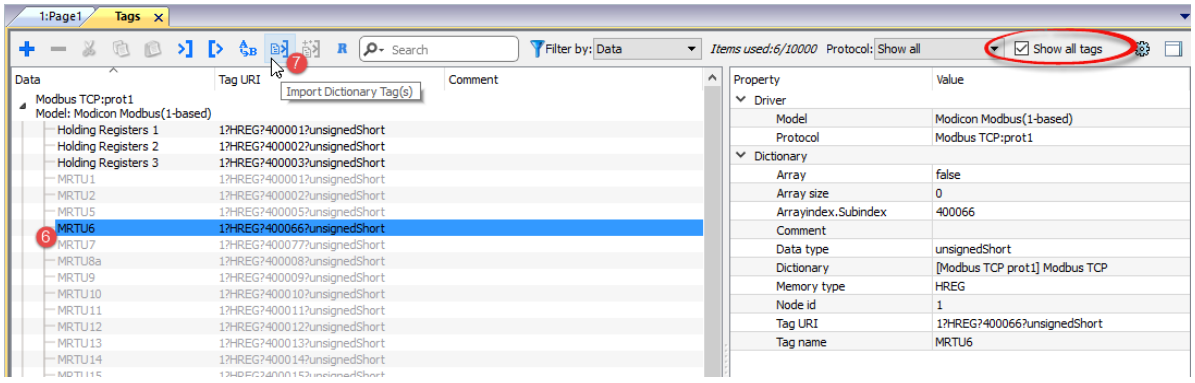
1. In **ProjectView, Tags** select the protocol from the filter list.



2. Click the **Import Tags** button: the dialog to choose the importer type appears. The list of the supported importers is depended from the selected protocol.

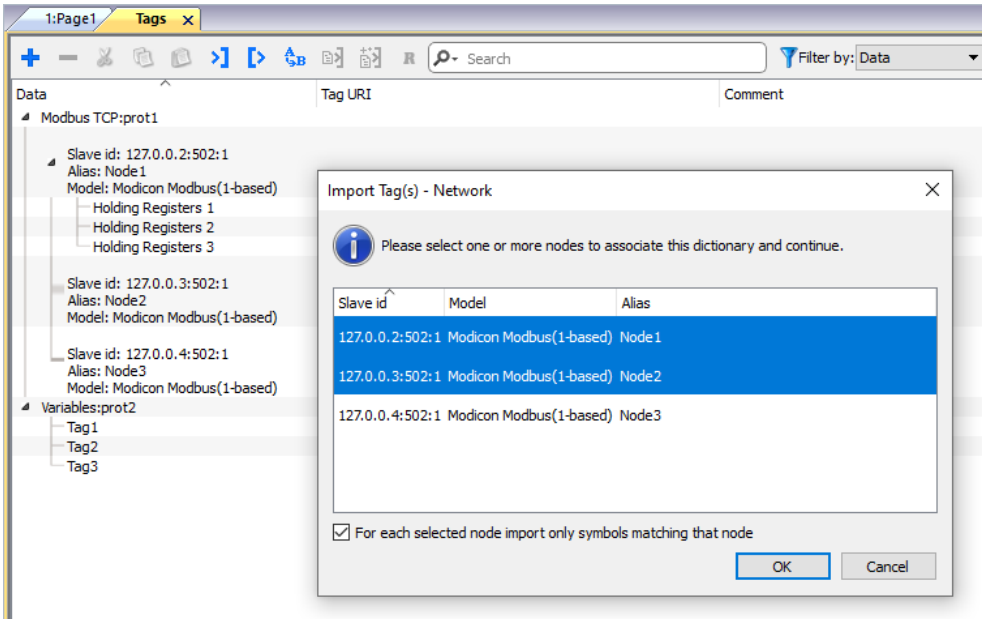


3. Select the importer type to use
4. Select the dictionary file
5. Press OK to attach the dictionary file to project file. The tags available within the Dictionary but not imported into the project are gray and are visible only when the "Show all tags" check box is selected.



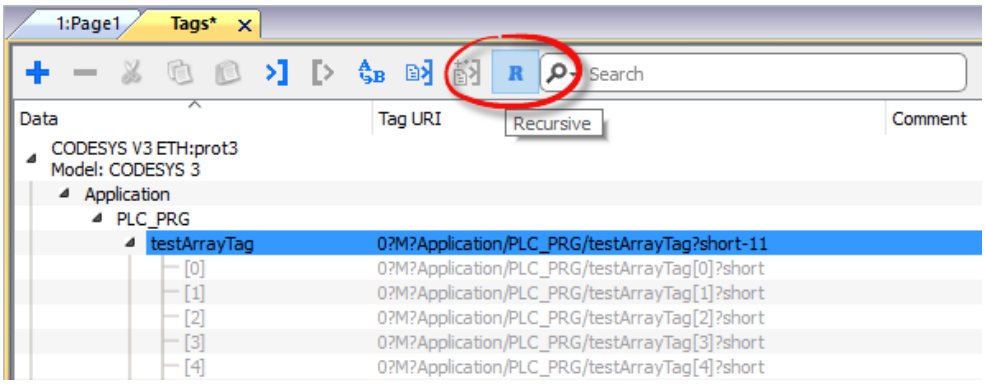
6. To import tags, select one or more tags or a node (hierarchical view only)
7. Click the **Import tag** button: tags are imported to the project and listed in black color.

When the project is configured to use a protocol network you must also select the protocol node where tags are to be imported. You can import the same tags on multiple protocols. When the tags file contains the node information, you can choose to use the information to filter the tags and import only those matching with the selected nodes.



Recursive

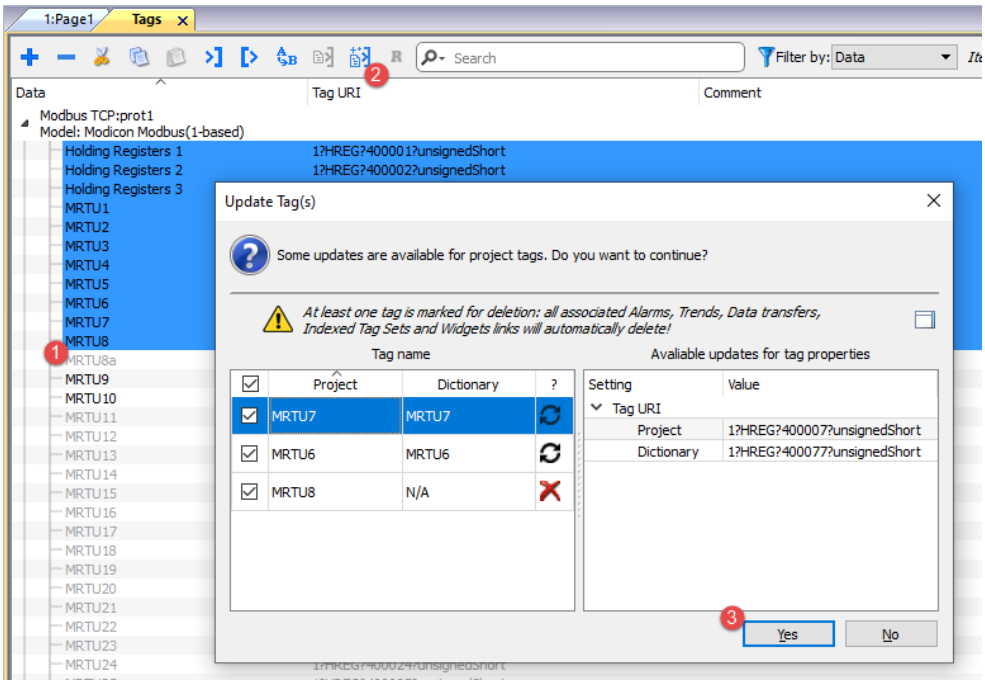
Recursive is a toggle button. When selected, when an array tag is imported even all array elements are imported into separate tags.



Updating the imported tags

To check the dictionary file and update the imported tags:

1. Select the tags that you want to check
2. If some change is found the update icon will be enabled, click the icon and the "Update Tag(s)" dialog with the list of found differences is showed
3. Unchecked the tags that you do not want yo update and click OK to confirm

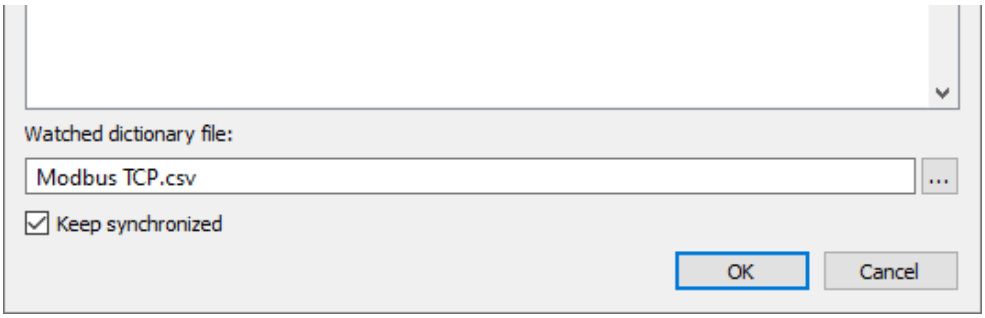


These tags need to be updated. The list of differences between project and dictionary is displayed.

These tags are no longer available in the dictionary. If updated, these tags will be removed from the project.

Keep Synchronized

Check the "Keep Synchronized" check box if you want that LRH SW checks and update the tags from file dictionary automatically without user intervention.

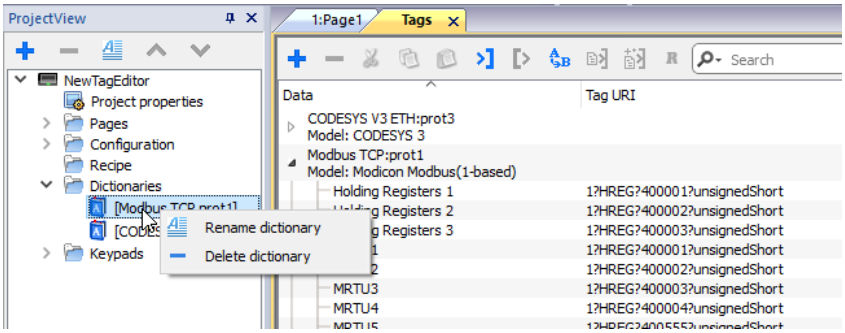


Dictionaries

Path: ***ProjectView*** *>* ***Dictionaries***

A dictionary is a list of tags imported in the Tag Editor for a specific protocol. Depending on the protocol type, tags are shown in linear view or in hierarchical view.

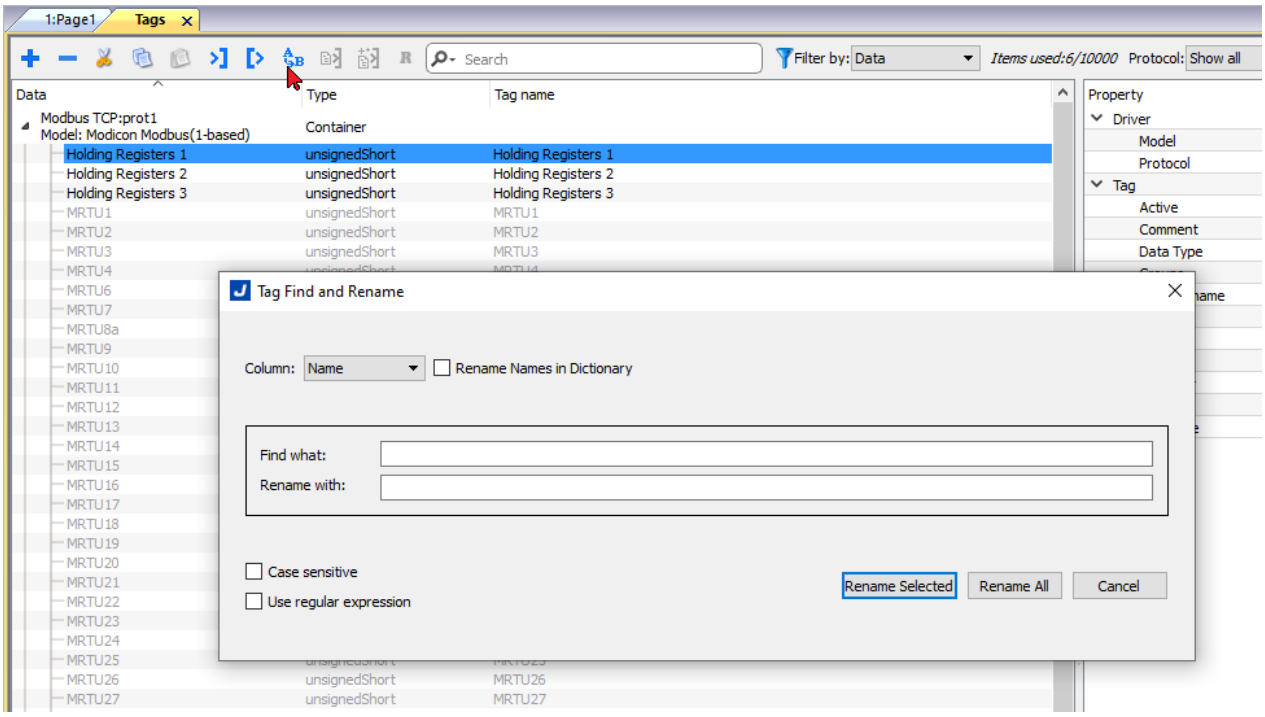
To remove a dictionary, right-click the dictionary name.



Tag find and rename

Tag find and rename feature will rename a tag reference inside the entire project.

Note this feature can be used not only to rename tags, but even to change values from each columns of the tags editor



Parameters	
Column	Select the column to modify with the find and rename operation
Rename Names in Dictionary	The tags' names rename will be extend to rename even to the internal dictionary tags' names. This parameter is useful when you have to substitute the dictionary with another dictionary that contains renamed tags This parameter is available only when the selected column is "Name"
Find what	String to search
Rename with	String to replace
Case sensitive	Takes account of upper and lower case letters
Use regular expression	Enable regular expression in search/replace pattern See https://en.wikipedia.org/wiki/Regular_expression for additional details regarding regular expressions. When regular expression is enabled, the "Find what" parameter will not offer predefined values but only free text handling.

RENAME SELECTED

Execute the rename only for the selected tags

RENAME ALL

Execute the rename for the entire tags database

References used in Java script and within custom widgets will not update.
Undo is not supported for this command

Regular expression example

Using the tags list of the above picture.

If you want add a prefix to all tags you don't need to use regular expression:

Find what:	**MRTU**
Rename with	**PLC01_MRTU**

But if you want add a postfix, you need to use a regular expression:

Find what:	**MRTU(.*)**
Rename with	**MRTU\1_PLC01**

Where

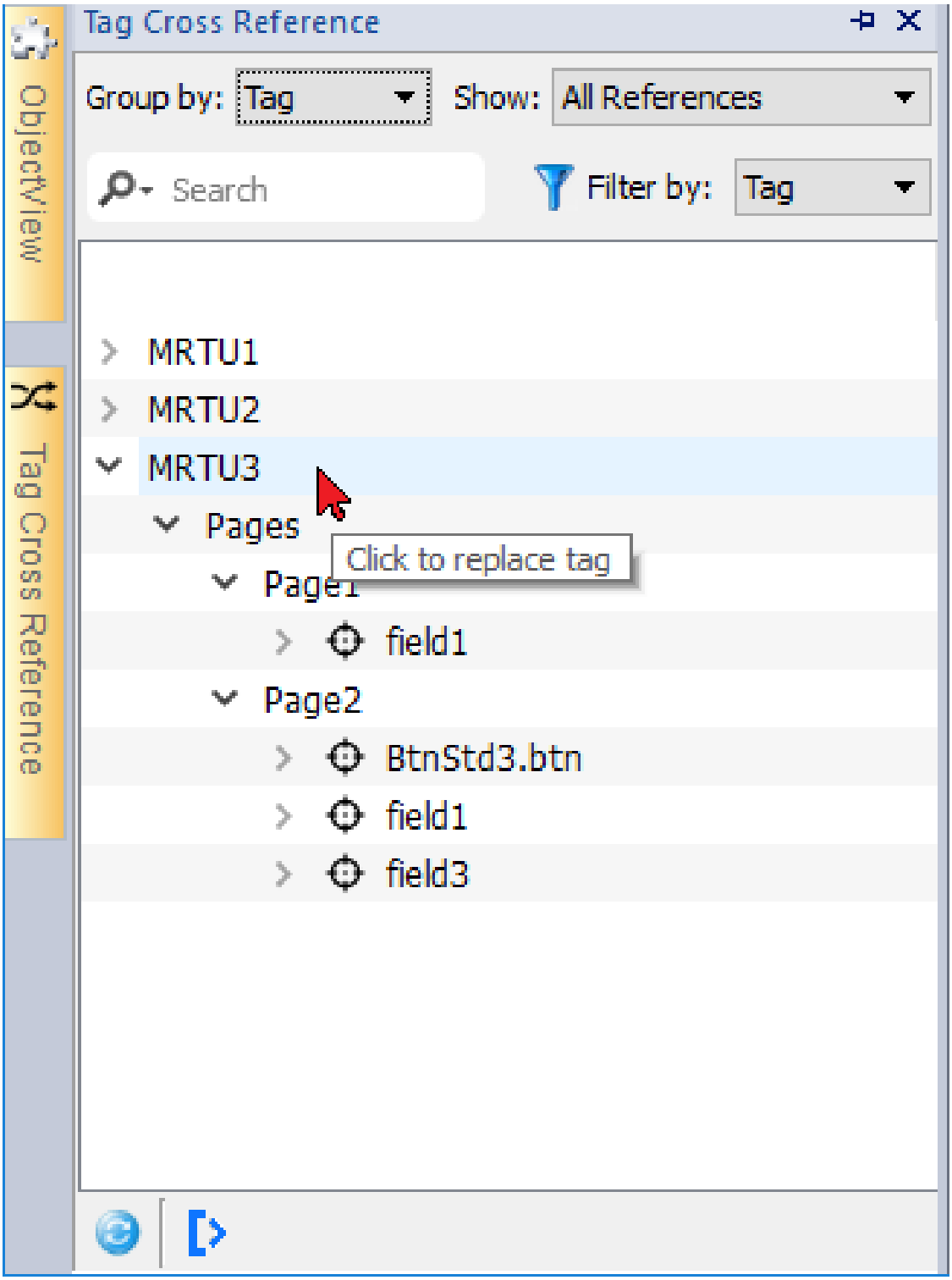
(.*)	is meaning any sequence of characters
\1	is a copy of the first sequence of characters enclosed by (...) found inside the search string

Tag find and replace

Using this feature you can search all occurrence of a tag inside the project and replace it with another tag.

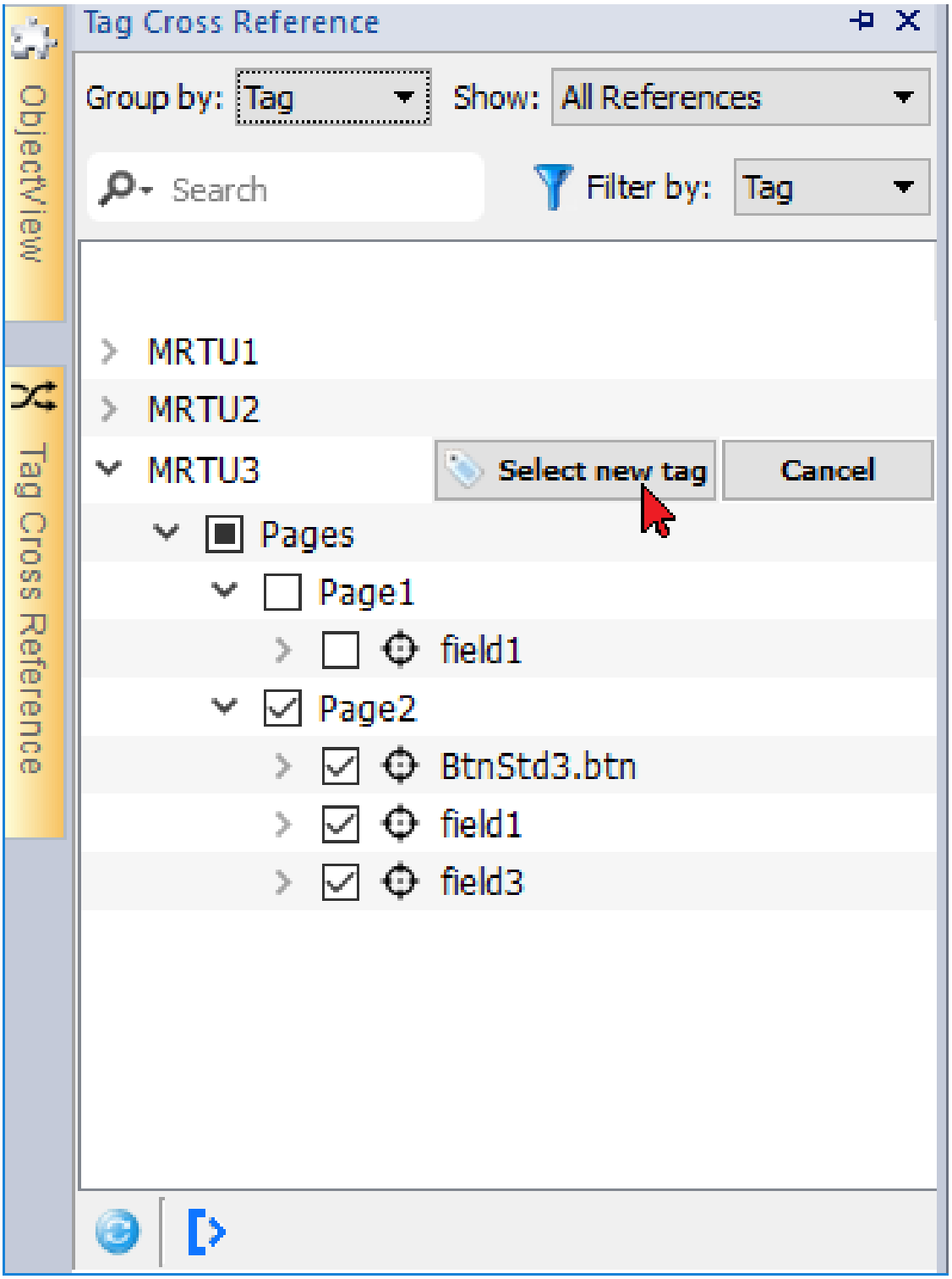
Tag replace is only applicable for Protocol tags which are shown in black color and not for System Variable, Alias and Recipe tags. (See "Opening the Tag Cross Reference pane" on page 120 for the different colors meaning)

From the Tag Cross Reference view, click the tag that you want replace



Using the check boxes select where you want apply the replace,

then click the “Select new tag” button to replace the data links of the selected objects or press “Cancel” to abort the operation.



References used in the Java script and within custom widgets may not be listed.
Undo is not supported for this command

10 Indexed addressing

Indexed addressing allows you to select a set of tags depending on the value of another tag. This is very useful, for example, to use the same graphics to visualize a set of data coming from different sources, all the user has to do is pick the source to monitor from a list.

Creating an indexed addressing set

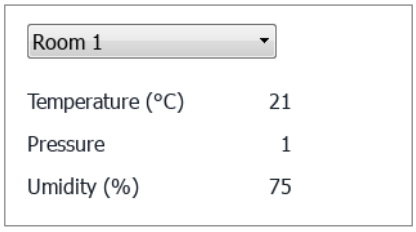
Scenario

In this scenario, environment data is collected from with four rooms, each equipped with temperature, pressure, and humidity sensors. Data is available as follows:

Room Number	Temperature	Pressure	Humidity
1	Room1-Temperature	Room1-Pressure	Room1-Humidity
2	Room2-Temperature	Room2-Pressure	Room2-Humidity
3	Room3-Temperature	Room3-Pressure	Room3-Humidity
4	Room4-Temperature	Room4-Pressure	Room4-Humidity

Using the indexed addressing feature, you can use a single table format to arrange all data in the HMI device.

Data from the three different sensors can be displayed in a single page where the room number is used as a selector (combo box) to pick the correct set of tags.



How to create an indexed tag set

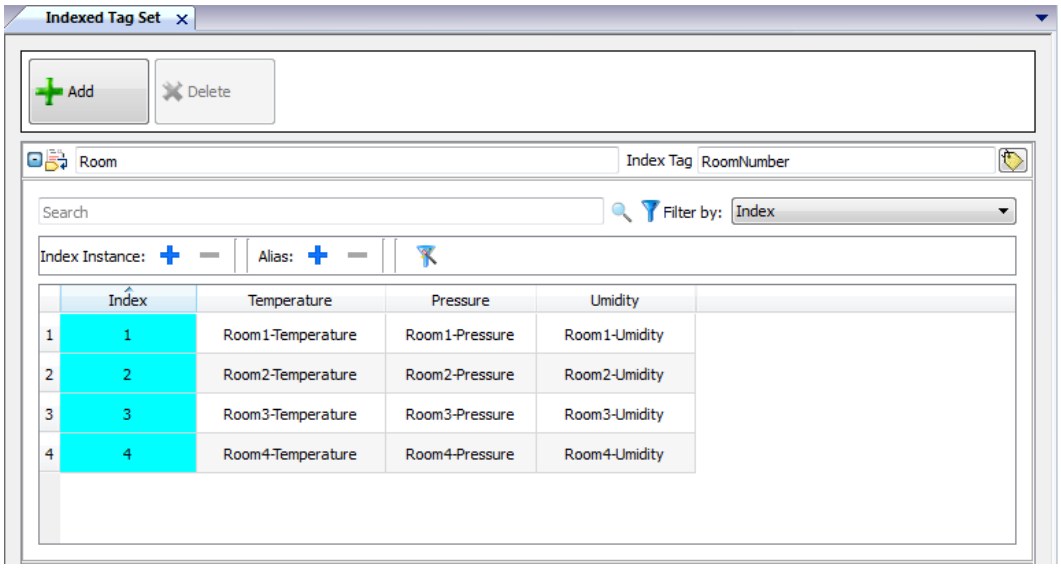
Path: ***ProjectView*** *>* ***Tags***

To do this you need to create an indexed tag set.

1. In the Tag Editor, define protocols and tag. Define a tag for each data to be indexed, in this example you must create a tag for each sensor in each room.

Name	Group	Driver	Address
Room1-Temperature		Modbus TCP:prot1	192.168.0.34:502:1 HREG 400001 unsignedShort
Room1-Pressure		Modbus TCP:prot1	192.168.0.34:502:1 HREG 400002 unsignedShort
Room1-Umidity		Modbus TCP:prot1	192.168.0.34:502:1 HREG 400003 unsignedShort
Room2-Temperature		Modbus TCP:prot1	192.168.0.34:502:1 HREG 400004 unsignedShort
Room2-Pressure		Modbus TCP:prot1	192.168.0.34:502:1 HREG 400005 unsignedShort
Room2-Umidity		Modbus TCP:prot1	192.168.0.34:502:1 HREG 400006 unsignedShort
Room3-Temperature		Modbus TCP:prot1	192.168.0.34:502:1 HREG 400007 unsignedShort
Room3-Pressure		Modbus TCP:prot1	192.168.0.34:502:1 HREG 400008 unsignedShort
Room3-Umidity		Modbus TCP:prot1	192.168.0.34:502:1 HREG 400009 unsignedShort
Room4-Temperature		Modbus TCP:prot1	192.168.0.34:502:1 HREG 400010 unsignedShort
Room4-Pressure		Modbus TCP:prot1	192.168.0.34:502:1 HREG 400011 unsignedShort
Room4-Umidity		Modbus TCP:prot1	192.168.0.34:502:1 HREG 400012 unsignedShort

2. Create a tag to be used as index tag. In this example you create a "RoomNumber" tag that could be of type UnsignedInt using Variable protocol.
3. From **ProjectView**, select **Config**> **Tags**, double-click **Indexed Tag Set**: the Indexed Tag Set editor is displayed.
4. Click + to add an Indexed Tag Set. In this example you will call it "Room".
5. Select the tag "RoomNumber" to use as a selector for the room number.
6. Create an **Index Instance** for each set of data. In this example, one for each room.
7. Create an **Alias** for each type of data and rename the table columns appropriately. In this example "Temperature", "Pressure" and "Humidity".
8. Double-click on each cell to associate the correct tag.



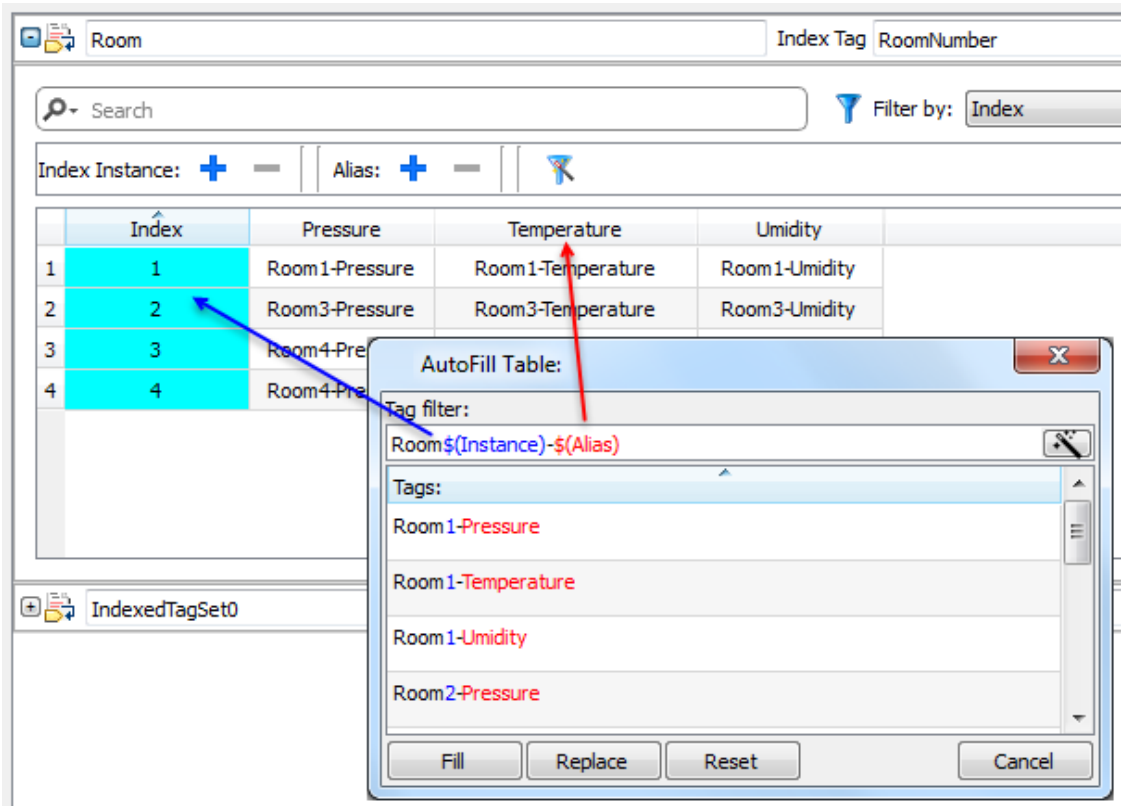
Note: The Index Tag datatype can be a number, a string or any type of simple data types.

Note: To reference an array data type use the array index = -1

Autofill function

An Indexed Tag Set table may become very complex and filling it may be an error prone procedure. Enable the Autofill feature to make sure aliases are entered correctly.

Click to enable the Autofill feature: the **Autofill Table** is displayed.



This function uses regular expression for populating the table with tags trying to match the filter where the keyword $(Instance) will be replaced with the defined Index values and the keyword $(Alias) with the defined alias labels.

See https://en.wikipedia.org/wiki/Regular_expression for additional details regarding regular expressions.

Autofill example

“Room$(Instance)-$(Alias)” will match all tag names:

Room1-Temperature,

Room1-Pressure,

Room1-Humidity,

Room2-Temperature,

…

“Room0*$(Instance)-$(Alias)” will match all tag names:

Room1-Temperature,

Room01-Pressure,

Room001-Humidity,

Room2-Temperature,

Room02-Pressure,

Room002-Humidity,

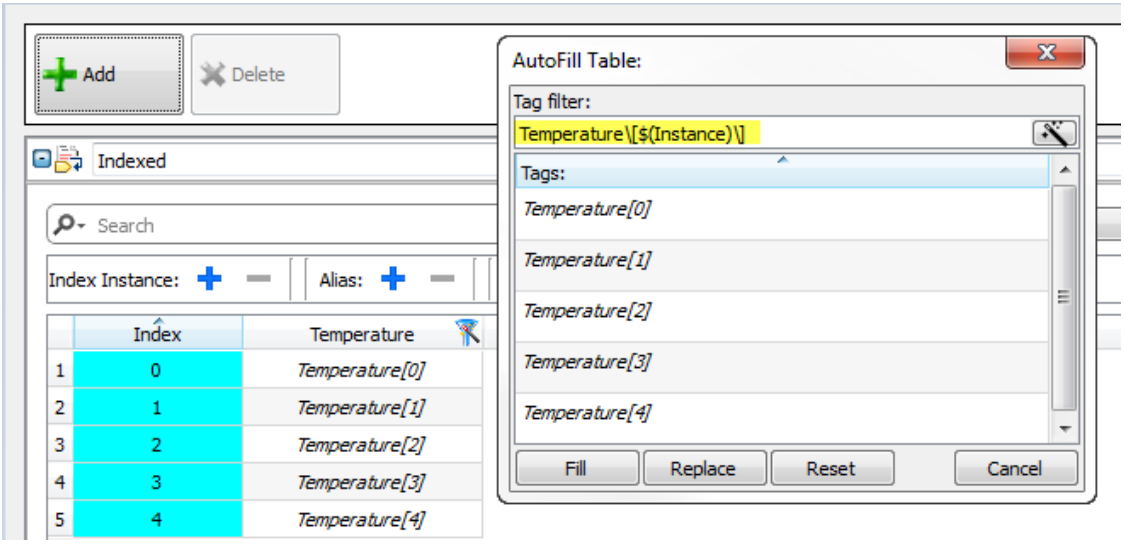
…

Autofill table elements

Element	Description
Fill	Fills in missing entries in the tag table using the set filter (if any). For example, when new instances or new aliases are added you can use this option to fill in the new entries.
Replace	Replace all table entries with those provided by the Autofill table.
Reset	Resets the tag filter to empty, no automatic fill is done.
	Suggests a valid filter expression for your project.

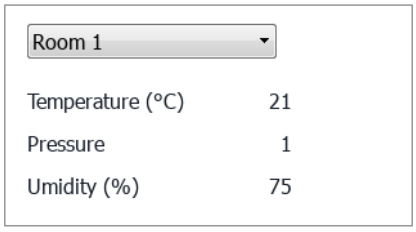
Note: Filters are saved as project preferences and can be set for the entire table or for a column. Once a filter is set for a column, the table filter is ignored. You can therefore selectively change the filter for handling a particular alias only.

Note: To reference the elements of an array use the \ character to disable the regular expression interpretation of the square brackets (array tags are differentiated by Italic).



Using indexed tag set in pages

Once an indexed tag set has been created, you can use it to create a page for the HMI device as in this example.

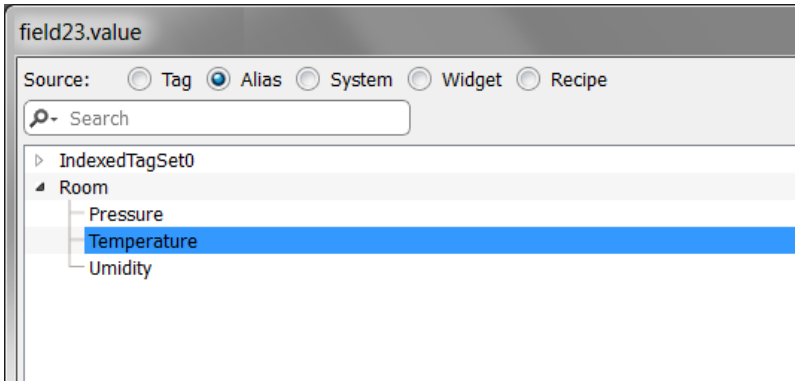


To create this page:

1. Create a page and add a combo box, three labels and three numeric fields.
2. Use the index tag created for the room number for the combo box, "RoomNumber" in this example. This will be the selector for the room number.
3. Create a list for the combo box. In this example use the following list.

Index	String List
0	Room Number
1	Room 1
2	Room 2
3	Room 3
4	Room 4

4. Attach to each numeric field value the corresponding Alias variable (**Room > Temperature, Room > Humidity, Room > Pressure**).



11 Tag cross reference

The **Tag Cross Reference** pane displays a list of tag names used in current project organized according to their location and use.

From this pane you can:

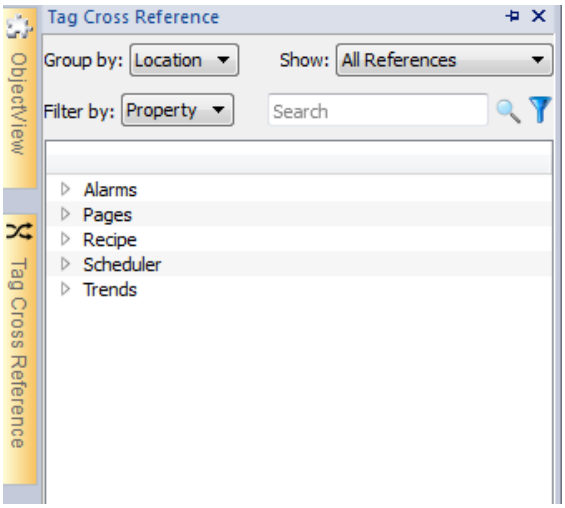
- verify where each tag is used (alarms, pages, recipes, schedulers, trends, and so on)
- identify invalid tag references (references to tags not defined in the tag editor)
- identify tags not used in the project

Note: The Tag Cross Reference pane may not be list all tags used in JavaScript code.

Opening the Tag Cross Reference pane

Path: ***View*** *>* ***Toolbars and docking windows*** *>* ***Tag Cross Reference***

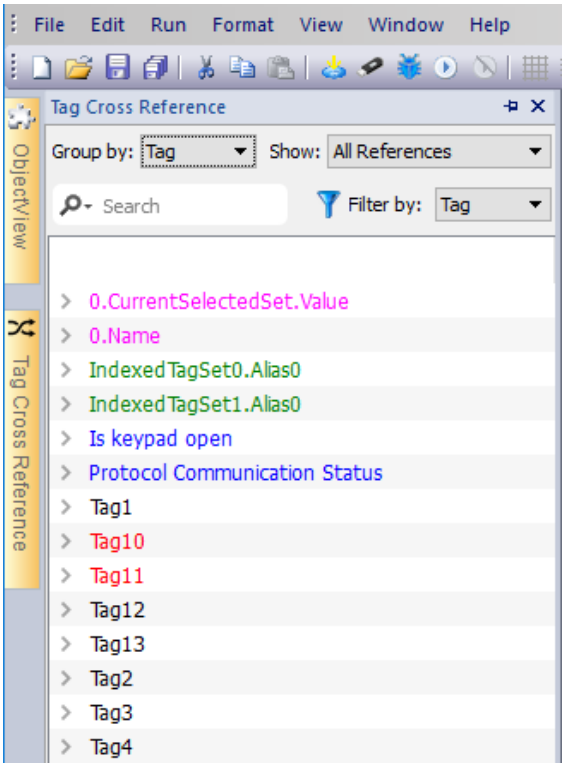
Click the **Tag Cross Reference** tab to open the Tag Cross Reference pane.



Meaning of colors

Black	Protocol Tags
Magenta	Recipe Tags
Blue	System Variable Tags
Dark Green	Alias Tags
Red	Invalid Tags

Example:



Working in the Tag Cross Reference pane

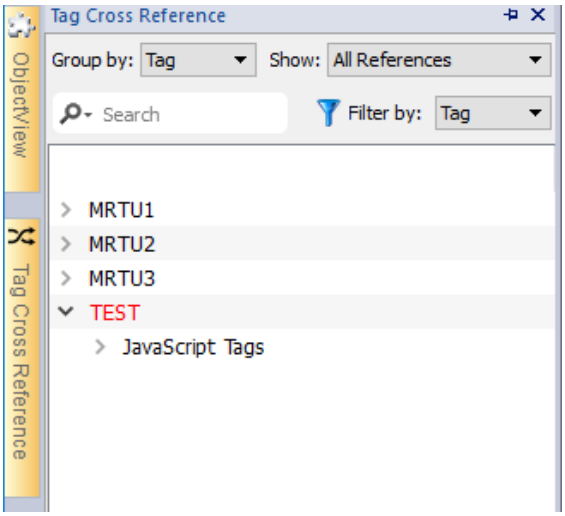
The Tag Cross Reference pane provides a set of standard functions.

Element	Function
Group by	Groups tags by **Location** (alarms, pages, trends and so on) or **Tag** name
Show	Filters tags and displays: • **All Reference**: all tags • **Invalid Tag Reference**: tags not listed in the Tag Editor. • **Unused Tags**: tags listed in the Tag Editor but not used in project.
Search field	Applies a filter to display a limited number of tags
Filter by	Filters tags by **Location**, **Tag** or **Property**.

Navigate the listed tags to find where they are used inside the project.

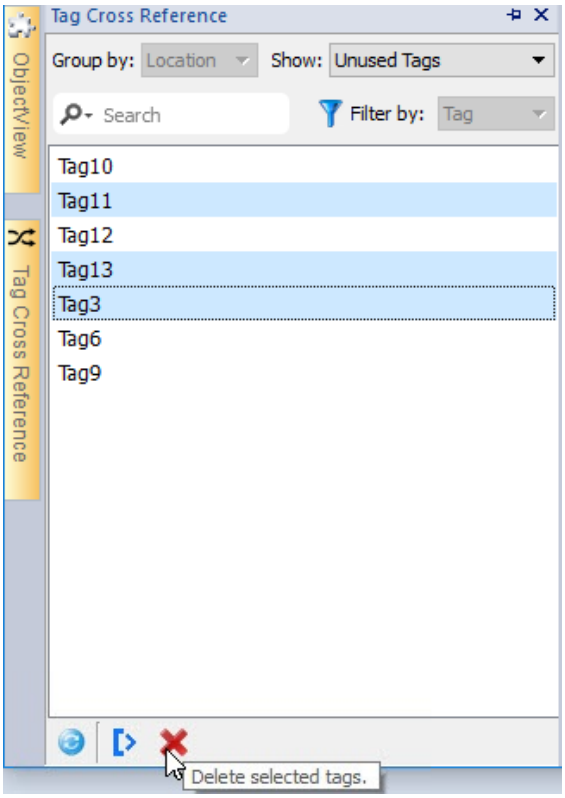
Double-click on a tag to open the editor or page where it is used.

Invalid tag references will be listed in red color:



Delete unused tags

From the unused tags view, is possible select one or more tags and delete them from the tag editor. To select a tag click a tag, to select multiple tags use SHIFT or CTRL keys.



Be aware that eventually tags referenced inside JavaScript may not be found (depends on how the code was written). Even it is not a good practical to using tags' references inside custom widgets, even tags' referenced inside Custom Widgets may not be detected from the Tag Cross Reference engine.

Updating data in the Tag Cross Reference pane

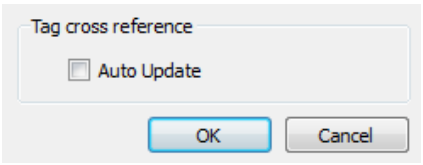
Manual update

By default, the information displayed in the Tag Cross Reference pane must be updated manually. To do this, click the refresh button  . A warning sign is displayed when a refresh is needed.

Automatic update

Path: ***View*** *>* ***Properties***

You enable the automatic update of the Tag Cross Reference pane from the LRH SW **Properties** page.



Select the **Auto Update** option.

Exporting data

Data displayed in the Tag Cross Reference pane can be exported in .csv file.

Data is organized in the exported file according to how it was grouped in the pane.

Grouped by	File format
Location	RESOURCE, RESOURCE DESC, WIDGET-ID, ATTRIBUTE, TAG
Tag	TAG, RESOURCE, RESOURCE DESC, WIDGET-ID, ATTRIBUTE

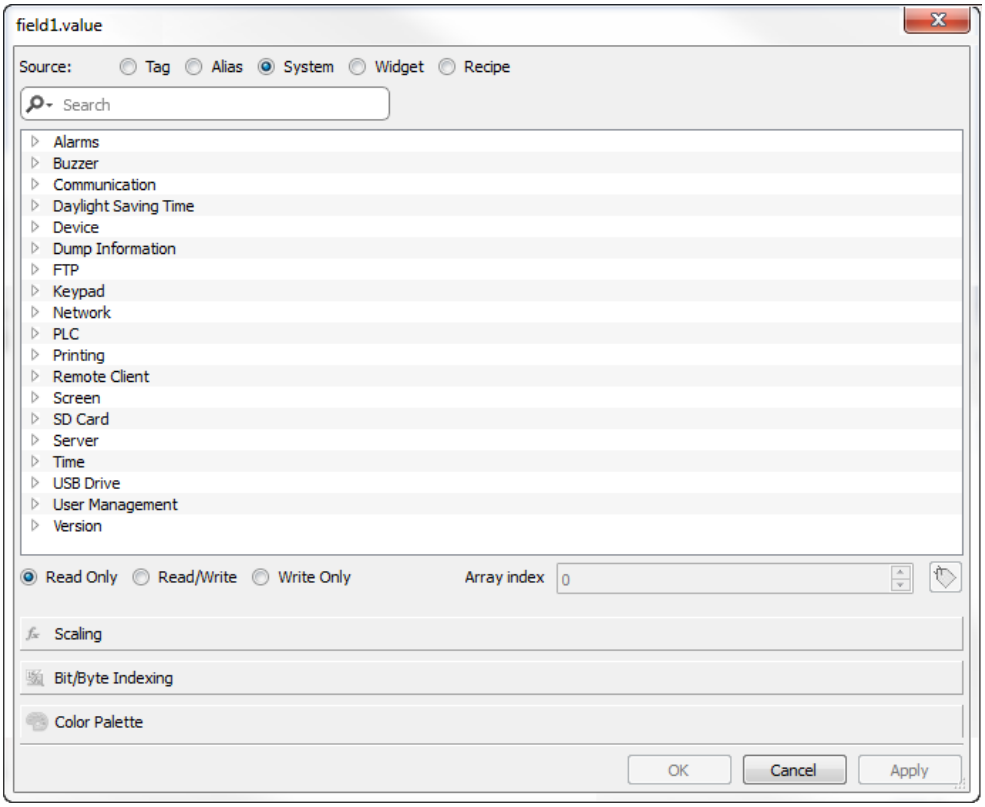
Note: The separators used in export operation depends on regional settings of your computer.

12 System Variables (Attach To)

Path: ***Source> Attach to***

System variables are special tags containing information about the HMI runtime.

Note: System Variables are available also as a standard protocol in the Protocol Editor. Use System Variables as a protocol when you have to transfer data between system variables and tags from devices, or to select custom refresh rate for a system variable.



Alarms variables

Number of alarms of the requested type.

Variable	Description	Data type
Alarm not acknowledged	True when alarms unacknowledged is pending (Not Triggered Not Acknowledged<>0) OR (Triggered Not Acknowledged<>0)	boolean read only
Alarm triggered	True when at least one alarm is triggered (Triggered Acknowledged<>0) OR (Triggered <>0) OR (Triggered Not Acknowledged<>0)	boolean read only
Number of missed alarm events	Alarms exceeding the event queue. Queue length is defined in the *engineconfig.xml* file.	int read only
Number of not triggered acknowledged	Alarm condition no longer active; alarms already acknowledged	int read only
Number of not triggered not acknowledged	Alarma condition no longer active; awaiting acknowledgment	int read only
Number of triggered acknowledged	Alarm condition active; alarms already acknowledged	int read only
Number of triggered alarms	Alarm active: aknowledgement not required	int read only
Number of triggered not acknowledged	Alarm condition active; awaiting acknowledgment	int read only

Note: For compatibility reasons, the older names are still valid but they usage is deprecated.

Buzzer variables

Adjust buzzer behavior.

Variable	Description	Data type
Buzzer Setup	**0** = disabled **1** = enabled (buzzer sounds as audible on any touchscreen event) **2** = buzzer status controlled by **Buzzer Control** system variable or by **Buzzer on Touch** property inside the "Project properties" on page 73  **Buzzer on touchscreen (Setup=1) is not available on Linux platforms. See "*Buzzer on Touch*" property in alternative.**	int
Buzzer Control	**0** = buzzer off **1** = buzzer on **2** = buzzer blink	int
Buzzer Off Time	Duration in milliseconds of off time when blink has been selected. Default = 1000. Range: 100–5000.	int
Buzzer On Time	Duration in milliseconds of on time when blink has been selected. Default = 1000. Range: 100–5000.	int

Communication variables

Communication status between HMI device and controllers.

Variable	Description	Data type
Protocol Communication Status	Summarize the status of the communication protocols. **0** = No protocol running, protocol drivers might not have been properly downloaded to the HMI device. **1** = Protocols loaded and started, no communication error. **2** = At least one communication protocol is reporting an error.	int Read only
Protocol Error Message	Communication error with error source. For example: "[xxxx]" where "xxxx" is the protocol abbreviation, the error source. Multiple acronyms appear in case of multiple error sources. Blank when no errors are reported.	ASCII string Read only
Protocol Error Count	Number of communication errors occurred since last reset. Reset value with Reset Protocol Error Count action, see "System actions" on page 192.	int Read only

Daylight Saving Time variables

Information on the system clock. The variables contain information on the "local" time. Standard Time (solar time) and Day Light Saving time (DST) are available.

Note: All variables are read only; you cannot use them to update the system clock.

Variable	Description
Standard Offset	Offset in minutes when standard time is set, with respect to GMT (for example: -8x60 = -480 minutes).
Standard Week	Week in which the standard time starts (for example: First = 1).
Standard Month	Month in which the standard time starts. Range: 0–11. (for example: November = 10).
Standard Day	Day of week in which the standard time starts (for example: Sunday = 0).
Standard Hour	Hour in which the standard time starts (for example: 02 = 2).
Standard Minute	Minute in which the standard time starts (for example: 00 = 0).
DST Offset	Offset in minutes when DLS time is set, with respect to GMT
DST Week	Week in which the DLS time starts
DST Month	Month in which the DLS time starts. Range: 0–11.
DST Day	Day of week in which the DLS time starts
DST Hour	Hour in which the DLS time starts
DST Minute	Minute in which the DLS time starts

Device variables

Device settings and operating status information.

Variable	Description	Data type
Available System Memory	Free available RAM memory in bytes.	uint64 read only
Backlight Time	Activation time in hours of the display backlight since production of the device.	unsignedInt read only
Display Brightness	Returns and adjusts brightness level. Even when set to 0, the backlight is still on and the **Backlight Time** counter increases. Range: 0–255	int
External Timeout	Non-operational time after which the display backlight is automatically turned off. The backlight is automatically turned on when the user touches the screen.	int

Variable	Description	Data type
	0 = Switch backlight on (switch display on) **1..n** = Timeout, in seconds, for switch off backlight (screen saver timer) The timeout value is rounded to multiples of one minute (60, 120, 180, etc,). Find the platform of your device at "HMI devices capabilities" on page 541	
Flash Free Space	Free space left in internal Flash memory.	uint64 read only
Manufacturer Code	Internal code that identify the HMI type	unsignedInt read only
System Font List	List of system fonts	string read only
System Mode	Runtime operation status. **1** = booting **2** = configuration mode **3** = operating mode **4** = restart **5** = shutdown	int
System UpTime	Time the system has been powered since production of the unit (hours).	unsignedInt read only

Dump information variables

Status of the copy process to external drives (USB or SD Card) for trend and event buffers.

Variable	Description	Data type
Dump Error Message	Return error message if any error occurs during the dump operation	string read only
Dump Archive Status	**0** = initial default state **1** = operation triggered **2** = operation complete successfully **3** = operation completed with errors	int read only
Dump Recipe Status	**0** = initial default state	int

Variable	Description	Data type
	1 = operation triggered **2** = operation complete successfully **3** = operation completed with errors	read only
Dump Trend Status	**0** = initial default state **1** = operation triggered **2** = operation complete successfully **3** = operation completed with errors	int read only
Reset Recipe Status	**0** = initial default state **1** = operation triggered **2** = operation complete successfully **3** = operation completed with errors	int read only
Restore Recipe Status	**0** = initial default state **1** = operation triggered **2** = operation complete successfully **3** = operation completed with errors	int read only

FTP client variables

The FTP client variables are updated when the FTP actions are used.

Variable	Description	Data type
FTP Current Command	Last FTP command	string read only
FTP Error Message	Last FTP error message	string read only
FTP Progress	Download/upload progress (0/100%)	short read only
FTP Status	Status of last FTP command: • 0 = idle • 1 = active • 2 = done • 3 = error	short read only

Keypad variables

Keypad status.

Variable	Description	Data type
Is keypad open	**0** = no keypad open **1** = keypad open	int read only

Network variables

Device network parameters.

Variable	Description	Data type
Adapters Parameters	This is a JSON string that can be use to read or update the network adapters parameters	string
Gateway	Gateway address of the main Ethernet interface of device	string read only
IP Address	IP address of the main Ethernet interface of device	string read only
Mac ID	MAC ID of the main Ethernet interface of device	string read only
Status	Contains the result of the last operation required by writing inside the Adapter Parameters. It is updated after each write operation. • Empty string is meaning no errors • Last error descriptions	string read only
Subnet Mask	Subnet Mask of the main Ethernet interface of device	string read only

Printing variables

Information on printing functions.

Variable	Description	Data type
Completion percentage	Percentage of completion of current print job. Range: 0–100	read only
Current disk usage	Folder size in bytes where PDF reports are stored. If *Flash* has been selected as *Spool media type*, this value corresponds to *reportspool*.	read only
Current job	Name of the report the job is processing. Current job is the following: • [report name] for a **Graphic Report** • [first line of text] for a **Text Report**	read only
Current RAM usage	Size in bytes of the RAM used to process the current job	read only
Disk quota	Maximum size in bytes of the folder where PDF reports are stored	read only
Graphic job queue size	Number of available graphic jobs in the printing queue	read only
Last error message	Description of the last returned error	string read only
RAM quota	Maximum size in bytes of the RAM used to generate reports	read only
Status	Printing system status. Values: • **idle** • **error** • **paused** • **printing**	string read only
Text job queue size	Number of available text jobs in the printing queue	read only

Remote Client variables

On remote clients, the below system variable can be used to know if the server (HMI device) is reachable.

Variable	Description	Data type
Connection status	**0** = client can not reach the server client. The connection with server is lost. **1** = client can reach the server. The connection with server is active. This is only a client side variable. On LRH SW HMI Runtime it will be always 0.	int (32 bit) read only

The following system variables are associated to the transferring files to a remote HMI device.

Variable	Description	Data type
Download from HMI error message	Error description	ASCII string read only
Download from HMI percentage	Download progress (0→100)	read only
Download from HMI status	**0** = idle, action is not in use or completed **1**= file download in progress **2** = error	int (32 bit) read only
Upload to HMI error message	Error description	ASCII string read only
Upload to HMI percentage	Upload progress (0→100)	read only
Upload to HMI status	**0** = idle, action is not in use or completed **1**= file upload in progress **2** = error	int (32 bit) read only

Version variables

Operating System and runtime version.

Variable	Description	Data type
Main OS Version	Version of Main OS.	string
Runtime Version	Version of runtime.	string

Screen variables

Screen status.

Variable	Description
Time remaining to unlock	Time remaining to unlock screen (see **LockScreen** action, "Page actions" on page 180)
X Screen resolution	Display horizontal screen size in pixel
Y Screen resolution	Display vertical screen size in pixel

SD card variables

Information on the external SD card.

Variable	Description	Data type
SD Card FreeSpace	Available space on card in bytes	uint64 read only
SD Card Name	Name of SD card	string read only
SD Card Size	Size in bytes of the card plugged in the slot	uint64 read only
SD Card Status	0 = SD card unplugged 1 = SD card plugged	int read only

Server variables

Server status.

Important: All variables refer to server, not to LRH SW Client.

Variable	Description	Data type
Current page	Name of current page	string
Current project	Name of current project	string
Operating mode time	Seconds elapsed since device started operating mode as in **System Date** format (milliseconds).	uint64
Project load time	Date when the project was loaded on the LRH SW HMI Runtime as in **System Date** format (milliseconds).	uint64

Time variables

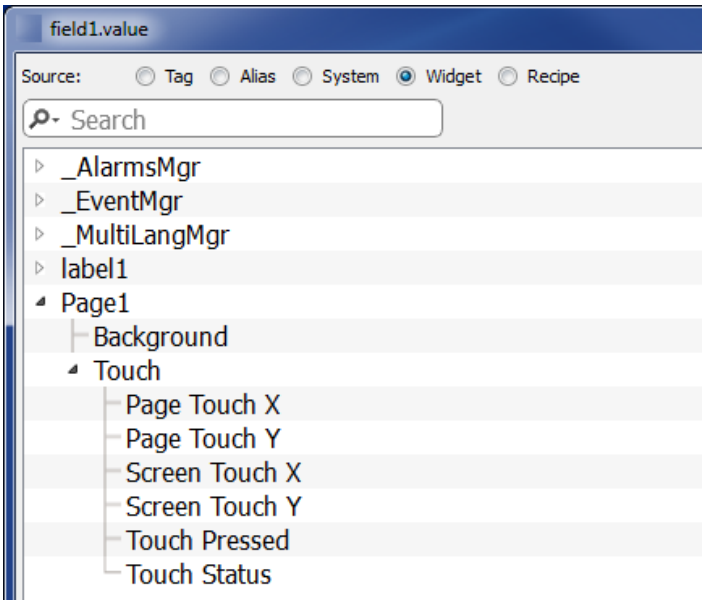
System time expressed in UTC format.

Variable	Description	Data type
Day Of Month	Range: 1–31	int
Day of Week	Range: 0 = Sunday, .. , 6 = Saturday	int
Hour	Range: 0–23	int
Minute	Range: 0–59	int
Month	Range: 1–12	int

Variable	Description	Data type
Second	Range: 0–59	int
System Time	The same as UTC time. It can also be set as date/time for this variable.	unsignedInt
Year	Current Year	int

Touch screen variables

Cursor status and position on the touchscreen. These are properties of the active page and can be selected in the **Widget** section.



Note: Page size can be different than HMI device display size.

Variable	Description	Java Script
Page Touch X **Page Touch Y**	Cursor position related to page	page.primaryTouch.x page.primaryTouch.y
Screen Touch X **Screen Touch Y**	Cursor position related touchscreen	page.primaryTouch.screenX page.primaryTouch.screenY

Variable	Description	Java Script
Touch Press	**0** = screen not pressed **1** = screen pressed	page.primaryTouch.pressed
Touch Status	Generic touch screen changes. This variable contains the concatenation of **Screen Touch X**, **Screen Touch Y** and **Touch Press** values (for example, “924,129,0”). The main usage of this variable is to trigger an event, using the OnDataUpdate feature, when something (x, y or click) is changed.	page.primaryTouchStatus

USB drive variables

Information on the external USB drive connected to the device.

Variable	Description	Data type
USB Drive free space	Available space in bytes	uint64 read only
USB Drive Name	Name of USB device	string read only
USB Drive Size	Size in bytes of the device plugged in the USB port	uint64 read only
USB Drive Status	0 = USB Drive unplugged 1 = USB Drive plugged	int read only

User management variables

Information on users and groups.

Variable	Description	Data type
This Client User-Name	Name of the user logged to the client where the system variable is displayed.	string read only
This Client Group-Name	Group of currently logged user	string read only
This Client ID	Only for LRH SW Clients. Local and remote clients connected to the same server (for example, runtime) get a unique ID.	short

Variable	Description	Data type
		read only
No Of Remote-Clients Alive	Number of LRH SW Clients connected to the server	short read only

JavaScript

From JavaScript, the variables can be accessed as properties of the _SysPropMgr object.

Example:

```
var sysVar = project.getWidget( "_SysPropMgr" );

var UserName   = sysVar.getProperty("This Client User-Name");
var UserGroup  = sysVar.getProperty("This Client Group-Name");
var clientId   = sysVar.getProperty("This Client ID");
var numClients = sysVar.getProperty("No Of Remote-Clients Alive");
```

13 System Variables (Protocol)

System Variables communication driver allows to create Tags that point to system information.

System Variables communication driver is not counted as physical protocol.
Refer to **Table of functions and limits** from main manual in "Number of physical protocols" line.

Protocol Editor Settings

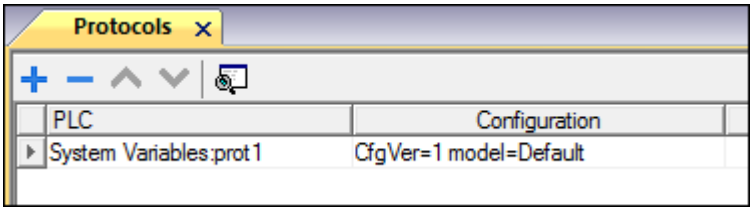
Adding a protocol

To configure the protocol:

1. In **Config** node double-click **Protocols**.
2. To add a driver, click **+**: a new line is added.
3. Select the protocol from the **PLC** list.

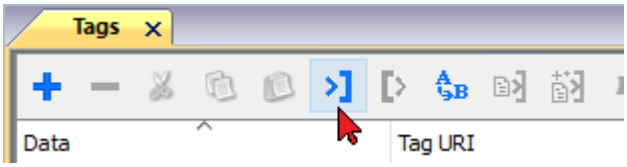
The protocol configuration dialog is displayed.

From PLC Model list select the specific System Variables type.



Tag Import

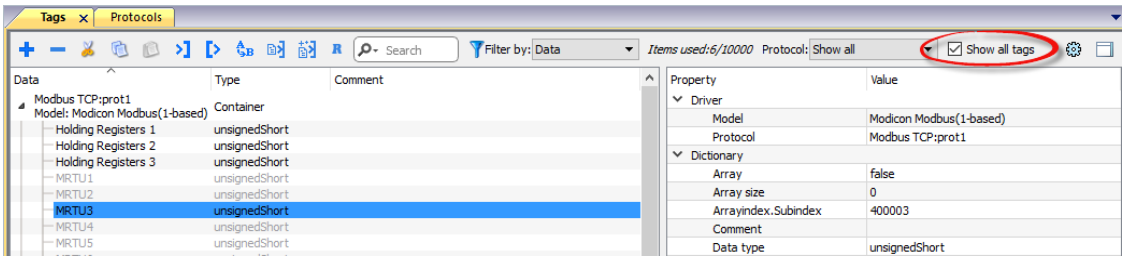
Select the driver in Tag Editor and click on the **Import Tags** button to start the importer.

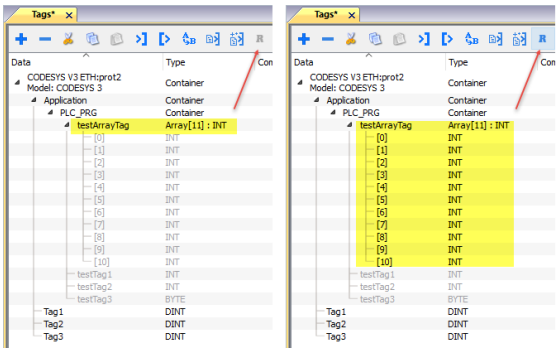
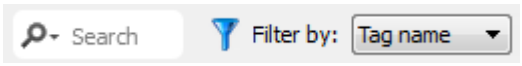


The system will require a generic XML file exported from Tag Editor by appropriate button.

Once the importer has been selected, locate the symbol file and click **Open**.

The tags available within the Dictionary but not imported into the project are gray and are visible only when the "Show all tags" check box is selected.



Toolbar item	Description
	**Import Tag(s)**. Select tags to be imported and click on this icon to add tags from tag dictionary to the project
	**Update Tag(s)**. Click on this icon to update the tags in the project, due a new dictionary import.
	Check this box to import all sub-elements of a tag. Example of both checked and unchecked result: 
	Searches tags in the dictionary basing on filter combo-box item selected.

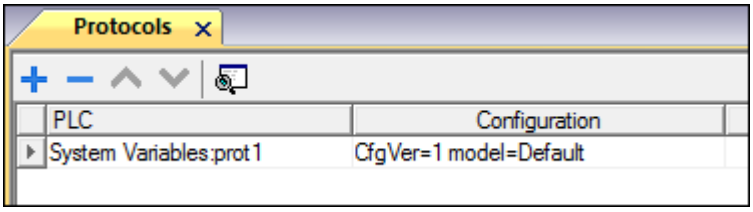
Default variables

System Variables - Default protocol allows to create Tags that point to HMI system variables regarding:

- Alarms
- Buzzer
- Communication
- Database
- Daylight Saving Time
- Device
- Dump information
- Network
- Screen
- SD Card
- Server
- Time
- USB Drive
- Version
- Virtual Com Switch

Protocol Editor Settings

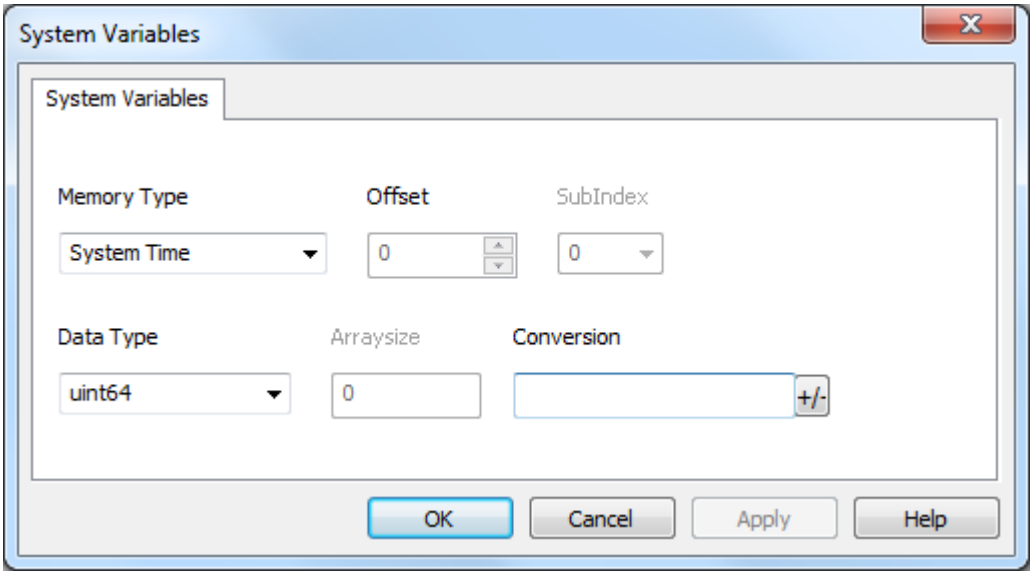
From PLC Model list of Protocol Editor dialog, select Default.



Tag Editor Settings

Path: ***ProjectView*** *>* ***Config*** *> double-click* ***Tags***

1. To add a tag, click **+**: a new line is added.
2. Select **System Variables** from the **Driver** list: tag definition dialog is displayed.



<table>
<tr><th>Element</th><th>Description</th></tr>
<tr><td>Memory Type</td><td>Represents the system variable to which the Tag refers to.
The below section shows the full list of possible system variables, grouped by category.

<table>
<tr><th colspan="3">Alarms Variables</th></tr>
<tr><th>Variable Name</th><th>Description</th><th>Data Type</th></tr>
<tr><td>Alarm not acknowledged</td><td>True when alarms unacknowledged is pending
(Not Triggered Not Acknowledged<>0) OR (Triggered Not Acknowledged<>0)</td><td>boolean
read only</td></tr>
<tr><td>Alarm triggered</td><td>True when at least one alarm is triggered
(Triggered Acknowledged<>0) OR (Triggered <>0) OR (Triggered Not Acknowledged<>0)</td><td>boolean
read only</td></tr>
<tr><td>Number of missed alarm events</td><td>Alarms exceeding the event queue. Queue length is defined in the engineconfig.xml file.</td><td>int
read only</td></tr>
<tr><td>Number of not triggered acknowledged</td><td>Alarm condition no longer active; alarms already acknowledged</td><td>int
read only</td></tr>
<tr><td>Number of not triggered not acknowledged</td><td>Alarm condition no longer active; awaiting acknowledgment</td><td>int
read only</td></tr>
<tr><td>Number of triggered acknowledged</td><td>Alarm condition active; alarms already acknowledged</td><td>int
read only</td></tr>
<tr><td>Number of triggered alarms</td><td>Alarm active: acknowledgment not required</td><td>int
read only</td></tr>
<tr><td>Number of triggered not acknowledged</td><td>Alarm condition active; awaiting acknowledgment</td><td>int
read only</td></tr>
</table>
</td></tr>
</table>

Element	Description

Buzzer Variables

Variable Name	Description	Data Type
Buzzer Setup	**0** = disabled **1** = enabled (buzzer sounds as audible on any touchscreen event) **2** = buzzer status controlled by Buzzer Control system variable or by Buzzer on Touch property inside the "Project properties" of main manual  **Buzzer on touchscreen (Setup=1) is not available on Linux platforms. See "Buzzer on Touch" property in alternative.**	int
Buzzer Control	**0** = buzzer off **1** = buzzer on **2** = buzzer blink	int
Buzzer Off Time	Duration in milliseconds of off time when blink has been selected. Default = 1000. Range: 100–5000	int
Buzzer On Time	Duration in milliseconds of on time when blink has been selected. Default = 1000. Range: 100–5000	int

Element	Description

Communication Variables

Variable Name	Description	Data Type
Protocol Communication Status	Summarize the status of the communication protocols. **0** = No protocol running, protocol drivers might not have been properly downloaded to the HMI device **1** = Protocols loaded and started, no communication error **2** = At least one communication protocol is reporting an error	int read only
Protocol Error Message	Communication error with error source. For example: "[xxxx]" where "xxxx" is the protocol abbreviation, the error source. Multiple acronyms appear in case of multiple error sources. Blank when no errors are reported.	string read only
Protocol Error Count	Number of communication errors occurred since last reset. Reset value with Reset Protocol Error Count action, see "System actions" of main manual	int read only

Database Variables

Variable Name	Description	Data Type
Database link error message	Last detected error description	string read only
Database link status	**0** = Undefined (not yet initialized) **1** = OnLine (ready) **2** = OffLine (not available) **3** = Transfer in progress **4** = Error	int read only
Database link error count	Errors counter. Increased after each error	int read only

Element	Description

Each database variable is an array where index select the database link connection (Range 1-10)
Variables are updated only when any database connector action is executed

Daylight Saving Time Variables

Variable Name	Description	Data Type
Standard Offset	Offset in minutes when standard time is set, with respect to GMT (for example: -8x60 = -480 minutes)	int read only
Standard Week	Week in which the standard time starts (for example: First = 1)	int read only
Standard Month	Month in which the standard time starts. Range: 0–11. (for example: November = 10)	int read only
Standard Day	Day of week in which the standard time starts (for example: Sunday = 0)	int read only
Standard Hour	Hour in which the standard time starts (for example: 02 = 2)	int read only
Standard Minute	Minute in which the standard time starts (for example: 00 = 0)	int read only
DST Offset	Offset in minutes when DLS time is set, with respect to GMT	int read only
DST Week	Week in which the DLS time starts	int read only
DST Month	Month in which the DLS time starts. Range: 0–11	int read only
DST Day	Day of week in which the DLS time starts	int read only
DST Hour	Hour in which the DLS time starts	int read only
DST Minute	Minute in which the DLS time starts	int read only

Element	Description

All variables are read only: they cannot be used to update the system clock.

Device Variables

Variable Name	Description	Data Type
Available System Memory	Free available RAM memory in bytes	uint64 read only
Backlight Time	Activation time in hours of the display backlight since production of the device	unsignedInt read only
Battery LED	Enables/disables the low battery LED indicator (when available) **0** = disabled **1** = enabled Not available on Linux platforms (find the platform of your device at "HMI devices capabilities" on page 541)	int
External Timeout	Non-operational time after which the display backlight is automatically turned off. The backlight is automatically turned on when the user touches the screen **-1** = Switch off backlight and disable touch (switch display off). **Backlight Time** counter is stopped. requires BSP v1.0.324 or higher. **-2** = Switch off backlight but not disable touch. If touch is pressed, event is not passed to applications but screen saver exit and backlight return on. Requires BSP v1.0.324 or higher. **0** = Switch backlight on (switch display on) **1..n** = Timeout, in seconds, for switch off	int

Element	Description

Device Variables

Variable Name	Description	Data Type
	backlight (screen saver timer)  The timeout value is rounded to multiples of one minute (60, 120, 180, etc,). Find the platform of your device at "HMI devices capabilities" on page 541	
Flash Free Space	Free space left in internal Flash memory	uint64 read only
Manufacturer Code	Code number that identifies the HMI	short read only
System RAM Usage	Current RAM memory used from HMI, expressed in byte	uint64 read only
System Font List	List of system fonts	string read only
System Mode	Runtime operation status **1** = booting **2** = configuration mode **3** = operating mode **4** = restart **5** = shutdown	int
System UpTime	Time the system has been powered since production of the unit (hours)	unsignedInt read only

Element	Description

Dump information Variables

Variable Name	Description	Data Type
Dump Error Message	Return error message if any error occurs during the dump operation	string read only
Dump Archive Status	**0** = initial default state **1** = operation triggered **2** = operation complete successfully **3** = operation completed with errors	int read only
Dump Recipe Status	**0** = initial default state **1** = operation triggered **2** = operation complete successfully **3** = operation completed with errors	int read only
Dump Trend Status	**0** = initial default state **1** = operation triggered **2** = operation complete successfully **3** = operation completed with errors	int read only
Reset Recipe Status	**0** = initial default state **1** = operation triggered **2** = operation complete successfully **3** = operation completed with errors	int read only
Restore Recipe Status	**0** = initial default state **1** = operation triggered **2** = operation complete successfully **3** = operation completed with errors	int read only

Network Variables0

Variable Name	Description	Data Type
Gateway	Gateway address of the main Ethernet interface of HMI	string read only
IP Address	IP address of the main Ethernet interface of HMI	string read only
Mac ID	MAC ID of the main Ethernet interface of HMI	string read only
Network Adapter	JSON string that can be use to read or update	string

Element	Description

Network Variables0

Variable Name	Description	Data Type
Parameters	the network adapters parameters	
Network Status	Contains the result of the last operation required by writing inside the Adapter Parameters. It is updated after each write operation. • Empty string is meaning no errors • Last error descriptions	string read only
Subnet Mask	Subnet Mask of the main Ethernet interface of HMI	string read only

Screen Variables

Variable Name	Description	Data Type
X Screen resolution	Display horizontal screen size in pixel	int read only
Y Screen resolution	Display vertical screen size in pixel	int read only

SD Card Variables

Variable Name	Description	Data Type
SD Card FreeSpace	Available space on card in bytes	uint64 read only
SD Card Name	Name of SD card	string read only
SD Card Size	Size in bytes of the card plugged in the slot	uint64 read only
SD Card Status	**0** = SD card unplugged **1** = SD card plugged	int read only

Element	Description

Server Variables

Variable Name	Description	Data Type
Page name	Name of current page	string read only
Current project	Name of current project	string read only
Project load time	Date when the project was loaded on the LRH SW HMI Runtime as in System Date format (milliseconds)	uint64 read only
Last operating mode start time	Seconds elapsed since device started operating mode	uint64 read only

All variables refer to server, not to LRH SW Client.

Time Variables

Variable Name	Description	Data Type
Day Of Month	Range: 1–31	int
Day of Week	Range: 0 = Sunday, .. , 6 = Saturday	int
Hour	Range: 0–23	int
Minute	Range: 0–59	int
Month	Range: 1–12	int
Second	Range: 0–59	int
System Time	The same as UTC time. It can also be set as date/time for this variable	unsignedInt
Year	Current Year	int

System time expressed in UTC format

Element	Description

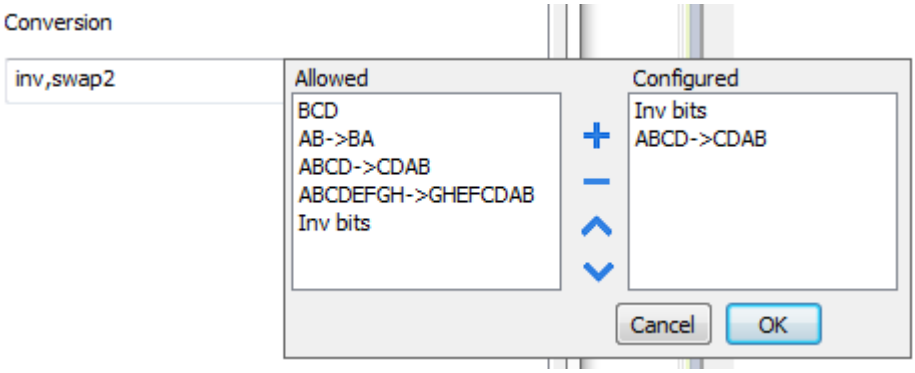
USB Drive Variables

Variable Name	Description	Data Type
USB Drive FreeSpace	Available space in bytes	uint64 read only
USB Drive Name	Name of USB device	string read only
USB Drive Size	Size in bytes of the device plugged in the USB port	uint64 read only
USB Drive Status	**0** = USB Drive unplugged **1** = USB Drive plugged	int read only

Version Variables

Variable Name	Description	Data Type
Main OS version	Version of Main OS	string read only
Runtime version	Version of Runtime	string read only
Project name	Project name	string read only
Project version	Project version	string
Project GUID	Project GUID (unique identifier)	string read only

<table>
<tr><th>Element</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th colspan="3">Virtual Com Switch Variables</th></tr>
<tr><th>Variable Name</th><th>Description</th><th>Data Type</th></tr>
<tr><td>VCS status</td><td>Provides status of VCS service.
0 = Service enabled
1 = Client connected in interleaved mode
2 = Client connected in exclusive mode
3 = Service disabled (default)</td><td>unsignedByte
read only</td></tr>
<tr><td>VCS disable</td><td>Provides manual override of VCS service.
0 = VCS service enabled
1 = VCS service disabled (default)</td><td>boolean</td></tr>
<tr><td>VCS port</td><td>Provides current listening TCP port on HMI by VCS service</td><td>unsignedShort</td></tr>
</table>
</td></tr>
<tr><td>Data Type</td><td>Each system variable has a specific data type, described in above tables.
The following table shows the details of any data type used for system variables.
<table>
<tr><th>Data Type</th><th>Memory Space</th><th>Limits</th></tr>
<tr><td>short</td><td>16-bit data</td><td>-32768 ... 32767</td></tr>
<tr><td>int</td><td>32-bit data</td><td>-2.1e9 ... 2.1e9</td></tr>
<tr><td>unsignedByte</td><td>8-bit data</td><td>0 ... 255</td></tr>
<tr><td>unsignedShort</td><td>16-bit data</td><td>0 ... 65535</td></tr>
<tr><td>unsignedInt</td><td>32-bit data</td><td>0 ... 4.2e9</td></tr>
<tr><td>uint64</td><td>64-bit data</td><td>0 ... 1.8e19</td></tr>
<tr><td>string</td><td colspan="2">Array of elements containing character code defined by selected encoding</td></tr>
</table>
</td></tr>
</table>

Element	Description
Arraysize	In case of string Tag, this property represents the maximum number of bytes available in the string Tag. Note: number of bytes corresponds to number of string chars if Encoding property is set to UTF-8 or Latin1 in Tag Editor. If Encoding property is set to UCS-2BE, UCS-2LE, UTF-16BE or UTF-16LE one char requires 2 bytes.
Conversion	Conversion to be applied to the tag.  Depending on data type selected, the list **Allowed** shows one or more conversion types.

Value	Description
Inv bits	**inv**: Invert all the bits of the tag. *Example:* 1001 → 0110 (in binary format) 9 → 6 (in decimal format)
Negate	**neg**: Set the opposite of tag value. *Example:* 25.36 → -25.36
AB -> BA	**swapnibbles**: Swap nibbles in a byte. *Example:* 15D4 → 514D (in hexadecimal format) 5588 → 20813 (in decimal format)
ABCD -> CDAB	**swap2**: Swap bytes in a word. *Example:* 9ACC → CC9A (in hexadecimal format) 39628 → 52378 (in decimal format)
ABCDEFGH ->	**swap4**: Swap bytes in a double word. *Example:*

<table>
<tr><th>Element</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>GHEFCDAB</td><td>32FCFF54 → 54FFFC32 (in hexadecimal format)
855441236 → 1426062386 (in decimal format)</td></tr>
<tr><td>ABC...NOP
->
OPM...DAB</td><td>swap8: Swap bytes in a long word.

Example:
142.366 → -893553517.588905 (in decimal format)
0 10000000110
00011100101110110110010001011010000111001010110000
1
→
1 10000011100
10101010000101000101101101101100101101100001001111
1
(in binary format)</td></tr>
<tr><td>BCD</td><td>bcd: Separate byte in two nibbles, read them as decimal (from 0 to 9)

Example:
23 → 17 (in decimal format)
0001 0111 = 23
0001 = 1 (first nibble)
0111 = 7 (second nibble)</td></tr>
</table>

Select conversion and click +. The selected item will be added to list Configured.

If more conversions are configured, they will be applied in order (from top to bottom of list Configured).

Use the arrow buttons to order the configured conversions.
</td></tr>
</table>

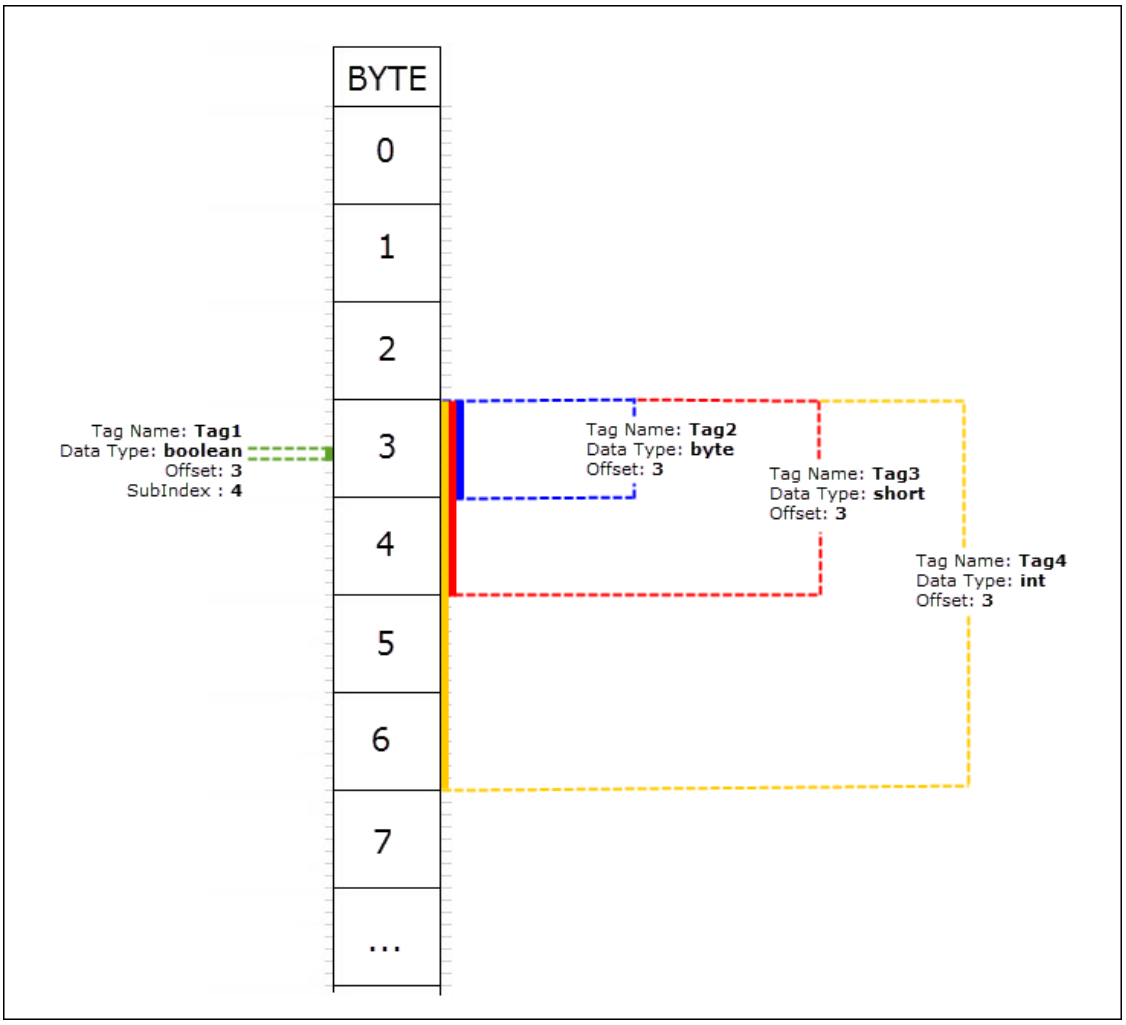
Retentive Memory variables

System Variables - Retentive Memory protocol allows to create Tags that point to a memory area whose content is maintained when HMI is powered off.

The physical support for retentive memory is based on FRAM technology.

Important: Not all HMI devices include FRAM memory. If FRAM memory is not available, persistency is supported using user memory storage (Flash or hard disk drive). Flash technology has a limitation in the maximum number of write operations. The use of Flash as storage media for retentive memory with frequent write operations may damage the memory components. Check HMI device data for availability of FRAM memory.

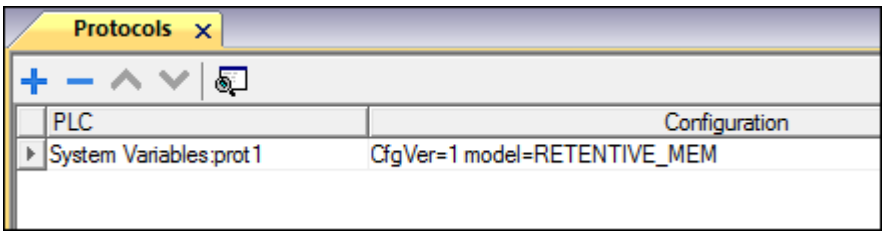
Important: Retentive memory is 16 KB flat memory area organized in bytes and accessible through an offset. Refer to schema below.



Retentive memory vs. recipes storage
Recipe data is saved in flash memory while retentive data is saved in a FRAM. Flash memory is not suitable for a high number of write operations, while FRAM supports a virtually unlimited number of write operations and should be preferred when frequent write operations are required.

Protocol Editor Settings

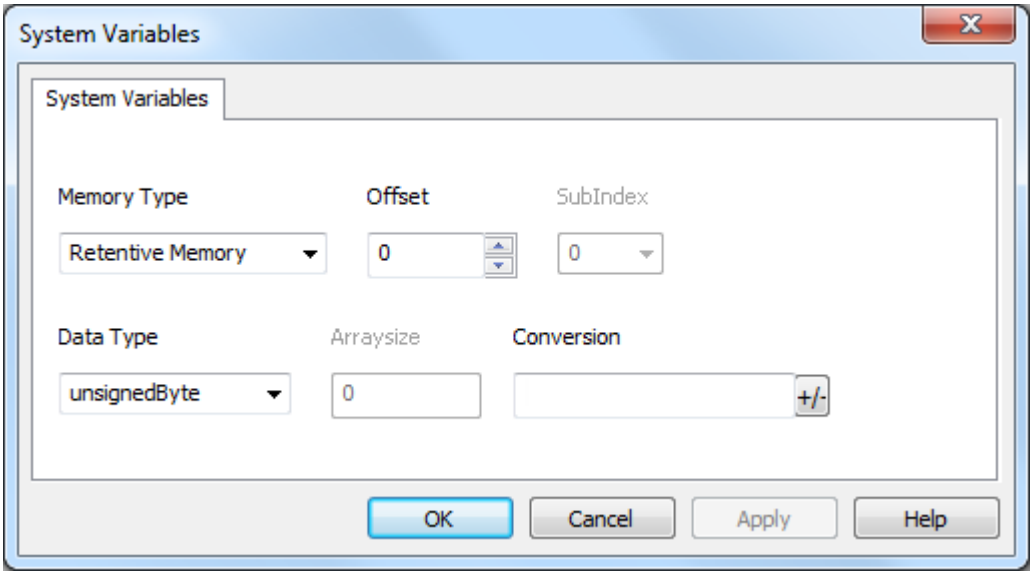
From PLC Model list of Protocol Editor dialog, select Retentive Memory.



Tag Editor Settings

Path: ***ProjectView*** *>* ***Config*** *> double-click* ***Tags***

1. To add a tag, click **+**: a new line is added.
2. Select **System Variables** from the **Driver** list: tag definition dialog is displayed.



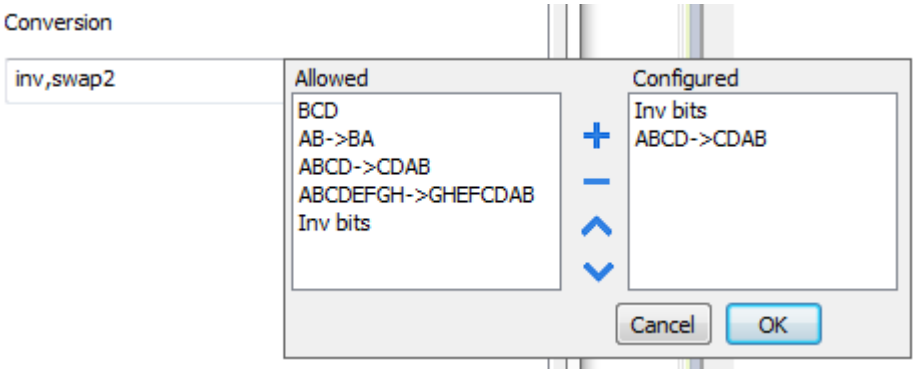
Element	Description
Memory Type	Fixed to Retentive Memory
Offset	Offset address where tag is located. Range: 0-16383
SubIndex	This parameter allows resource offset selection based on selected Data Type
Data Type	

Data Type	Memory Space	Limits
boolean	1-bit data	0 ... 1
byte	8-bit data	-128 ... 127
short	16-bit data	-32768 ... 32767
int	32-bit data	-2.1e9 ... 2.1e9
int64	64-bit data	-9.2e18 ... 9.2e18
unsignedByte	8-bit data	0 ... 255
unsignedShort	16-bit data	0 ... 65535
unsignedInt	32-bit data	0 ... 4.2e9
uint64	64-bit data	0 ... 1.8e19
float	IEEE single-precision 32-bit floating point type	1.17e-38 ... 3.4e38
double	IEEE double-precision 64-bit floating point type	2.2e-308 ... 1.79e308

<table>
<tr><th>Element</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Data Type</th><th>Memory Space</th><th>Limits</th></tr>
<tr><td>string</td><td colspan="2">Array of elements containing character code defined by selected encoding</td></tr>
<tr><td>binary</td><td colspan="2">Arbitrary binary data</td></tr>
</table>

Note: to define arrays. select one of Data Type format followed by square brackets like “byte []”, “short[]”…
</td></tr>
<tr><td>Arraysize</td><td>

In case of array tag, this property represents the number of array elements.
In case of string tag, this property represents the maximum number of bytes available in the string tag.

Note: number of bytes corresponds to number of string characters if Encoding property is set to UTF-8 or Latin1 in Tag Editor.
If Encoding property is set to UCS-2BE, UCS-2LE, UTF-16BE or UTF-16LE one character requires 2 bytes.
</td></tr>
<tr><td>Conversion</td><td>
Conversion to be applied to the tag.


Depending on data type selected, the list Allowed shows one or more conversion types.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>Inv bits</td><td>inv: Invert all the bits of the tag.

Example:
1001 → 0110 (in binary format)
9 → 6 (in decimal format)</td></tr>
<tr><td>Negate</td><td>neg: Set the opposite of tag value.

Example:
25.36 → -25.36</td></tr>
<tr><td>AB -> BA</td><td>swapnibbles: Swap nibbles in a byte.</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Element</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td></td><td>Example:
15D4 → 514D (in hexadecimal format)
5588 → 20813 (in decimal format)</td></tr>
<tr><td>ABCD -> CDAB</td><td>swap2: Swap bytes in a word.

Example:
9ACC → CC9A (in hexadecimal format)
39628 → 52378 (in decimal format)</td></tr>
<tr><td>ABCDEFGH -> GHEFCDAB</td><td>swap4: Swap bytes in a double word.

Example:
32FCFF54 → 54FFFC32 (in hexadecimal format)
855441236 → 1426062386 (in decimal format)</td></tr>
<tr><td>ABC...NOP -> OPM...DAB</td><td>swap8: Swap bytes in a long word.

Example:
142.366 → -893553517.588905 (in decimal format)
0 10000000110
00011100101110110110010001011010000111001010110000001
→
1 10000011100
10101010000101000101101101101100101101100000100111101
(in binary format)</td></tr>
<tr><td>BCD</td><td>bcd: Separate byte in two nibbles, read them as decimal (from 0 to 9)

Example:
23 → 17 (in decimal format)
0001 0111 = 23
0001 = 1 (first nibble)
0111 = 7 (second nibble)</td></tr>
</table>

Select conversion and click +. The selected item will be added to list Configured.

If more conversions are configured, they will be applied in order (from top to bottom of list Configured).

Use the arrow buttons to order the configured conversions.
</td></tr>
</table>

.

Cleaning Retentive Memory

Use the **ClearRetentiveMemory** action to clear the content of the retentive memory.

Tip: Use this action to set the memory content to a known status at any time.

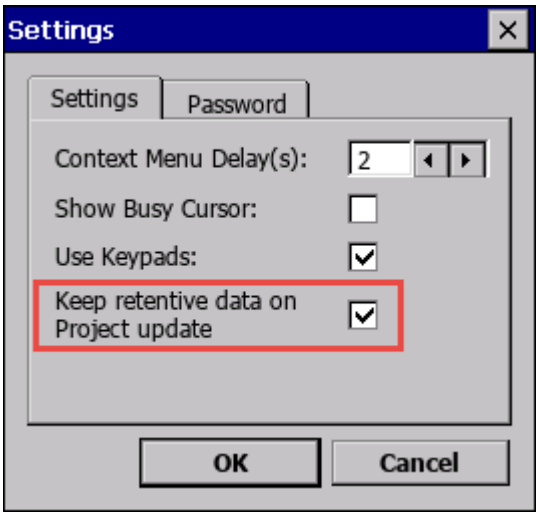
See *Actions* > *Tag Action*s section of main manual for more details.

JavaScript interface for this action is:
project.clearRetentiveMemory();

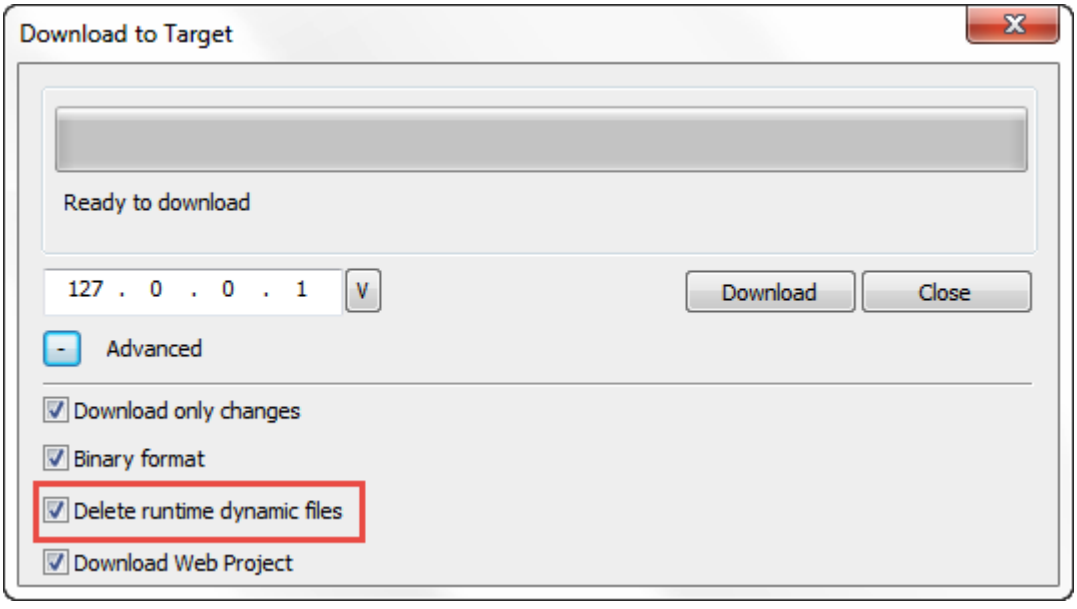
Preserving Retentive Memory at project download

When a project file is downloaded to an HMI, or when the active project is modified, the content of retentive memory is usually deleted.

If is needed to preserve the content of retentive data at project download or update, select the **Keep retentive data on project update** option in the settings tabs of the HMI device.

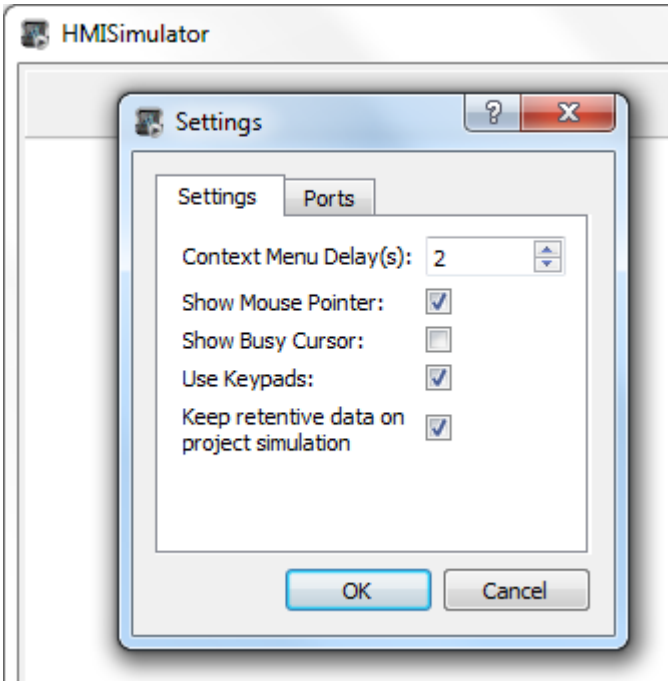


This setting will be ignored if **Delete runtime dynamic files** option is selected from *Download to Target* window.



Preserving Retentive Memory in Simulator

Simulator of LRH SW supports the retentive memory. To enable retentive memory during project simulation use the option "Keep retentive data on project simulation" in context menu.



Services variables

Services variables give the possibility to read the status and delivering commands to VNC Server.

Protocol Editor Settings

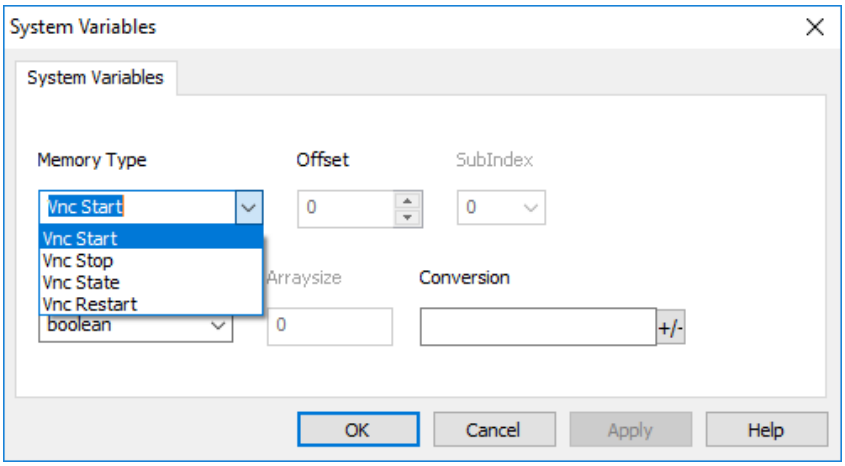
*Path: **ProjectView**> **Protocols***

1. Click **+** and select **System Variables**: the **System Variables** dialog is displayed.
2. Select **Services** from the **PLCModels** list.

Tag Editor Settings

*Path: **ProjectView**> **Config** > double-click **Tags***

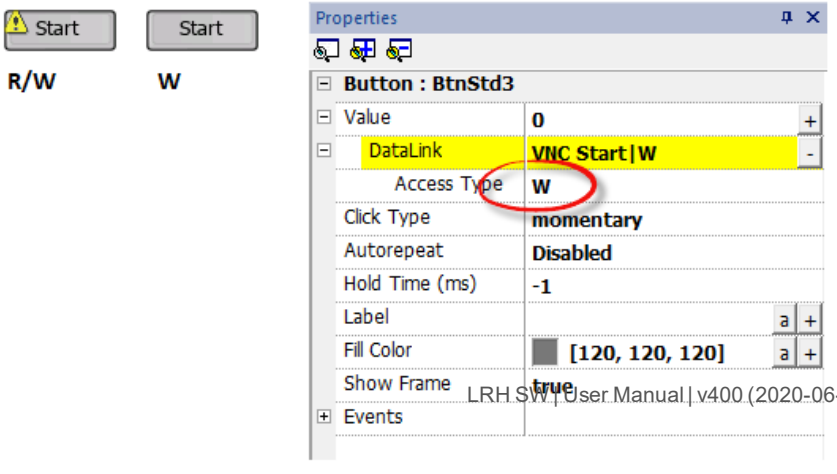
1. To add a tag, click **+**: a new line is added.
2. Select **System Variables** from the **Driver** list: tag definition dialog is displayed.



VNC status variables are supported only from Linux devices with BSP version 1.0.344 or greater. (See "HMI devices capabilities" on page 541)

Element	Description	Data type
VNC Start	Write 1 inside this tag to force the VNC server to start. This is a write only variable, command will executed any time you rewrite it.	boolean Write Only
VNC Stop	Write 1 inside this tag to force the VNC server to stop. This is a write only variable, command will executed any time you rewrite it.	boolean Write Only
VNC Restart	Write 1 inside this tag to force the VNC server to restart. This is a write only variable, command will executed any time you rewrite it.	boolean Write Only
VNC State	VNC server state 0 IDLE 10 RUNNING -1 ERROR	int Read Only

Write Only Variables cannot be read. Be sure to not use the R/W access mode to avoid the read error icon.



PLCM09 variables

PLCM09 device is a Wireless Modem with LED and digital I/O. The behavior of the related System Variables are depend on how the module has been configured inside the System Settings (see "PLCM09 Plug-in Wireless Modem" for additional information)

Protocol Editor Settings

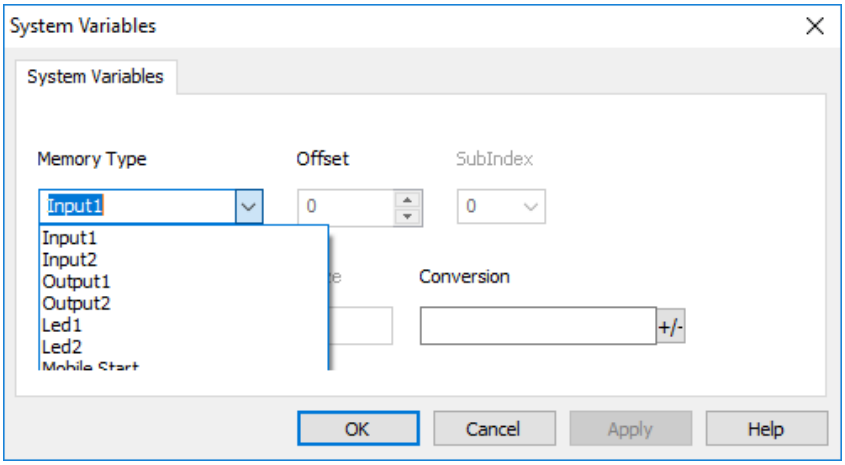
Path: ***ProjectView*** *>* ***Protocols***

1. Click **+** and select **System Variables**: the **System Variables** dialog is displayed.
2. Select PLCM09 from the **PLCModels** list.

Tag Editor Settings

Path: ***ProjectView*** *>* ***Config*** *> double-click* ***Tags***

1. To add a tag, click **+**: a new line is added.
2. Select **System Variables** from the **Driver** list: tag definition dialog is displayed.



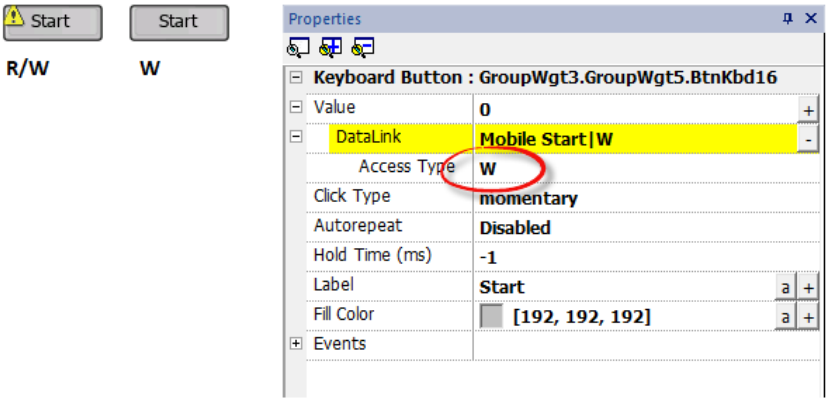
Element	Description	Data type
Input1 **Input2**	Value of the Input signals	boolean Read only
Output1 **Output2**	Value of the output signals Output variables are read/write only when configured as "User controlled" (see "PLCM09 Plug-in Wireless Modem" for additional information). In the other configurations, where output signals are controlled directly from the PLCM09 module, the system variables are read only.	boolean Read/Write
Led1 **Led2**	Value of the LED status • 0 = Off • 1 = On • 2 = Blink	unsignedByte Read/Write

Element	Description	Data type
	LED variables are read/write only when configured as "User controlled" (see "PLCM09 Plug-in Wireless Modem" for additional information). In the other configurations, where LED status are controlled directly from the PLCM09 module, the system variables are read only.	
Mobile Start	Write 1 inside this tag to force the wireless mode to start. This is a write only variable, command will executed any time you rewrite it.	boolean Write Only
Mobile Stop	Write 1 inside this tag to force the wireless mode to stop. This is a write only variable, command will executed any time you rewrite it.	boolean Write Only
Mobile Restart	Write 1 inside this tag to force the wireless mode to restart. This is a write only variable, command will executed any time you rewrite it.	boolean Write Only
Mobile State	Mobile connection state 0 IDLE 1 STARTING 10 RUNNING 100 CONNECTING 200 CONNECTED 300 STOPPING -1 GENERICERROR -10 SYSTEMERROR -100 MODEMNOTFOUND -101 MODEMBUSY -110 MODEMCOMM -120 MODEMTIMEOUT -130 MODEMERROR -200 SIMMISSING	int Read Only

Element	Description	Data type
	-300 PINREQUIRED -301 NEWPINREQUIRED -310 PINERROR -320 PUKREQUIRED -330 PUKERROR -400 ROAMINGBLOCKED -500 BADCREDENTIALS	
Mobile Signal	Mobile signal quality (0-100) Value of the signal detected when the device is started	byte Read Only
Mobile Operator	Mobile operator name (e.g. 'Vodafone')	string[8] Read Only
Mobile Access Technology	Mobile access technology -1 N/A 0 GSM (2G) 2 UTRAN (2G) 3 GSM W/EGPRS (2G) 4 UTRAN W/HSDPA (3G) 5 UTRAN W/HSUPA (3G) 6 UTRAN W/HSDPA and HSUPA (3G)	int Read Only
Mobile Registration Status	Mobile registration status -1 N/A 0 Not registered. Wireless Modem is not currently searching a new operator to register. 1 Registered on home network. 2 Not registered. Wireless Modem is currently searching a new operator to register.	int Read Only

Element	Description	Data type
	3 Registration denied. 4 Unknown 5 Registered on roaming	
Mobile RX/TX	Number or received/transmitted bytes	unsignedInt[2] Read Only
Mobile Start Time (Sec)	When mobile connection was started (in seconds since epoch)	unsignedInt Read Only

Write Only Variables cannot be read. Be sure to not use the R/W access mode to avoid the read error icon.



JavaScript (Mobile Connection State)

The mobile connection state can be retrieved even from the below JavaScript interface where the "protocolSysVar" is the protocol instance code (e.g. "prot1", "prot2", etc.)

```
Mobile_State = tagMgr.invokeProtocolCommand(protoSysVar,"get_mobile_state", ""); //
get modem status
```

14 Actions

Actions are functions used to interact with the system and are normally executed when events are triggered.

Events can be triggered by various widgets, for example on press and on release of a button. Not all actions are available for all the events of an object.

Actions are linked to widgets in the **Event** section of the Property pane (Page Editor).

Alarm actions

Mainly used to acknowledge or reset alarms.

SelectAllAlarms

Selects all alarms.

Parameter	Description
Mode	**TOGGLE** Reverses the select status. Alarms that are not triggered or have no pending acknowledge or reset requests will never be selected. **SELECT** Selecting all alarms that are triggered or that have acknowledge or reset request pending **UNSELECT** Unselect all alarms

SelectAlarm

Select a specif alarm.

Parameter	Description
AlarmID	Alarm ID
Selection Flag	**TRUE** Select the alarm. Alarms that are not triggered or have no pending acknowledge or reset requests will not selected. **FALSE** Unselect the alarm.

AckAlarm

Acknowlege a specific alarm or all selected alarms.

Parameter	Description
AlarmID	Specific Alarm ID **SELECTED** All selected alarms

Acknowledges selected alarms.

ResetAlarm

Resets a specific alarm or all selected alarms that are not triggered and acknowledged.

Parameter	Description
AlarmID	Specific Alarm ID **SELECTED** All selected alarms

EnableAlarms

Enable or disable a specific alarm or all selected alarms.

Disabled alarms will not generate alarm events.

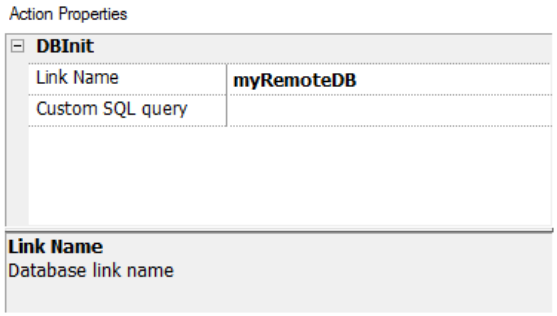
Parameter	Description
AlarmID	Specific Alarm ID **SELECTED** All selected alarms
Selection Flag	**TRUE** Enable the alarm(s). **FALSE** Disable the alarm(s).

Database actions

DBInit

Important: This action is used only once on an empty database. It is not an initialization command to be called any time the HMI device starts.

Creates the set of tables required by the project. You do not need to use this action if the database already contains the necessary tables.

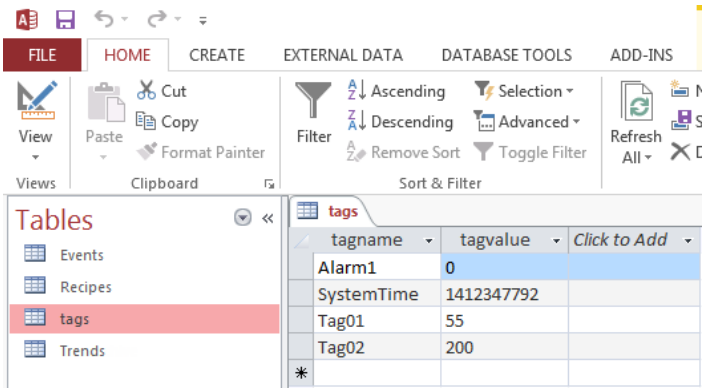


Use **Custom SQL query** parameter to define the pages to be created. Leave empty to generate default table names

Tip: Add this command inside a SetUp page of your project, used by authorized personal only when installing the application for the first time.

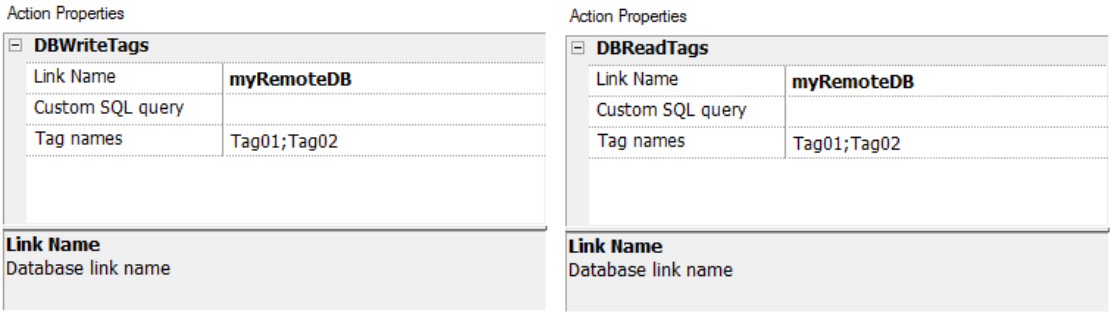
JavaScript Interface

```
project.dbInit(dbLinkName, sqlCustomQuery);
```



DBWriteTags, DBReadTags

Transfer the values of the selected tags to/from the remote database.



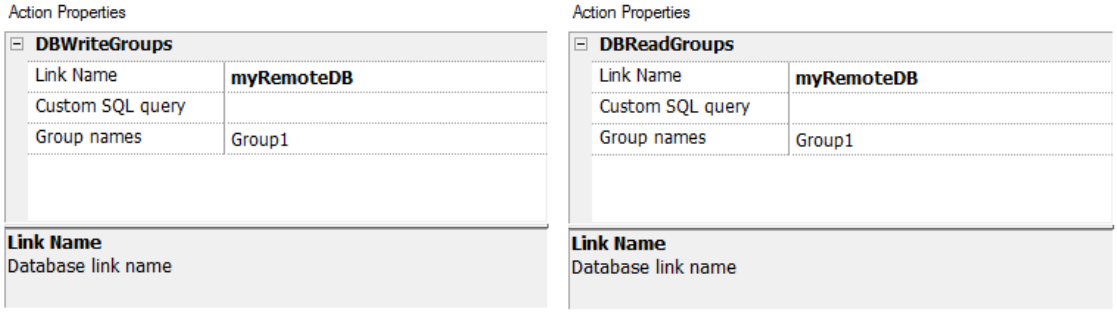
JavaScript Interface

```
project.dbWriteTags(dbLinkName, sqlCustomQuery, Tags);
```

```
project.dbReadTags(dbLinkName, sqlCustomQuery, Tags);
```

DBWriteGroups, DBReadGroups

Transfer groups of tags between the HMI device and the database.



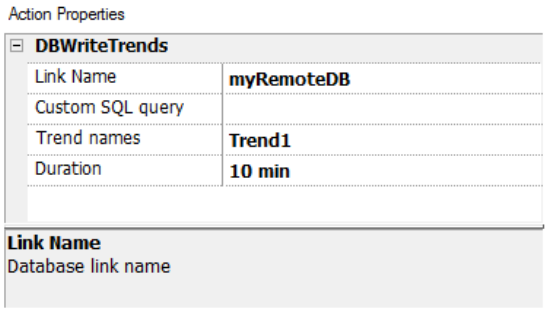
JavaScript Interface

```
project.dbWriteGroups(dbLinkName, sqlCustomQuery, Groups);
```

```
project.dbReadGroups(dbLinkName, sqlCustomQuery, Groups);
```

DBWriteTrend

Inserts the values of the last data sampled in the selected range of time inside the Trends table of the remote database.

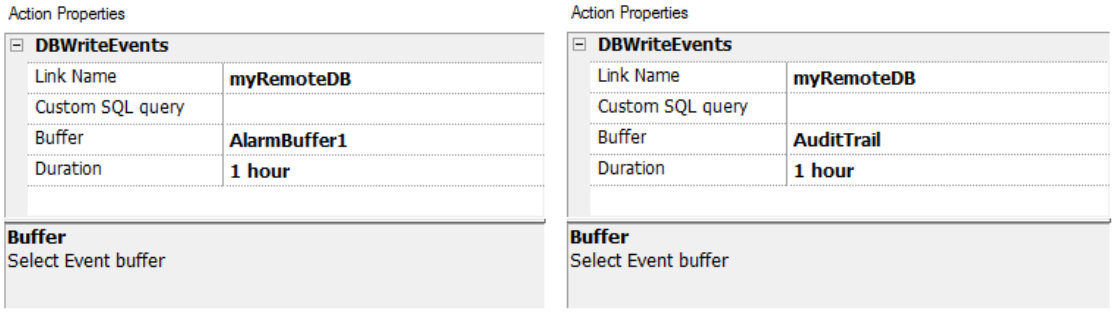


JavaScript Interface

```
project.dbWriteTrends(dbLinkName, sqlCustomQuery, trendName, durationIndex)
```

DBWriteEvents

Inserts the values of the last events in the selected range of time inside the Events table of the remote database.

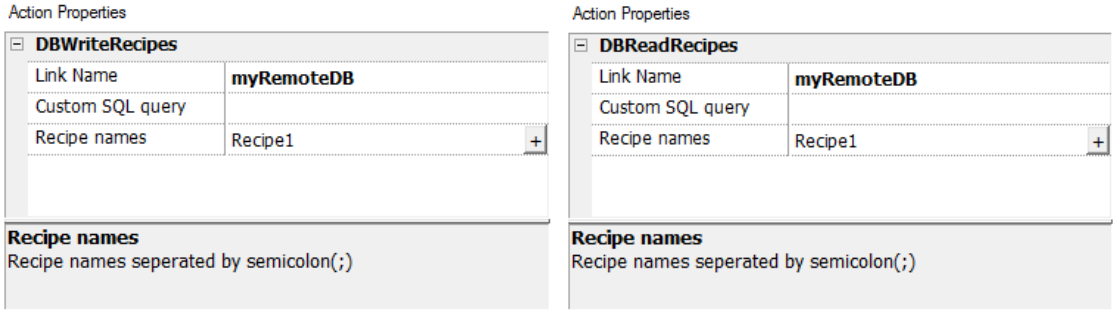


JavaScript Interface

```
project.dbWriteEvents (dbLinkName, sqlCustomQuery, archiveName, durationIndex)
```

DBWriteRecipes, DBReadRecipes

Transfer the recipe data to/from the remote database.



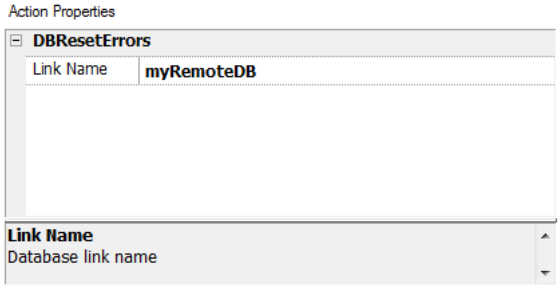
JavaScript Interface

```
project.dbWriteRecipes(dbLinkName, sqlCustomQuery, recipeNames)
```

```
project.dbReadRecipes(dbLinkName, sqlCustomQuery, recipeNames)
```

DBResetErrors

Reset all the three status variables of the selected database link. "Database variables" on page 1.



JavaScript Interface

```
project.dbResetErrors(dbLinkName)
```

Event actions

Used by Alarm History widget to scroll events/alarms backward/forward in table view (event buffer widget).

ScrollEventsBackward

Scrolls events/alarms backward in table view (event buffer widget).

ScrollEventsForward

Scrolls events/alarms forward in table view (event buffer widget).

MultiLanguage actions

Selects the application language.

SetLanguage

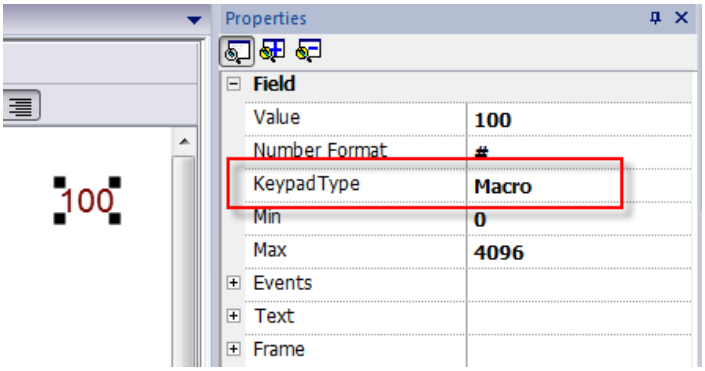
Sets the language used. The selected language will be applied at runtime to all applicable widgets.

Keyboard actions

Changes the use of keypads.

SendKey

Sends one character to a numeric widget. The **KeypadType** property of the numeric widget must be set as **Macro**.

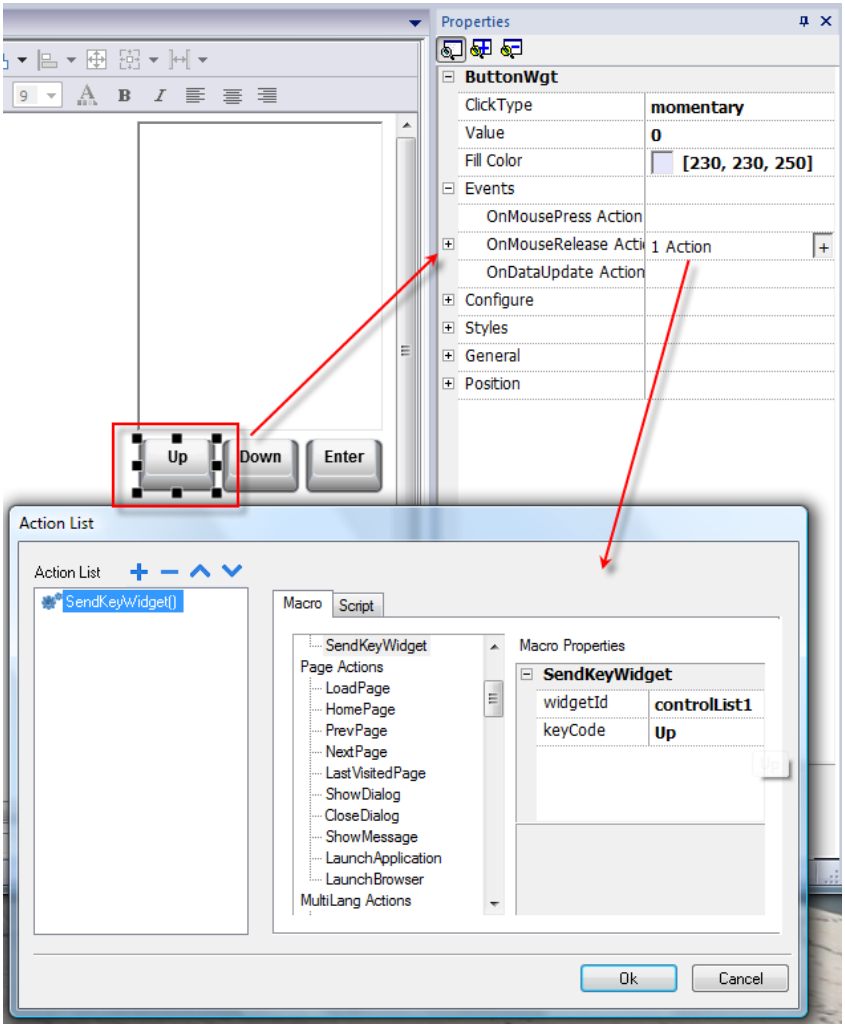


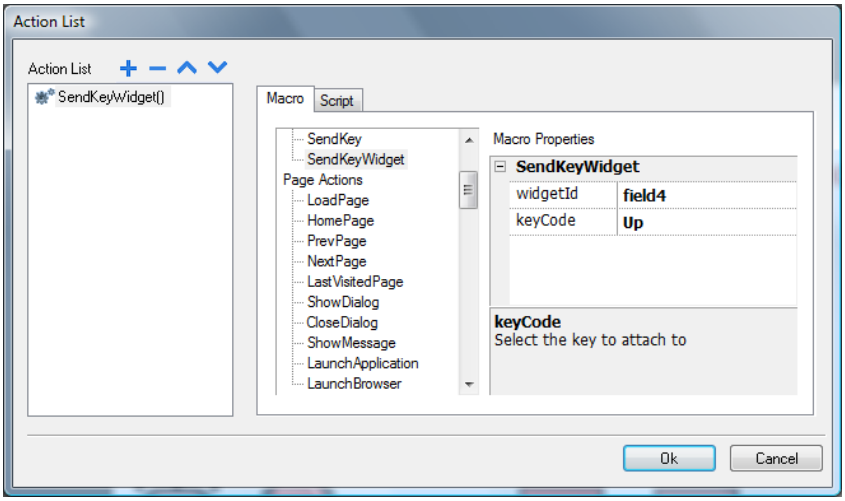
SendKeyWidget

Sends one character to a specific widget.

Example

The **Up** and **Down** buttons use the **SendKeyWidget** action in association with the **Control List Widget**.





ShowKeyPad

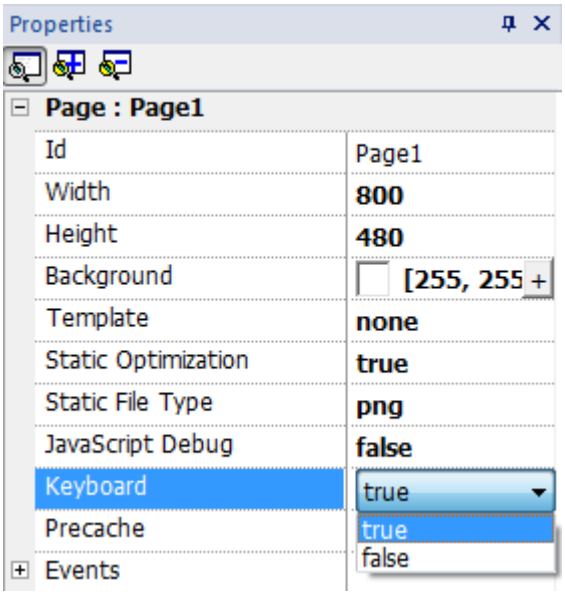
Shows the default operating system touch keypad.

Note: might not be supported by all operating systems.

Keyboard

Enables/disables the use of actions when using external keyboards. Action execution can be enabled/disabled both at project and at page level.

The effect is equivalent to the use of the property Keyboard for project and page.



Media Player actions

Interact with the Media Player widget at runtime.

Action	Description
PlayMedia	Starts playing the video.
StopMedia	Stops the video.
ReloadMedia	Restarts video from the beginning.
PauseMedia	Pauses the video.
BrowseMedia	Selects the video to play.

FTP actions

Used to upload and download files to and from a remote FTP server.

ftpGET

Download files from a remote FTP server

Parameter	Description
FtpConfig	Configuring the FTP parameters
FtpRemoteFileName	File name on the remote FTP server to download (source)
FtpLocalFileName	File name on local HMI device (destination)

ftpPUT

Upload files to a remote FTP server

Parameter	Description
FtpConfig	Configuring the FTP parameters
FtpLocalFileName	File name on local HMI device (source)
FtpRemoteFileName	File name on the remote FTP server to download (Destination)

Filenames can contain wildcards.

When transferred, system variables are updated with the status of ongoing operations (see "FTP client variables" on page 131for details).

FTP Server Configuration

To configure the FTP parameter, enter the following information for the **FtpConfig** setting:

Parameter	Description
FTP Address	FTP server IP Address
Server Port	Port for FTP connection (default = 21).
Authentication	Select the FTP authentication to use: • Normal (Username and password required) • Anonymous
User Name	Username of the remote FTP account
Password	Password of the remote FTP account

Click **+** to add more FTP servers configuration.

Tip: Use tags if you want change the server parameters dynamically from the LRH SW HMI Runtime.

FTP JavaScript Interface

ftpConfig

```
ftpCONFIG (IPAddress, Port, Authentication, UserName, Password)
```

Set the FTP parameters to use on next FTP calls

Parameter	Description
IPAddress	FTP server IP Address.
Port	Port for FTP connection (default = 21).
Authentication	Select the FTP authentication to use: • Normal (Username and password required) • Anonymous
UserName	Username of the remote FTP account
Password	Password of the remote FTP account

ftpGET

```
ftpGET (remoteFileName, localFileName, [callback])
```

Download files from a remote FTP server

Parameter	Description
remoteFileName	File name on the remote FTP server to download (source)
localFileName	File name on local HMI device (destination)
callback	Function that will be call at the end of the FTP transfer

ftpPUT

```
ftpPUT (remoteFileName, localFileName, [callback])
```

Upload files to a remote FTP server

Parameter	Description
remoteFileName	File name on the remote FTP server to download (source)
localFileName	File name on local HMI device (destination)
callback	Function that will be call at the end of the FTP transfer

Example:

```
project.ftpCONFIG("192.168.0.200", "21", "true", "admin", "admin");
```

```
project.ftpGET( "data.txt",
                "\\USBMemory\\data.txt",
                function(ftpStatus) {fnFtpGetFinished(ftpStatus);} );

function fnFtpGetFinished(ftpStatus) {
        alert(ftpStatus);
}
```

Page actions

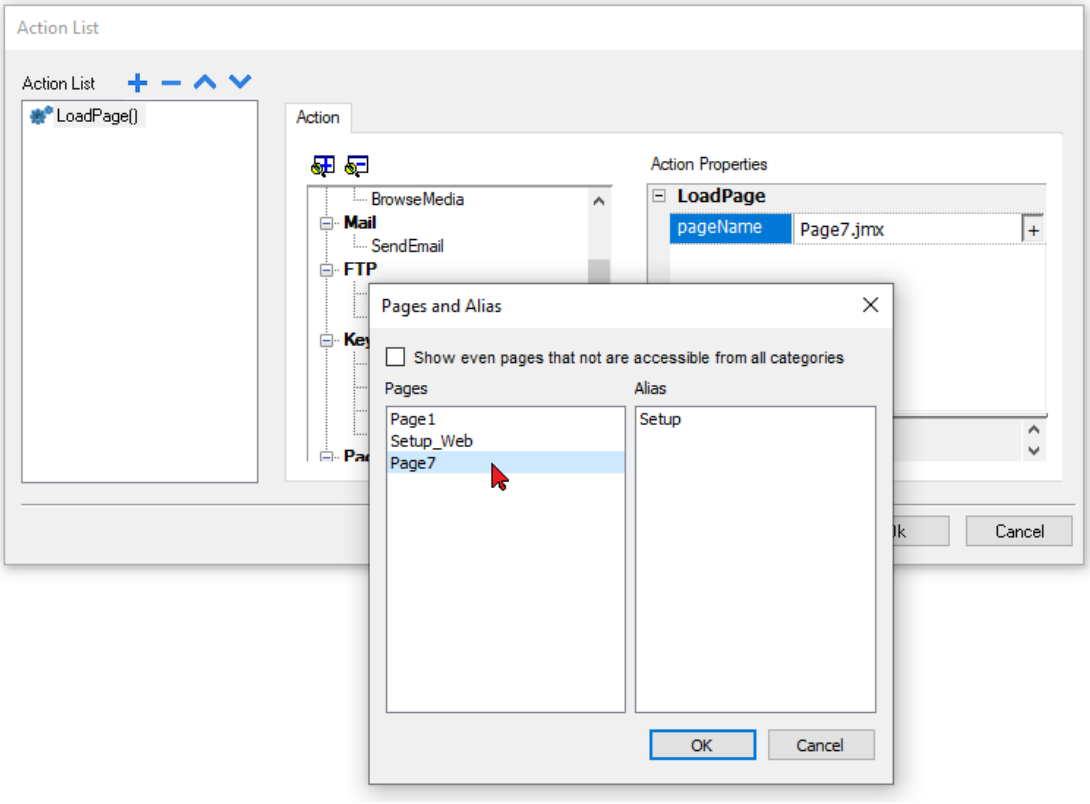
Page navigation. Page actions can be used with the following events:

- OnMouseClick,
- OnMouseRelease,
- OnMouseHold
- OnActivate
- OnDeactivate
- Alarms
- Schedulers.

LoadPage

Go to the selected page of the project.

Starting from LRH SW v4.0 in addition to the pages you can use the aliases (see "Alias pages" on page 69)



HomePage

Go to the home page.

You can set the home page in the **Behavior** section of the **Project Widget**, see "Project" on page 80

PrevPage

Go to the previous page.

NextPage

Go to the next page.

LastVisitedPage

Go to the previously displayed page

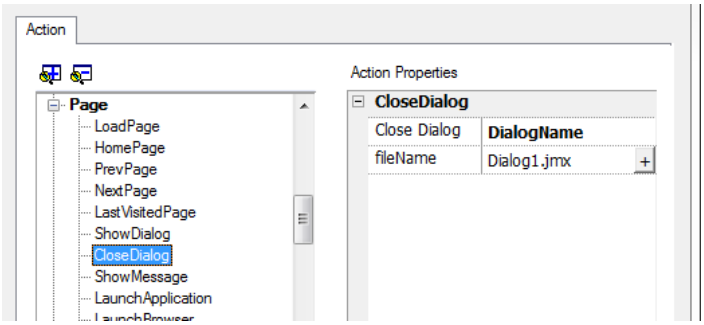
ShowDialog

Opens a dialog page defined in the project.

CloseDialog

Close dialog pages.

Note: This action is applicable only to dialog pages.



CloseDialog options

Option	Description
All	Closes all open dialogs
Selected	Closes only active dialog
DialogName	Closes dialog specified as **fileName** property

JavaScript Interface

project.closeDialog(DialogID);

Where *DialogID*:

All	Closes all open dialogs
Selected	Closes only active dialog
DialogName.jmx or AliasName	Closes dialog specified as **fileName** parameter

Examples

Example	Behavior
project.closeDialog("All");	All open dialogs are closed
project.closeDialog("Selected");	The selected dialog is closed
project.closeDialog("Dialog1.jmx");	All instances of Dialog1 are closed

The function *project.closeDialog();* without parameter works as *project.closeDialog("Selected");*.

ShowMessage

Displays a popup message. Enter the text of the message to be displayed.

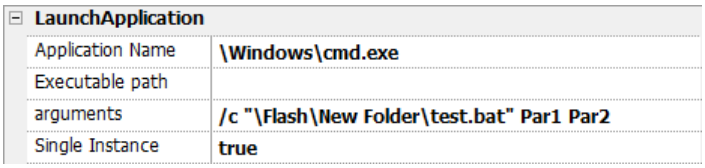
LaunchApplication

Launches an external application.

Parameter	Description
App Name	Executable name with extension (for example, "notepad.exe" to run Notepad)
Path	Application path.
Arguments	Application specific arguments (for example, *\flash\qthmi\Manual.pdf* to open the document “Manual.pdf”)
Single Instance	Argument to start the application in a single instance or multiple instances. When single instance is selected, the system first verifies whether the application is already running; if so, then the application is brought to the foreground, if not, then the application is launched.
FlushRuntimeCache	Flush all runtimes cache to free as more ram as possible before running the application.

Note: Arguments with spaces must be quoted (for example, "\Storage Card\Manual.pdf")

Example:

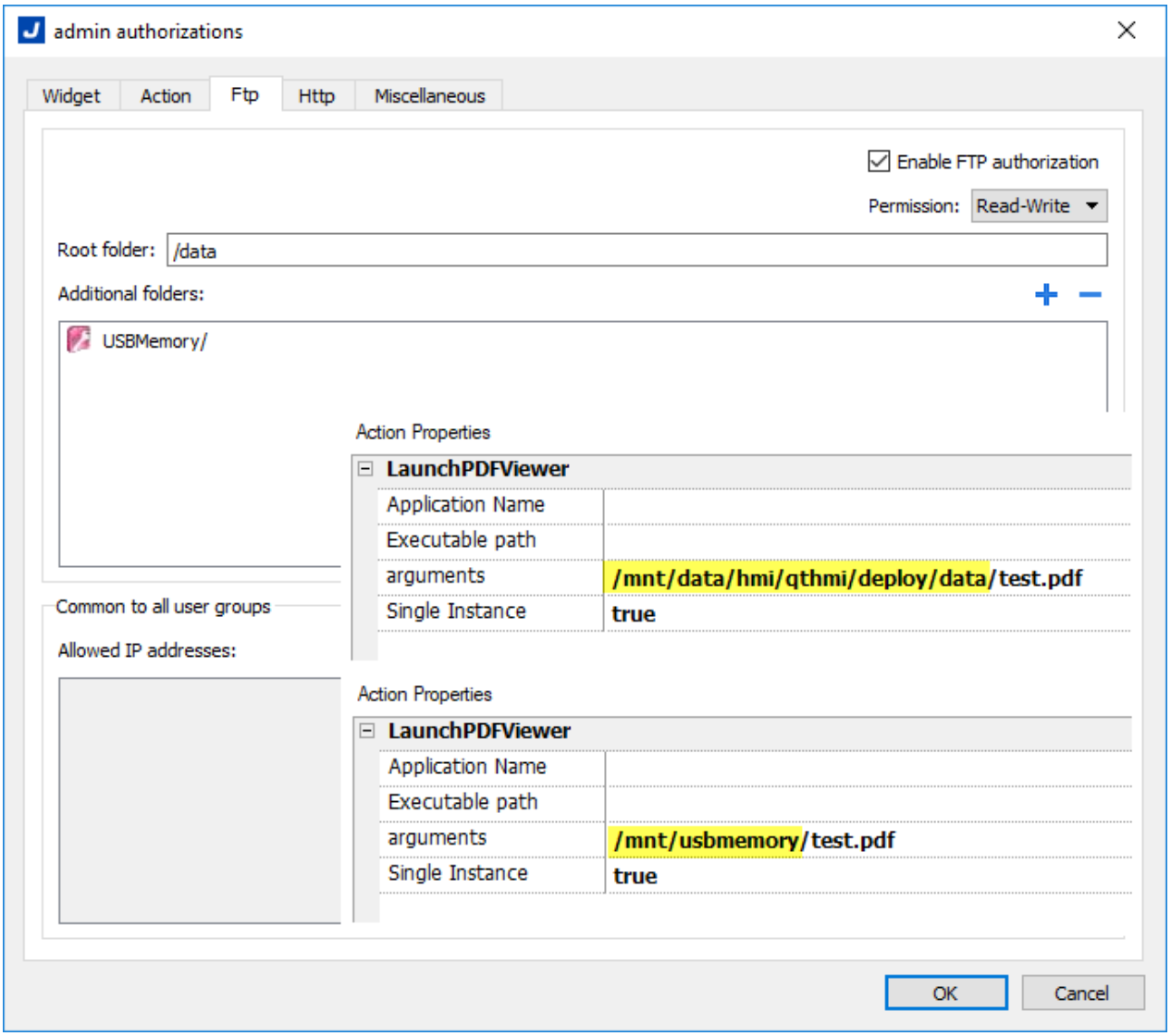


LaunchPDFViewer

Starts PDF Viewer.

Note that the pathname of the arguments field uses native OS format (see "HMI devices capabilities" on page 541).

On **Linux devices**, the HMI application is installed on path /mnt/data/hmi/qthmi/deploy/ and pathname's syntax use the slash character.



-hide-open-button (available only on Linux devices)

Using this option, the icon to open a different file will be removed from the PDF toolbar (to restrict navigation to PDF file already opened and passed via command line).

LaunchUpdater

Updates project and runtime from an external device.

Use **Path** parameter to specify the folder that will contain the update package file. Leave the path parameter empty if you prefer select the file manually on the HMI device when the macro is invoked.

When the LaunchUpdater macro is executed, the below dialog is showed on HMI device

JavaScript Interface

project.launchUpdater(strPath)

Examples

```
project.launchUpdater("\\USBMemory")
```

LockScreen

Temporarily locks the touch screen. Allows cleaning the touch screen.

The system variable **Time remaining to unlock** displays the time remaining to unlock.See "Screen variables" on page 134

LoadProject

Unload current project and load the selected project inside the HMI device.

The project name has to be specified using relative path, as for the below example:

LastVisitedProject

Unload current project and return to previous project

Print actions

Manages print tasks.

PrintGraphicReport

Prints a graphic report.

Parameter	Description
reportName	Assigns a name to the report
silent	**false** = allows to set printer properties at runtime
fileName	File name (available only for PDF reports) Supported placeholders: • %n = Report name • %p = Project name • %y = Year, %M = Month, %d = Day • %h = Hour, %m = Minutes, %s = Seconds.
folderPath	Folder Path (available only for PDF reports) Note that the pathname of the arguments field uses native OS format (see "HMI devices capabilities" on page 541). • On Linux devices Path for USB Device is "/mnt/usbmemory" "testFolder" will be inside "/mnt/data/hmi/qthmi/deploy/testFolder"
Signed	When the output is a PDF file, generate a signed file using the x.509 certificate of the panel. **On Linux devices, the BSP v1.0.507 or greater is required** The algorithm to use to signing is defined inside the project properties parameters See "Project" on page 80 for the available algorithms See also: • "Signed PDF files" on page 320

EmptyPrintQueue

Flushes the current printing queue. If executed while executing a job, the queue is cleared at the end of the job.

PausePrinting

Puts the current printing queue on hold. If executed while executing a job, the queue is paused at the end of the job.

ResumePrinting

Restarts a queue previously put on hold.

AbortPrinting

Stop the execution of the current job and removes it from the queue. If the queue has another job, then, after aborting, the next job starts.

Recipe actions

Used to program recipe management.

DownLoadRecipe

Copy recipe data from HMI device flash memory to the controller (e.g. PLC, local variable, depending on the protocol).

Parameter	Description
RecipeName	Name of recipe to download
RecipeSet	Number of recipe set to copy. **curSet** = download currently selected recipe set

UpLoadRecipe

Saves recipe data from the controller (e.g. PLC, local variable, depending on the protocol) to the device Flash Memory.

Parameter	Description
RecipeName	Name of recipe to upload
RecipeSet	Number of recipe set to copy. **curSet** = upload currently selected recipe set

WriteCurrentRecipeSet

Sets the selected recipe as current recipe set.

Parameter	Description
RecipeName	Name of recipe to set as current recipe
RecipeSet	Recipe set to define as current recipe set

DownLoadCurRecipe

Downloads current set of recipe data to the controller.

No parameter is required.

UploadCurRecipe

Uploads set of controller data to current recipe set.

No parameter is required

ResetRecipe

Restores factory settings for recipe data. Original recipe data will overwrite uploaded recipes

Select the recipe that you want to reset to factory data.

DumpRecipeData

Dumps recipe data to internal or external storage. Data is saved in .csv format.

Parameter	Description
RecipeName	Name of recipe to dump
FilePath	Destination folder • Internal = *\Flash\QTHMI\workspace\Dump* • USB drive = *\USBMemory* • SD Card = *\Storage Card* • Public Network = *\\<hostname or IP>\sharePath* • Private Network = *\\<username>:<password>@<hostname or IP>\sharePath*  Note: supported formats for external memory are FAT or FAT32 (NTFS format is not supported).  Note: Private networks are supported only from Linux devices with BSP 1.0.25 and above.
FileName	Tag that specifies a filename. The below wildcards are supported • %r = Recipe name • %d = Dataset name Example: %r_%d
DateTimePrefixFileName	**true** = the dumped file will have date and time as prefix to its name (for example D2012_01_01_T10_10_recipe1.csv)
TimeSpec	Time format: • **Local** = the time values exported are the time of the HMI device. • **Global** = the time values exported are in UTC format.

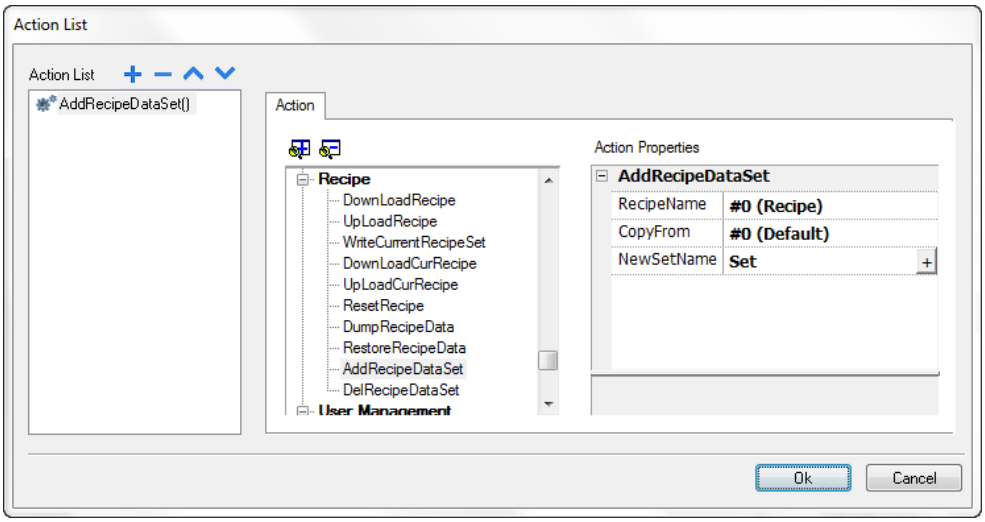
RestoreRecipeData

Restores previously saved recipe data.

Parameter	Description
RecipeName	Recipes to restore: • AllRecipes Data of all recipes will replaced with the data read from the external file • CurrentRecipe Only the data of the current selected recipe will replaced with the data read from the external file
RecipeDataSet	Available only when RecipeName=CurrentRecipe. Select the data sets to restore: • AllRecipeDataSet All data set will restored • curSet Only the data set of the current selected data set will restore
Restore Type	Available only when RecipeDataSet=AllRecipeDataSet. This parameter define the behavior when the numbers of data sets inside the file to restore is not matching with the data set number inside the HMI device • Replace All data sets that are inside the device are removed and replaced with the data sets from the csv file • Match Replace only the data set inside the device that have the same data set id • MatchAndAdd Replace the data set inside the device that have the same data set id and add the additional data set found inside the csv file (Note: data sets that are inside the device but not inside the csv file are not removed from the device)
FilePath	Source folder • Internal = *\Flash\QTHMI\workspace\Dump* • USB drive = *\USBMemory* • SD Card = *\Storage Card* • Public Network = *\\<hostname or IP>\sharePath* • Private Network = *\\<username>:<password>@<hostname or IP>\sharePath* Note: supported formats for external memory are FAT or FAT32 (NTFS format is not supported). Note: Private networks are supported only from Linux devices with BSP 1.0.25 and above.
FileName	Attached tag from which read the file name at runtime.
BrowseForFile	**true** = shows the Open dialog to browse the file to read. **false** = no dialog is shown,

AddRecipeDataSet

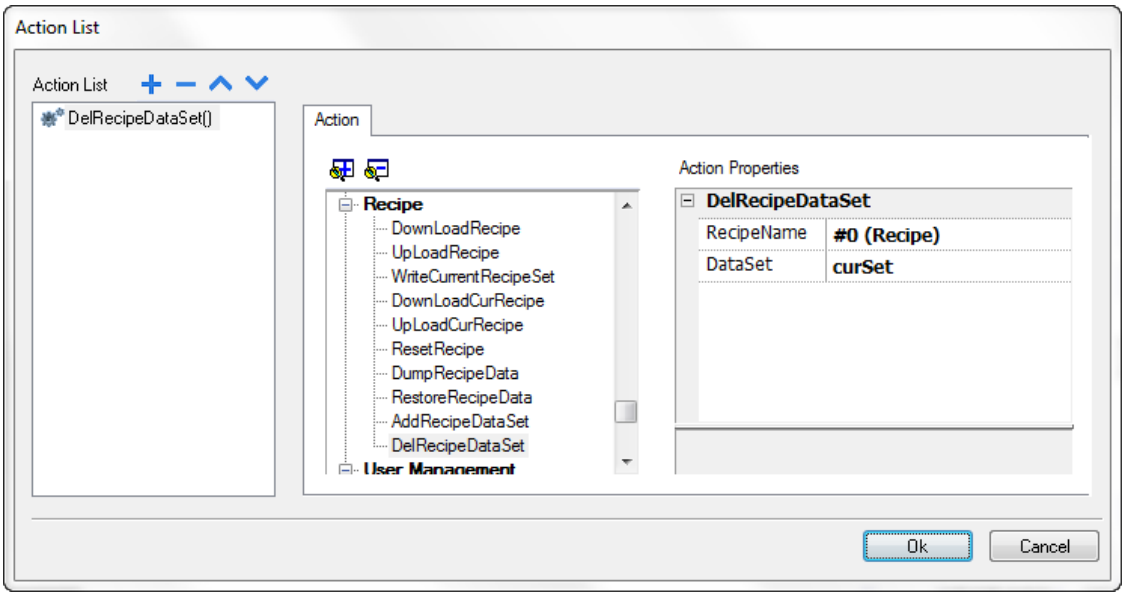
Adds a new dataset to the selected recipe. The new dataset is appended at the end of the already defined datasets.



Parameter	Description
RecipeName	Recipe where the dataset is added.
CopyFrom	Dataset from where parameters values are copied from to initialize the new dataset
NewSetName	Name of new dataset. Here you can you can use a tag reference.

DelRecipeDataSet

Deletes a dataset from the selected recipe. Deleting a dataset will rearrange the position number of the datasets that follow.



Parameter	Description
RecipeName	Recipe where the dataset is to be deleted.
DataSet	Dataset to be deleted.

Remote Client actions

Used to upload and download files to and from a remote HMI device. These actions can only be used from a remote LRH SW Client to access remote files via FTP.

Important: Enable FTP support and give all necessary user rights to the folders used to transfer files.

UploadToHMI

Opens a file Open dialog to select a file to be uploaded to the remote HMI device.

Parameter	Description
Destination	Destination path on HMI device for file upload
Filter	File extensions of the files to be displayed separated by commas (for example, *.txt)

DownloadFromHMI

Opens a file Open dialog to select a file to be downloaded from the remote HMI device.

Note: Only files matching the set filter are displayed and can be downloaded.

Parameter	Description
Source	Source path on the HMI device for file download
Filter	File extensions of the files to be displayed separated by commas (for example, *.txt)

JavaScript Interface

```
boolean project.uploadToHMI(dirPath, strFilter);
```

```
boolean project.downloadFromHMI(dirPath, strFilter);
```

Parameter	Description
dirPath	Source path on the HMI device for file download/upload
strFilter	File extensions of the files to be displayed separated by commas (for example, *.txt)

Return values:

True	Transfer successful
False	Transfer failed

Note: When transferred, system variables are updated with the status of ongoing operations.

System actions

Used to manage system properties.

Restart

Restarts the runtime.

DumpTrend

Stores historical trend data to external drives (USB drive or SD card).

Parameter	Description
TrendName	Name of historical trend to store
FolderPath	Destination folder: • Internal = *\Flash\QTHMI\workspace\Dump* • USB drive = *\USBMemory* • SD Card = *\Storage Card* • Public Network = *\\<hostname or IP>\sharePath* • Private Network = *\\<username>:<password>@<hostname or IP>\sharePath* Note: supported formats for external memory are FAT or FAT32 (NTFS format is not supported). Note: Private networks are supported only from Linux devices with BSP 1.0.25 and above.
FileFormat	**Binary** = the buffer is dumped in binary format (a .dat file and .inf file). Both these files are then required to convert data in .csv format by an external utility. **Compatibility CSV** = the buffer is dumped to the specified location as a .csv file format compatible with versions 1.xx **Compact CSV** = the buffer is dumped to the specified location as a .csv file using a newer format See "Exporting trend buffer data" on page 264
DateTimePrefix	**true** = the dumped file will have date and time as prefix to its name (for example D2012_01_01_T10_10_Trend1.csv)

Parameter	Description
TimeSpec	Time format: • **Local** = the time values exported are the time of the HMI device. • **Global** = the time values exported are in UTC format.
FileName	Enabled when the DateTimePrefixFileName=true The below wildcards are supported • %n = Trend name • %y = Year • %M = Month • %d = Day • %h = Hour • %m = Minutes • %s = Seconds Example: \%n\%y%M%d\%h%m%s

Additional parameters available only when the selected FileFormat is **Compact CSV**

When both “Select Fields” and “Select Curves” parameters are empty, the .csv file is dumped in the old “Compact CSV” without columns' selection format. See also "Exporting trend buffer data" on page 264

Parameter	Description
Select Fields	Select the columns to export inside the dumped file. Available columns are: • Date and Time • Date • Time • Value • Quality Note that "Attach to tag" can be used to define columns to be exported at the runtime from the HMI application. The tag must contain a string with the list of fields to be exported separated by commas. Example: • "" *(Empty string = all available fields)* • "DateTime,Value,Quality" • "Date,Time,Value"
Select Curves	Select the curves to export inside the dumped file Note that "Attach to tag" can be used to define curves to be exported at the runtime from the HMI application. The tag must contains a string with the list of curve names to be exported separated by commas.

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td></td><td>Example:
Empty string or "All curves" will export all datasets
"Name1,Name2,Name3"
"Name1,Name3"</td></tr>
<tr><td>Date Format</td><td>Select the Date and Time format
Using "Attach to tag" is possible define the date format at runtime through a string
<table><tr><th colspan="2">Date Placeholder</th></tr><tr><td>d</td><td>The day as number without a leading zero (1 to 31)</td></tr><tr><td>dd</td><td>The day as number with a leading zero (01 to 31)</td></tr><tr><td>ddd</td><td>The abbreviated localized day name (e.g. 'Mon' to 'Sun')</td></tr><tr><td>dddd</td><td>The long localized day name (e.g. 'Monday' to 'Sunday').</td></tr><tr><td>M</td><td>The month as number without a leading zero (1-12)</td></tr><tr><td>MM</td><td>The month as number with a leading zero (01-12)</td></tr><tr><td>MMM</td><td>The abbreviated localized month name (e.g. 'Jan' to 'Dec').</td></tr><tr><td>MMMM</td><td>The long localized month name (e.g. 'January' to 'December').</td></tr><tr><td>yy</td><td>The year as two digit number (00-99)</td></tr><tr><td>yyyy</td><td>The year as four digit number</td></tr></table><table><tr><th colspan="2">Time Placeholder</th></tr><tr><td>h</td><td>The hour without a leading zero (0 to 23 or 1 to 12 if AM/PM display)</td></tr><tr><td>hh</td><td>The hour with a leading zero (00 to 23 or 01 to 12 if AM/PM display)</td></tr><tr><td>m</td><td>The minute without a leading zero (0 to 59)</td></tr><tr><td>mm</td><td>The minute with a leading zero (00 to 59)</td></tr><tr><td>s</td><td>The second without a leading zero (0 to 59)</td></tr><tr><td>ss</td><td>The second with a leading zero (00 to 59)</td></tr><tr><td>zzz</td><td>The millisecond with leading zero</td></tr></table></td></tr>
</table>

Parameter	Description	
	Time Placeholder	
	z	The millisecond
	AP	Use AM/PM display. AP will be replaced by either "AM" or "PM".
	ap	Use am/pm display. ap will be replaced by either "am" or "pm".
Language	Select the language to use.	

Note: execution of the DumpTrend action will automatically force a flush to disk of the data temporarily maintained in the RAM memory. See "History trend widget" on page 267 for details on how to save sampled data to disk.

Note: external drives connected to USB port must have format FAT or FAT32. NTFS format is not supported.

WARNING: Be aware there are limits in the max number of files that can create inside a folder. Limits are depending of different factors and are not simple to calculate, you can think as 999 the max number of files that can be use inside a folder.

To convert binary dump files to .csv

The TrendBufferReader.exe tool is stored in the *Utils* folder of the LRH SW installation folder.

Use the following syntax:

```
TrendBufferReader -r Trend1 Trend1.csv 1
```

where:

`Trend1` = name of the trend buffer without extension resulting from the dump (original file name is trend1.dat)

`Trend1.csv` = name for the output file.

WARNING: The TrendBufferReader.exe is an old utility that not work with the new multi tags buffers. Using of this utility is not recommendable. The utility is not more maintenanced because now there is the possibility to dump trend buffer directly in .csv format.

.csv file structure

The resulting .csv file has five columns

Column	Description
Data Type	Data type of sampled tag: 0 = empty 1 = boolean 2 = byte 3 = short 4 = int 5 = unsignedByte 6 = unsignedShort 7 = unsignedInt 8 = float 9 = double
Value	Value of the sample
Timestamp (UTC)	Timestamp in UTC format
Sampling Time(ms)	Sampling interval time in milliseconds
Quality	Tag value quality. Information coded according the OPC DA standard and stored in a byte data (8 bits) defined in the form of three bit fields; Quality, Sub status and Limit status. The eight quality bits are arranged as follows: QQSSSSLL. For a complete and detailed description of all the single fields, please refer to the OPC DA official documentation.

Commonly quality values

The most commonly used quality values returned by the HMI acquisition engine are:

Quality Code	Quality	Description
0	BAD	The value is bad but no specific reason is given
4	BAD	Specific server problem with the configuration. For example, the tag has been deleted from the configuration file (tags.xml).
8	BAD	No value may be available at this time, for example the value has not been provided by the data source.
12	BAD	Device failure detected
16	BAD	Timeout before device response.
24	BAD	Communication failure

Quality Code	Quality	Description
28	BAD	No data found for upper or lower bound value Trend interface specific flag.
32	BAD	No data collected (for example, archiving not active. Trend interface specific flag. This value is also used to indicate a temporary offline status (for any condition where sampling was stopped).
64	UNCERTAIN	No specific reason.
65	UNCERTAIN	No specific reason. The value has 'pegged' at some lower limit.
66	UNCERTAIN	No specific reason. The value has 'pegged' at some higher limit.
67	UNCERTAIN	No specific reason. The value is a constant and cannot move.
84	UNCERTAIN	Returned value outside its defined limits defined. In this case the **Limits** field indicates which limit has been exceeded but the value can move farther out of this range.
85	UNCERTAIN	Returned value outside its defined limits defined. In this case the **Limits** field indicates which limit has been exceeded but the value can move farther out of this range. The value has 'pegged' at some lower limit.
86	UNCERTAIN	Returned value outside its defined limits defined. In this case the **Limits** field indicates which limit has been exceeded but the value can move farther out of this range. The value has 'pegged' at some higher limit
87	UNCERTAIN	Returned value outside its defined limits defined. In this case the **Limits** field indicates which limit has been exceeded but the value can move farther out of this range. The value is a constant and cannot move.
192	GOOD	-

DeleteTrend

Deletes saved trend data.

Define the name of the trend from which you want to delete logs.

DumpEventArchive

Stores historical alarm log and audit trail data to external drives, such as USB memory or SD card.

Parameter	Description
EventArchive	Name of buffer to dump data
FolderPath	Destination folder • Internal = *\Flash\QTHMI\workspace\Dump* • USB drive = *\USBMemory* • SD Card = *\Storage Card* • Public Network = *\\<hostname or IP>\sharePath* • Private Network = *\\<username>:<password>@<hostname or IP>\sharePath* Note: supported formats for external memory are FAT or FAT32 (NTFS format is not supported). Note: Private networks are supported only from Linux devices with BSP 1.0.25 and above.
DumpConfigFile	Dump the description files of the archives
DumpAsCSV	**true** = the buffer is dumped to the specified location as a .csv file **false** = the buffer is dumped in binary format (a .dat file and .inf file). Both these files are then required to convert data in .csv format by an external utility.
DateTimePrefix	**true** = the dumped file will have date and time as prefix to its name (for example D2012_01_01_T10_10_alarmBuffer1.csv)
timeSpec	Time format: • **Local** = the time values exported are the time of the HMI device. • **Global** = the time values exported are in UTC format.
csv Colums	Select the columns to dump into the .csv file. Available only when the EventArchive is an alarms buffer
FileName	The below wildcards are supported • %n = Event archive name • %y = Year • %M = Month • %d = Day • %h = Hour • %m = Minutes • %s = Seconds

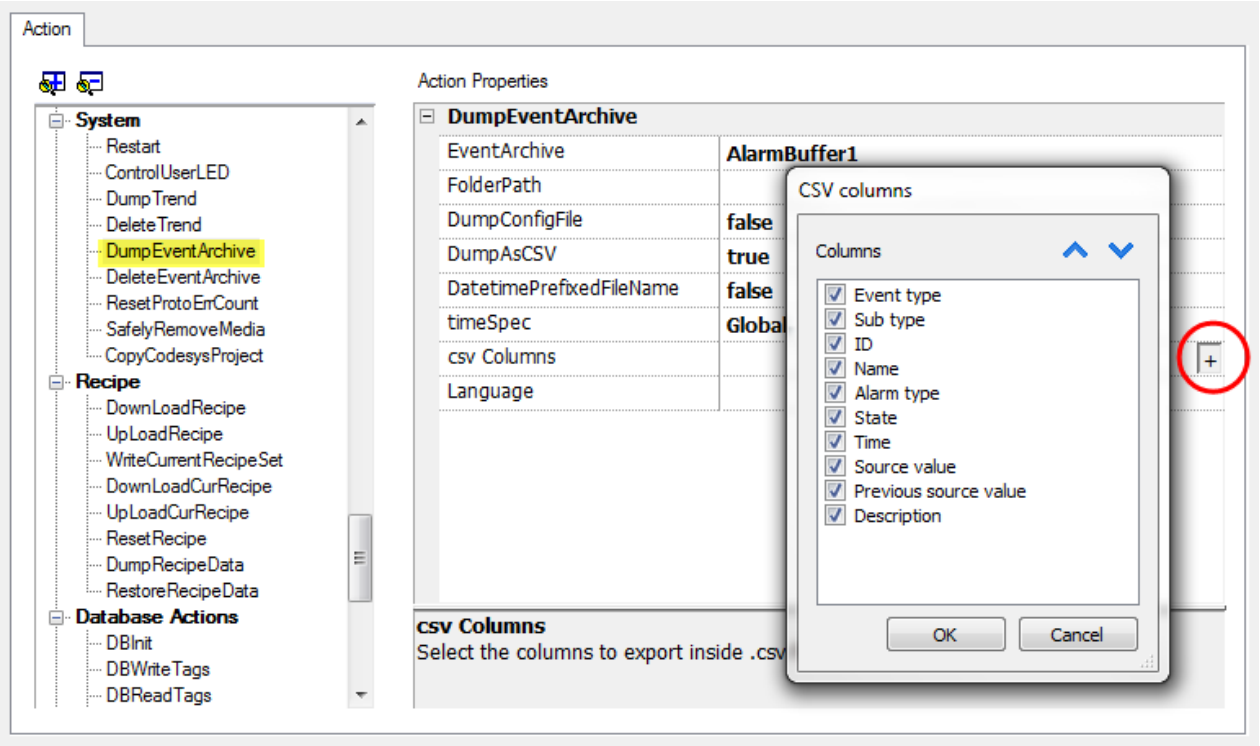
Parameter	Description
	Example: \%n\%y%M%d\%h%m%s Available only when the DateTimePrefixFileName=true
Language	Select the language to use. Available only when the EventArchive is an alarms buffer

Dumping in CSV Format

DumpAsCSV = true

For Alarms buffers, the additional "csv Colums" parameter give the possibility to select the columns to export inside the .csv file

Note: available only for Alarms buffers.



Dumping in BINARY Format

DumpAsCSV = false

When exporting Event buffers in binary format and **DumpConfigFile** is set to true (recommended settings), there are two folders:

- **data**, containing data files,
- **config**, containing configuration files for .csv conversion.

Once the two folders are copied from the USB drive to the computer disk, the folder structure will be:

config\

alarms.xml

eventconfig.xml

data\

AlarmBuffer1.dat

AlarmBuffer1.inf

\

AlarmBufferReader.exe

To convert dump files to .csv

The AlarmBufferReader.exe tool is stored in the *Utils* folder of the LRH SW installation folder.

Use the following syntax:

```
AlarmBufferReader AlarmBuffer1 FILE ./AlarmBuffer1.csv
```

where:

- `AlarmBuffer1` = name of the dumped .dat without extension
- `AlarmBuffer1.csv` = name for the output file.

The utility AuditTrailBufferReader.exe is available for Audit Trail buffers.

Use the following syntax:

```
AuditTrailBufferReader AuditTrail FILE ./AuditTrail.csv
```

where:

- `AuditTrail` = name of the dumped buffer without extension and
- `AuditTrail1.csv` = name for the output file.

WARNING: The AlarmBufferReader.exe is an old utility that not work with newer buffer formats. Using of this utility is not recommendable. The utility is not more maintenanced because now there is the possibility to dump alarm buffer directly in .csv format.

DeleteEventArchive

Deletes saved Event buffers log data.

Specify the name of Event buffer to delete from the Event logs.

ResetProtoErrCount

Resets the Protocol Error Count system variable.

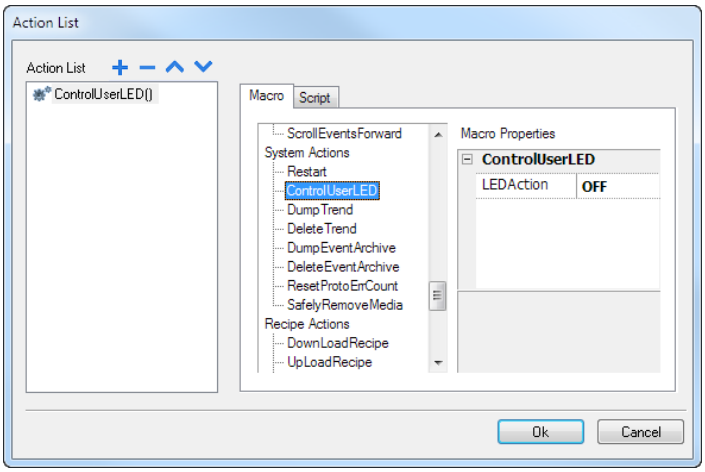
See "System Variables (Attach To)" on page 125 for details.

SafelyRemoveMedia

Provides for safe removal of SD card or USB drive fromHMI.

ControlUserLED

Sets the user LED behavior.



Not available on Linux platforms (find the platform of your device at "HMI devices capabilities" on page 541)

SaveEventArchive

Save the records located within the audit trail to a signed file. The file signature will ensure that the records within the report are not altered.

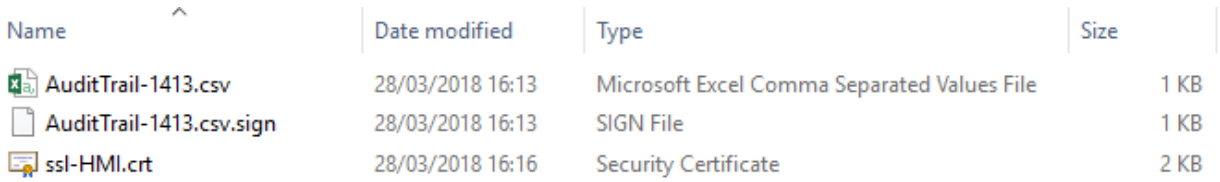
Parameter	Description
EventArchive	Name of buffer to dump data
FolderPath	Destination folder • Internal = *\Flash\QTHMI\workspace\Dump* • USB drive = *\USBMemory* • SD Card = *\Storage Card* • Public Network = *\\<hostname or IP>\sharePath* • Private Network = *\\<username>:<password>@<hostname or IP>\sharePath* Note: supported formats for external memory are FAT or FAT32 (NTFS format is not supported). Note: Private networks are supported only from Linux devices with BSP 1.0.25 and above.
FileName	The below wildcards are supported

Parameter	Description
	• %n = Event archive name • %y = Year • %M = Month • %d = Day • %h = Hour • %m = Minutes • %s = Seconds Example: \%n\%y%M%d\%h%m%s
Format	Format of the output file • CSV
Signed	Generate the file signature.  **On Linux devices, the BSP v1.0.239 or greater is required**  The algorithm to use to signing is defined inside the project properties parameters See "Project" on page 80 for the available algorithms See also: • "Signed CSV files" on page 318
TimeSpec	Time format: • **Local** = the time values exported are the time of the HMI device. • **Global** = the time values exported are in UTC format.
PeriodMode	Defines the time window to export • All events • Today • Yesterday • Last week • Last month • Current week • Current month • Custom The additional parameters "periodFrom" and "periodTo" will be shown
Separate Date and Time	Uses two separate columns for Date and Time
Date Format	Select the Date and Time format

Signed file

When the "Signed file" parameter is true, two files will be added in addition to fileame.csv:

- filename.csv.sign
  The file signature will ensure that the records within the file filename.csv file have not been altered
- ssl-HMI.crt
  A copy of the certificate of the HMI device required to verify the authenticity of the report.



For more information about the certificate and how to verify signed files, see "x.509 Certificate" on page 316.

For more information about the exported information see "Exporting audit trail as .csv files" on page 344.

LogMessage

Add a message into the audit trail buffer.

This macro give the possibility to developer to decide to keep track of some events (e.g. when a button is pressed, when a page is activate, etc.) into the audit trail. The attach to tag to have the possibility to define the message to log at runtime is supported.

Parameter	Description
EventArchive	Name of the audit buffer where add the message
Message	Message to add inside the audit buffer

DeleteOldFiles

This macros delete files older that a give number of days.

In PC there is no restriction in using path. In panels it is allowed in dynamic media and data partition /mnt/data)

It will be developer responsibility to configure the application to avoid the possibility to delete system files.

Parameter	Description
FolderPath	Folder where search the files to delete
FileTypes	List of files to delete separate by comma. Wildcard are supported Example: *.png,*.jpg
OlderDays	Minimum number of days without changes

Tag actions

Interacts with tags.

DataTransfer

Exchanges data between:

- two controllers,
- registers within a controller,
- from system variables to controllers,
- from controllers to system variables

The various tag types include a controller tag, a system variable, a recipe tag and widget property.

ToggleBit

Toggles a bit value of a tag.

BitIndex allows you to select the bit to be toggled: toggling requires a read-modify-write operation; the read value is inverted and then written back to the tag.

SetBit

Sets the selected bit to “1".

BitIndex allows you to select the bit position inside the tag.

ResetBit

Resets the selected bit to “0”

BitIndex allows you to select the bit position inside the tag.

WriteTag

Writes constant values to the controller memory. Specify tag name and value.

StepTag

Increments or decrements tag value.

Parameter	Description
TagName	Name of tag to increase/decrease
Step	Step value
Do not step over limit	Enables step limit
Step Limit	Value of step limit, if enabled.

BiStep

This action is similar to the StepTag action but the direction Increment/Decrement is automatically chosen by the rotation of the Wheel. Tag value will be increased when the Wheel is rotated clockwise. Tag value will be decreased in when the Wheel is rotated counterclockwise.

Property	Description
TagName	Name of Tag on which execute BiStep Tag action
Step	Value to be added/subtracted to Tag at every wheel rotation (depends on Event step property)
Event step	This property allows to chose if adding/subtracting step values at every single wheel step, or at every rotation event. **false** = The step value is added/subtracted to the Tag at every rotation event. *Example: rotate the wheel performing 5 wheel steps in a single event, Tag will be increased/decreased by 1.* **true** = The step value is added/subtracted to the Tag at every single wheel step. *Example: rotate the wheel performing 5 wheel steps in a single rotation, Tag will be increased/decreased by 5.*
Do not step over limit	If true, enables lower and upper limits, which represents the lower and the higher value that the Tag can assume due a BiStep Tag action
LowerLimit	If "Do not step over limit" is true, this property represents the lower value that the Tag can assume due a BiStep Tag action
UpperLimit	If "Do not step over limit" is true, this property represents the higher value that the Tag can assume due a BiStep Tag action

Available only inside OnWell Actions

ActivateGroup

Forces the update of a group of tags.

Tags are updated either when used in the current page or continuously, if defined as active in the Tag Editor. This action forces all the tags of a group to be continuously updated.

DeactivateGroup

Deactivates a group of tags, that is stops forcing the update of a group of tags.

EnableNode

Enable/disables action for offline node management. No communication is done with a disabled node.

Parameter	Description
Protocol ID	Unique identifier of selected protocol
NodeID	Node identifier in selected protocol. Can be attached to a tag.
Enable	Node communication status: **False** = disabled **True** = enabled

Parameter	Description
	When attached to a tag, tag = 0 means **False**

ClearRetentiveMemory

When set to 0, clears the content of the Retentive Memory.

ForceReadTag

Force a refresh of the specified tag from the remote controller.

Trend actions

Used for Live Data Trends and Historical Trends Widget.

RefreshTrend

Refreshes the **Trend** window.

It can be used in any Trends/Graphs widgets. Specify the widget as a parameter for the action.

ScrollLeftTrend

Scrolls the **Trend** window to the left side, by one-tenth (1/10) of the page duration.

Note: with the real-time trends pause the trend using the **PauseTrend** action, or the window will be continuously shifted to the current value.

ScrollRightTrend

Scrolls the **Trend** window to the right side, by one-tenth (1/10) of the page duration.

Note: with the real-time trends pause the trend using the **PauseTrend** action, or the window will be continuously shifted to the current value.

PageLeftTrend

Scrolls the **Trend** window by one-page. For example, if the page size is 10 minutes, then use the **PageLeftTrend** action to scroll the trend left for 10 minutes.

PageRightTrend

Scrolls the **Trend** window by one-page. For example, if the page size is 10 minutes, then use the **PageRightTrend** action to scroll the trend right for 10 minutes.

PageDurationTrend

Sets the page duration of the **Trend** window.

Define trend name and page duration.

Note: you can set page duration at runtime using a combo box widget.

ZoomInTrend

Reduces page duration.

ZoomOutTrend

Extends page duration.

ZoomResetTrend

Reset the zoom level back to the original zoom level.

ZoomInYAxisTrend

Reduces Y Axis.

ZoomOutYAxisTrend

Extends Y Axis.

ZoomResetYAxisTrend

Reset the Y Axis zoom level back to the original zoom level.

PauseTrend

Stops plotting the trend curves in the **Trend** window.

When used with real time trend the plotting stops when the curve reaches the right border of the graph. This action does not stop trend logging.

ResumeTrend

Resumes trend plotting if paused.

ShowTrendCursor

Shows value of the curve at a given point on the X axis.

It activates the trend cursor. A cursor (vertical line) will be displayed in the trend widget.

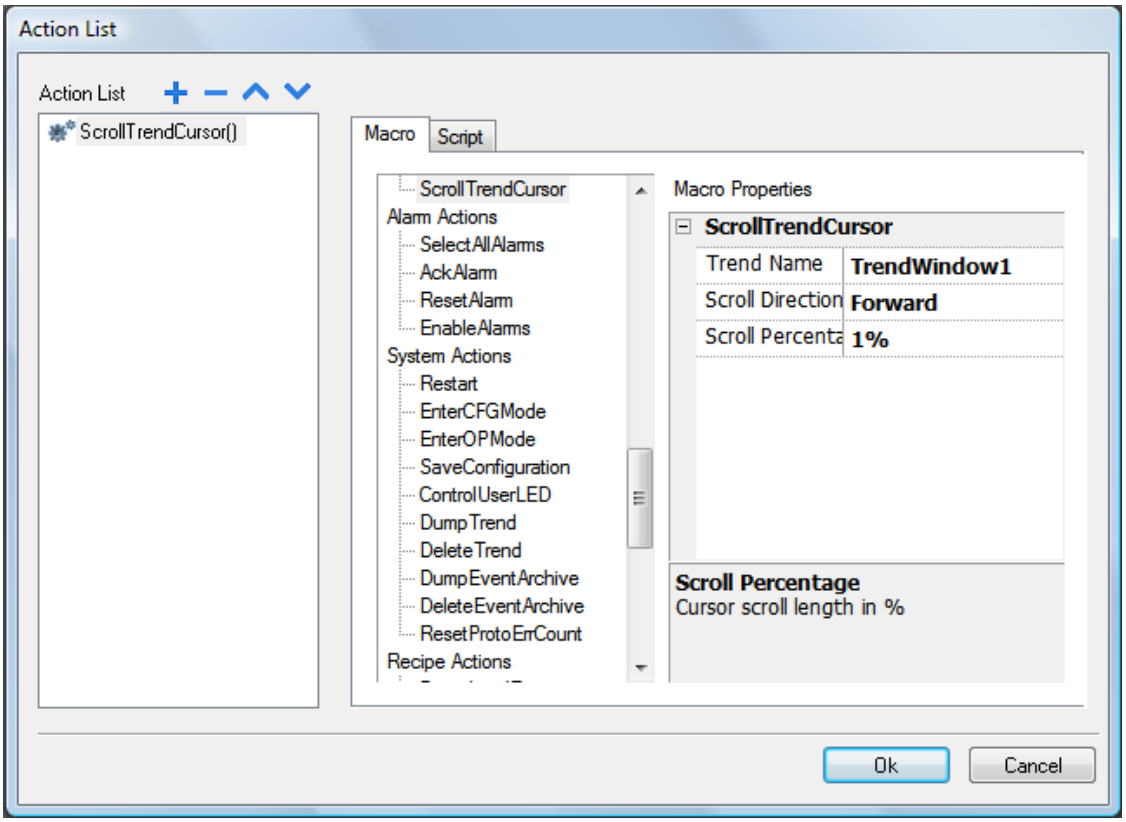
When the graphic cursor is enabled, the scrolling of the trend is stopped.

The **ScrollCursor** action moves the graphic cursor over the curves, or over the entire **Trend** window.

ScrollTrendCursor

Scrolls the trend cursor backward or forward.

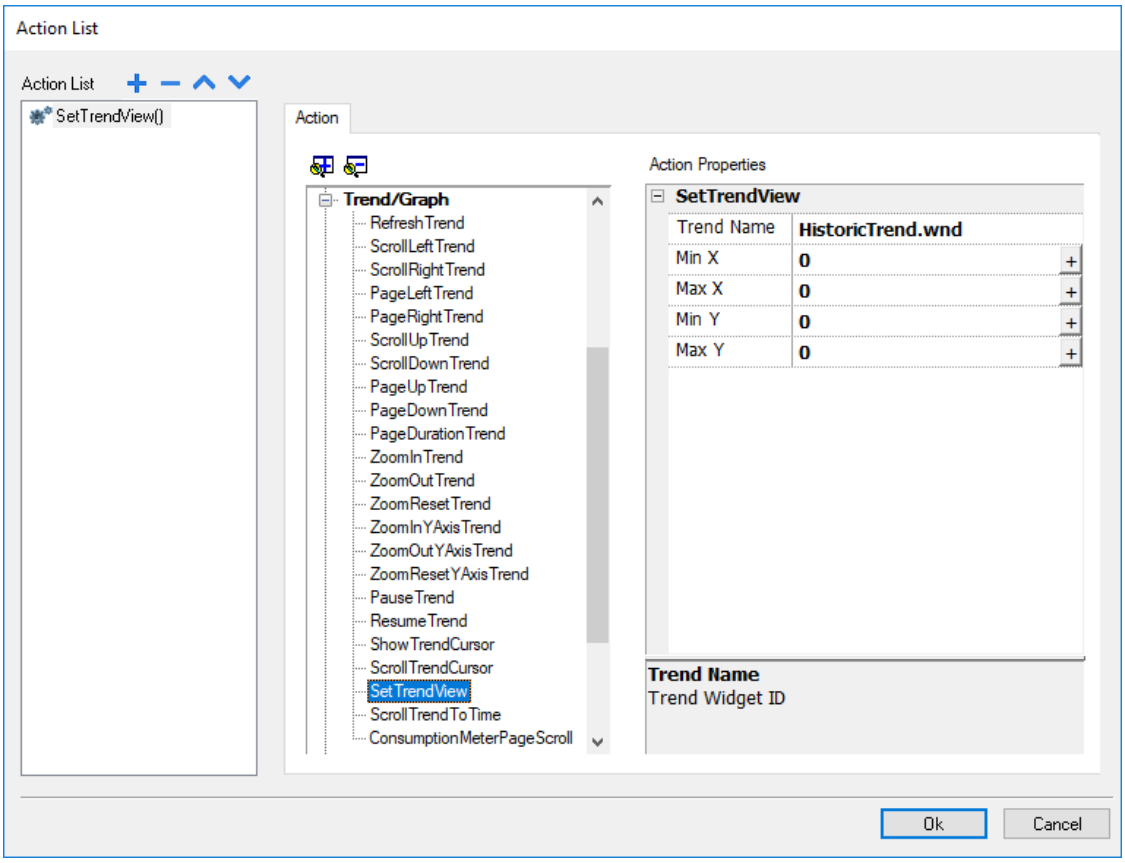
The Y cursor value will display the trend value at the point of the cursor. Scrolling percentage can be set at 1% or 10%. The percentage is calculated on the trend window duration.



SetTrendView

Use this macro to change the axis ranges of the trend view.

When both Min X=0 and Max X=0, the static values defined inside the properties of widget are used. The same for the Y axe.



ScrollTrendToTime

Scrolls the **Trend** window to a specified point in time.

Use this action when you need to scroll to a specific position in a trend window when a specific event occurred.

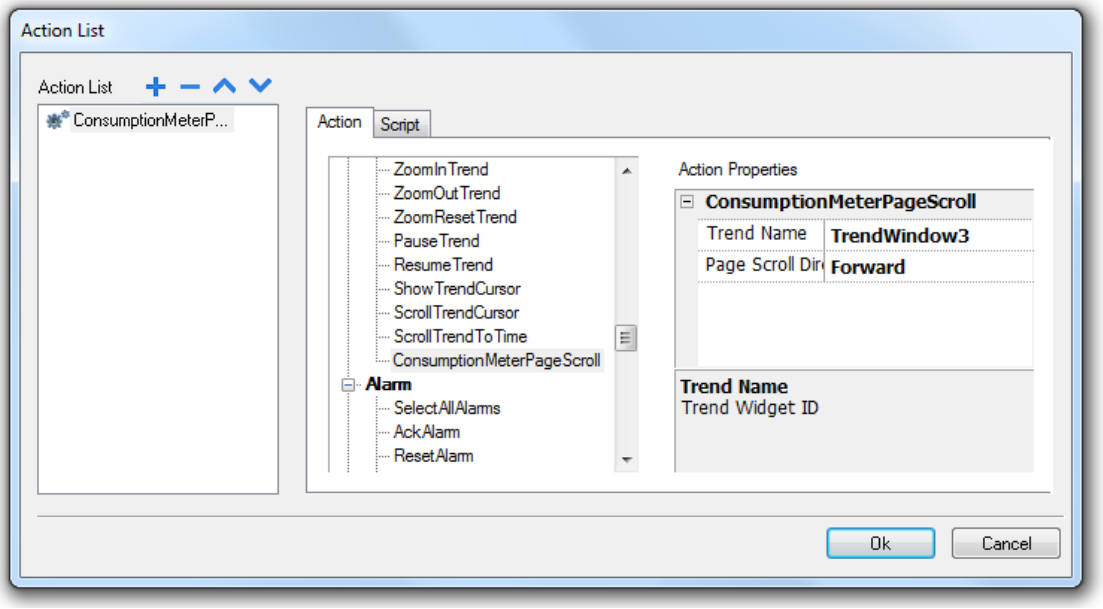
Example

1. Configure an action for an event (for example, an alarm) that executes a data transfer of the system time into a tag.
2. Select that tag as **ScrollTrendtoTime** parameter: the trend windows will be centered at the time when the event was triggered.

ConsumptionMeterPageScroll

Scrolls the page backward or forward in a Consumption Meter widget.

Parameter	Description
Trend Name	Trend widget ID (for example, TrendWindow3)
Page Scroll Direction	Direction of page scrolling (Forward/backward)



Text Editor actions

Macros used to interacts with the TextEditor widget.

Reference to "TextEditor widget" on page 447 for details

User management actions

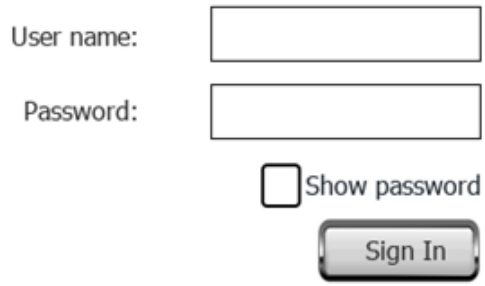
User management and security settings.

LogOut

Logs off the current user. The default user is then automatically logged in. If no default user has been configured, the logon window is displayed.

SwitchUser

Switches between two users without logging off the logged user: the user login dialog appears. User can click **Back** to go back to the previously logged user.



The server continues running with the previously logged user, until the next user logs on. One user is always logged onto the system.

ChangePassword

Change current user password: a dialog appears

No parameter is required.

ResetPassword

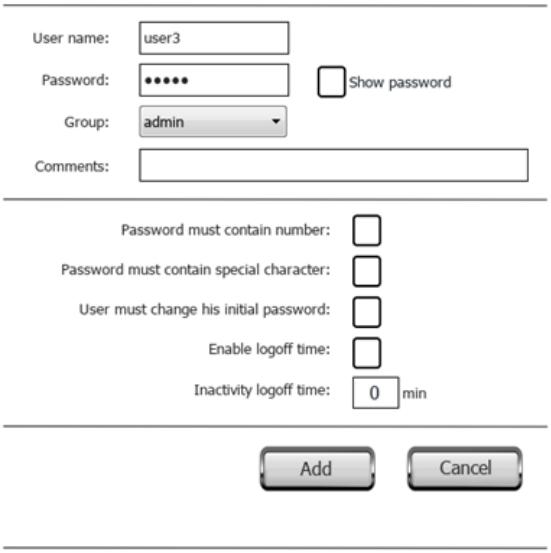
Restores the original password together with the settings specified in the project for the current user.

No parameter is required.

AddUser

*Reserved to users with **Can manage other users** property set.*

Adds a user at runtime: a dialog appears.



DeleteUser

*Reserved to users with **Can manage other users** property set.*

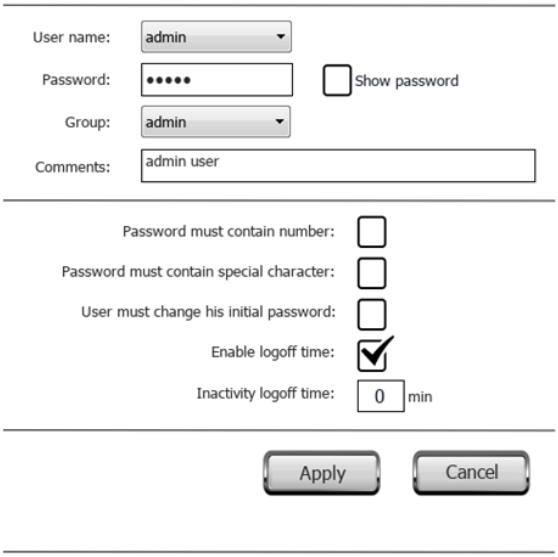
Deletes a user at runtime: a dialog appears.

No parameter is required.

EditUsers

*Reserved to users with **Can manage other users** property set.*

Edits user settings.



DeleteUMDynamicFile

Deletes the dynamic user management file. Changes made to users settings at runtime are erased. The original settings are restored from the project information.

No parameter is required.

ExportUsers

Exports user settings to an .xml file (*usermgnt_user.xml*) in encrypted format to be restored when needed.

Set destination folder for the export file.

Important: The user file is encrypted and cannot be edited.

Note: supported formats are FAT or FAT32. NTFS format is not supported.

ImportUsers

Imports user settings from a previously saved export .xml file (*usermgnt_user.xml*).

Set source folder for the import file.

Note: supported formats are FAT or FAT32. NTFS format is not supported.

Widget actions

ShowWidget

Shows or hides page widgets.

Property	Description
Widget	Widget to show/hide

SlideWidget

Shows the sliding effect of a widget, or of a widget group.

Note: The widget or grouped widgets can actually be outside of visible part of the page in the project and slide in and out of view.

Property	Description
Widget	Widget to slide
Direction	Sliding direction
Speed	Transition speed of sliding widget
X Distance	Travel distance of X coordinate in pixels
Y Distance	Travel distance of Y coordinate in pixels
Slide Limit	Enable/Disable movement limits of the widget with respect to the x, y coordinates
X Limit	Limit position of slide action for x coordinate
Y Limit	Limit position of slide action for y coordinate
Toggle Visibility	Show/hide widget at the end of each slide action
Image Widget	Image displayed during slide action

BeginDataEntry

Displays a keypad and starts data entry on a data field without touching the widget itself. This action can be used to activate data entry using a barcode scanner.

Java Script Interface

```
project.beginDataEntry(wgtName [, pageName])
```

Parameter	Description
wgtNameWidget	Widget name
pageName	Active page for data entry. Optional parameter. Useful to select a data field inside a non-modal active dialog box.

TriggerIPCamera

Captures an image from an IP Camera. Only works on pages that include an IP Camera widget.

MoveIPCamera

Sends remote commands to a camera that supports them. See "IP Camera widgets" on page 428 for details. Make sure that the IP Camera supports movement commands.

RefreshEvent

Refreshes the event buffer for **Alarm History** widget. See "Alarms History widget" on page 241 for details.

ContextMenu

Displays the context menu.

If **Context Menu** property of Project Widget has been set to **On delay** context menu can appear also touching for a few seconds the background area of the screen. See "Project properties" on page 73

ReplaceMedia

Replaces existing media files with new files from USB/SD card. Can be used to replace video files of MediaPlayer widgets, or images of project.

Note: New media files must have same name and format of the files to be replaced.

Parameter	Description
Media Type	Type of file to update
Device	Device where new media files are supplied
sourcePath	Folder where new media files are stored (for example, "\USBMemory")
Image Resize	Resizes new images to the size of images to be replaced. Not applicable to video files.
Silent	Replaces media automatically. As defau a dialog is displayed for the user to specify file location.

Java Script Interface

```
void replaceMedia(var sourcePath, var bSilent, var Device, var nMediaType, var
bResize)
```

```
project.replaceMedia("Images", true, "\USBMemory", 1, true);
```

ScrollTable

Scroll rows of the table forward or backward.

Parameter	Description
Table Widget	Table widget name
Direction	The number of rows to jump, forward when positive, backward when negative.

Java Script Interface

```
page.getWidget(TableWgt).scrollTo(Direction);
```

ShiftTableDataSrcColumns

Shift left or right the columns of a data table. Note the remapping is applicated to the data source widget.

Parameter	Description
Data source widget	Data source widget id
Columns Shift	Data source widget columns are shifted (left or right, depeding on sign) by this amount
Fixed left columns	A custom amount of columns (on the left of table) can be kept fixed during shifting
Remap Filter	Table widget filter (if defined) is connected to a data source widget column. This column, by default, is not remapped by shift action, but can be forced to true

Java Script Interface

```
var ColumnOrder = [0,1,2,3,4,5,6,7,8,9,10];
var json = {_c:ColumnOrder};

page.getWidget("TableDataSrcWgt").remapColumns(json);
```

ResetTableDataSrcColumns

Restore the original columns order (see "ShiftTableDataSrcColumns" macro)

SetTableSortingColumn

Select a column and the criteria to use to sort the rows of the table.

Parameter	Description
Table Widget	Table to sort
Data Source Column	Column to use to sort the table
Sorting Mode	Can be: Ascending, Descendent or Toggle
Sorting Type	Can be: Alphabetic or Numeric

Java Script Interface

```
var column = "Column1";    // Colum name (TableDataSource)
var mode = 0;              //0=Ascending, 1=Descending
var type = 1;              //0=Aphabetic, 1=Numeric

var sorting_rule_1 = { _c : column, _m : mode, _t : type };
var json = [ sorting_rule_1 ]
page.getWidget("TableWgt").setSortingRules(json);
```

15 The LRH SW Client

LRH SW Client is a standalone application which provides remote access to the LRH SW HMI Runtime, and is included in the LRH SW. The LRH SW Client uses the same graphic rendering system as the runtime in the HMI devices, it relies on a specified LRH SW HMI Runtime as server for live data.

LRH SW Client acts as a remote client and communicates to the server, sharing the local visualization with the tag values that are maintained or updated by the communication protocol.

HMI projects contain properties indicating which page is currently displayed on the HMI and can force the HMI to switch to a specific page. You can use these properties to synchronize pages showed on the HMI device and LRH SW Client or to control an HMI device with a PLC. See "Project" on page 80 for details.

To avoid unexpected behavior:

- **be sure to use the same version of the LRH SW HMI Runtime**
- **use "FreeType Font Rendering" to be sure to use the same font rendering engine on both HMI Client and HMI Device (see "Runtime" on page 74)**

Client application on PC

To run the LRH SW Client application on PC:

1. From the **Start** menu > **LRH SW** >**LRH SW Client**: the client opens in a browser-like style window.
2. Type the server/device IP address in the address bar (for example: http://192.168.1.12): LRH SW Client will connect to the server and the same graphical application running on the device will be loaded in the client window.

The Client application toolbar

Element	Description
HMI server address	IP address of the remote HMI device (e.g. 192.168.0.1:80)
Connection status	Network request status. Red during data exchange.
Reload from cache	Reloads project
BookMark	Bookmarks preferred pages and reload them.
Settings	Opens **Settings** dialog

Reload options

Option	Description
F5	Reloads project from cache
Shift + F5	Downloads project to client

Transferring files to a remote HMI device

You can upload and download files to and from a remote HMI device using two dedicated actions. These actions can only be used from a remote LRH SW Client and access remote files via FTP.

See "Remote Client actions" on page 191 and "Remote Client variables" on page 133.

Important: Enable FTP support and give all necessary user rights to the folders used to transfer files.

Workspace

Project files are uploaded from the device and stored in LRH SW Client into the following cache folder.

%appdata%\Lovato\[build number]\client\cache

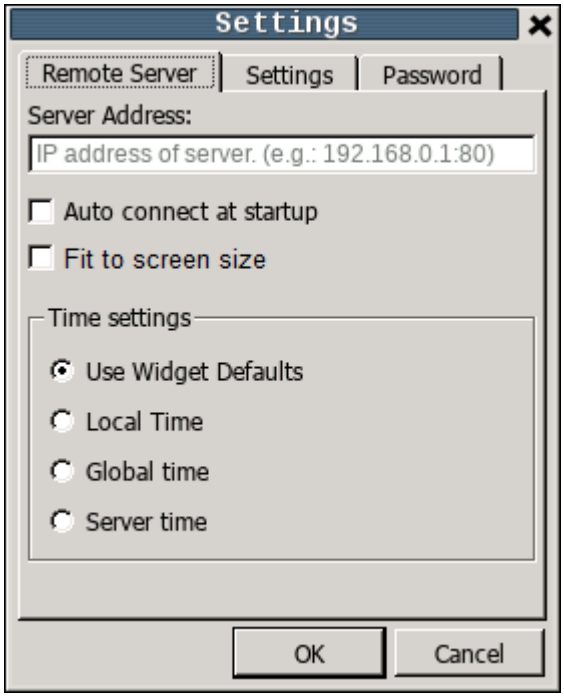
where:

[build number] = folder named as build number, for example 01.90.00.608.

Client application on HMI

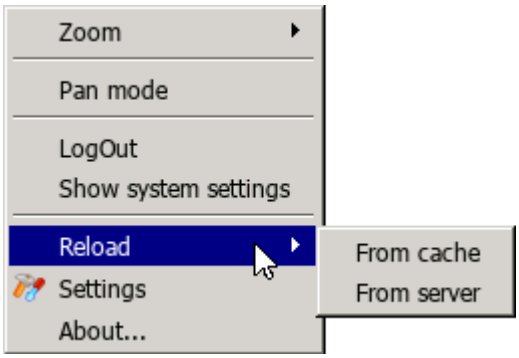
To run the LRH SW Client application on Linux HMI device:

1. From the **Run** > **Update Package** menu, create an Update Package and install the HMI Client application in to the HMI device (see "Update package" on page 95 for additional information)
2. Type the server/device IP address in the Setting dialog that will be available when HMI device start (for example: http://192.168.1.12): HMI Client will connect to the server and the same graphical application running on the device will be loaded in the client window.



Context Menu

The Context Menu, available with a right mouse click, will show the below commands:

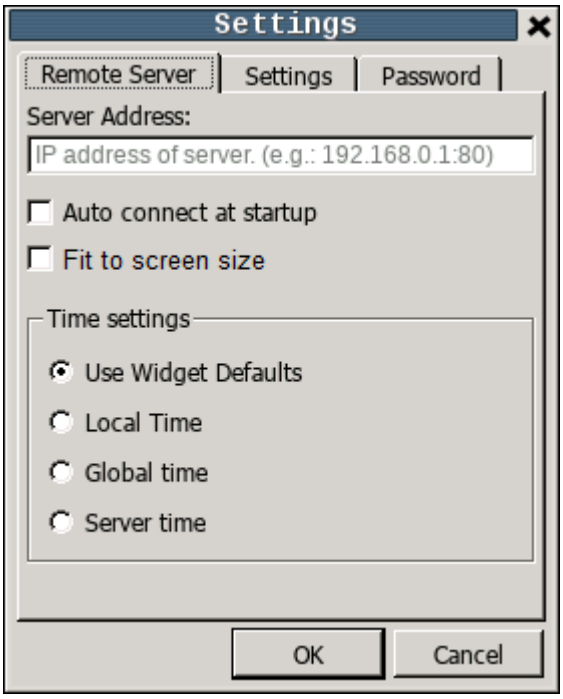


Option	Description
Zoom	Select view size at runtime • Zoom In • Zoom Out • Zoom 100%
Pane Mode	Enables/disables pan mode after a zoom in
Logout	Logs off the current user.
Show system settings	Allow the HMI settings and the management of system components. See "System Settings" on page 543 for details.
Reload	Reload remote project • From cache • From server
Setting	Open the HMI Client Settings. See "Settings and time zone options" below for details  Could be password protected
About	Shows information about the HMI Client version.

Settings and time zone options

In the **Settings** dialog you can configure client settings and decide how to display project time stamp information.

Remote Server



Connection settings

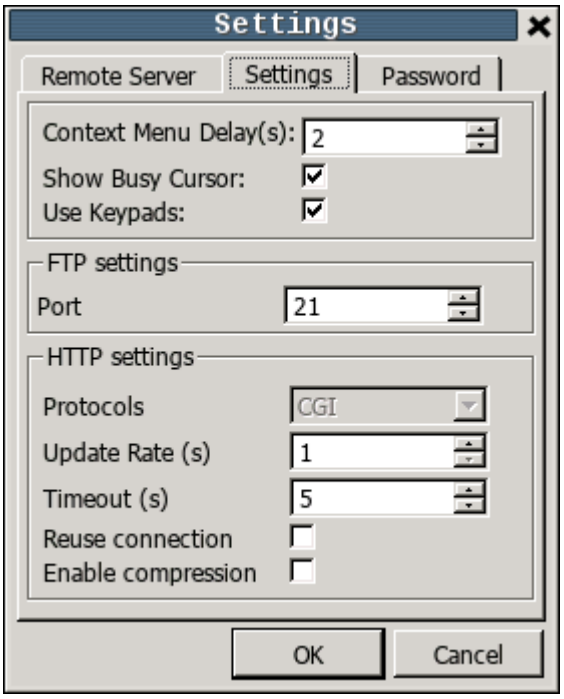
Parameter	Description
Server Address	IP address of the remote HMI device (e.g. 192.168.0.1:80)
Auto connect at startup	When the panel starts, use the Server Address to try to connect automatically to the remote server.
Fit to screen size	Adapts the view to the screen size

Time settings

Parameter	Description
Use Widget Defaults	Displays time information according to the widget settings.
Local Time	Translates all timestamps in the project into the computer local time where the client is installed.
Global Time	Translates all timestamps in the project into UTC format.
Server Time	Translates all timestamps in the project into the same used by HMI device/server in order to show the same time.

Important: Make sure you set the HMI RTC correct time zone and DST options.

Settings



Interface Settings

Parameter	Description
Context Menu Delay(s)	Context menu activation delay. Range: 1–60 seconds.
Show Busy Cursor	Display an hourglass when the system is busy
Use Keypads	Display keypads when user touches a data entry field. Set to **disable** when an external USB keyboard is connected to the device.

FTP settings

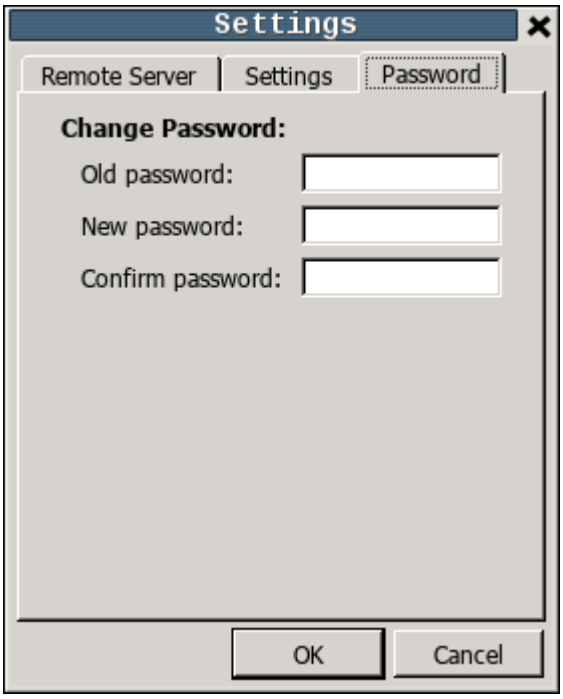
Parameter	Description
Port	FTP communication port

HTTP settings

Parameter	Description
Protocols	Communication protocol used by LRH SW Client to communicate with an HMI device.
Update Rate	Polling frequency to synchronize data from server. Default = 1 s.
Timeout	Maximum wait time before a request is repeated by the LRH SW Client. Default = 5 s.
Reuse connection	Enables reuse of the same TCP connection for multiple HTTP requests to reduce network traffic.

Parameter	Description
	Note: When enabled, this option may cause high latency if the proxy server does not immediately terminate old requests thus saturating connection sockets. This is often the case with 3G connections.
Enable compression	Compresses data to reduce download times. Default = disabled. ***CAUTION: enabling this option could causes excessive CPU overhead.***
Time Settings	Used by the client to adapt the widget time stamp information.

Password



This dialog give the possibility to change the internal password of the HMI device for the admin user (the default password is "admin").

Password protection is not available on PC version of the HMI client

16 Using the integrated FTP server

LRH SW HMI Runtime system uses an integrated FTP server.

Connect to the HMI device FTP server using any standard FTP client application. The FTP server responds on the standard port 21 as default.

Important: The server supports only one connection at a time; if you are using a multiple connection FTP client disable this feature on the client program or set the maximum number of connections per session to 1.

FTP settings

FTP default credentials

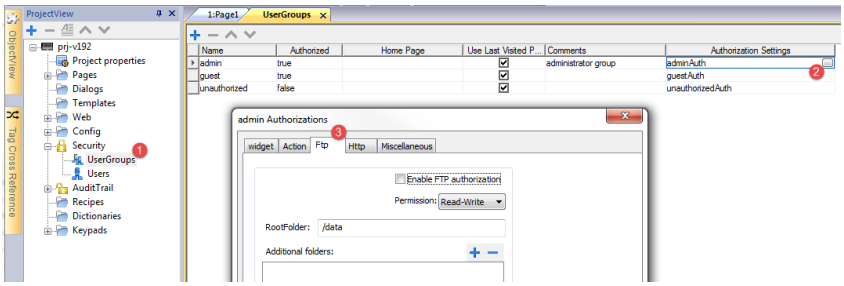
When User Management/Security is disabled use the following credentials for incoming connections:

User name	admin
Password	admin

Changing FTP settings

*Path: **ProjectView*> *Security*> *UserGroups* > *Authorization Settings***

You can change FTP permissions and account information in the **Ftp** tab of the **admin authorizations** dialog.



See "Configuring groups and authorizations" on page 326 for details.

17 Using VNC for remote access

VNC is a remote control software which allows you to see and control the HMI application remotely using your local mouse and keyboard.

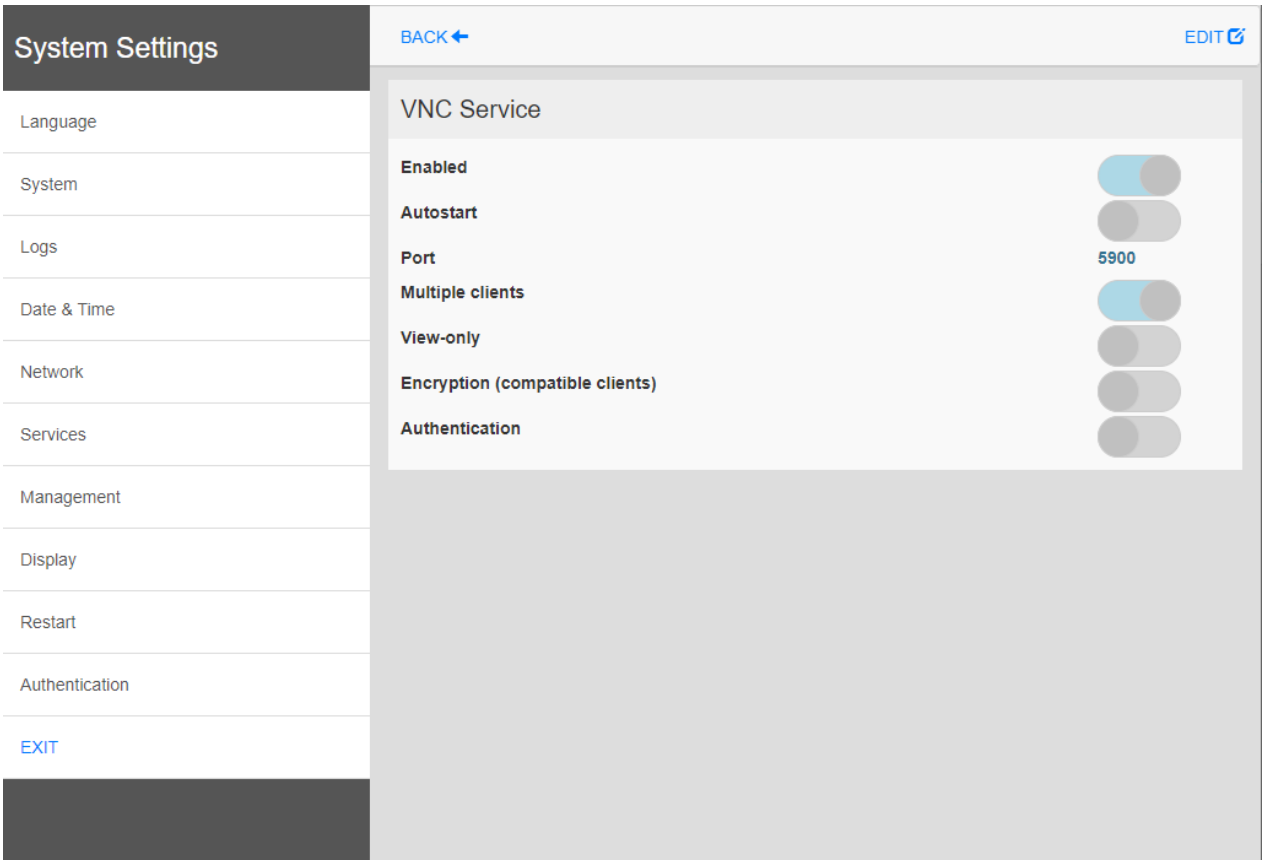
Remote access is particularly useful for administration and technical support. In order to use it you need to:

- start a server in the HMI device
- install a viewer on the remote device

Starting VNC server on Linux devices

VNC server is a service embedded inside the BSP that can be activated from the Services tab of the device System Settings. See "System Settings" on page 547 for details.

VNC should be disabled after use and autostart is not recommended.



Starting VNC viewer

No VNC viewer is provided as part of LRH SW.

Many compatible VNC viewers are available for free download (for example, TightVNC).

18 Alarms

The alarms handling system has been designed to provide alerts through pop-up messages, typically to display warning messages indicating any abnormal condition or malfunction in the system under control.

Whenever a bit changes, or the value of a tag exceeds a threshold set in the alarm configuration, a message is displayed. Specific actions can also be programmed to be executed when an alarm is triggered.

Important: No default action is associated with any alarm.

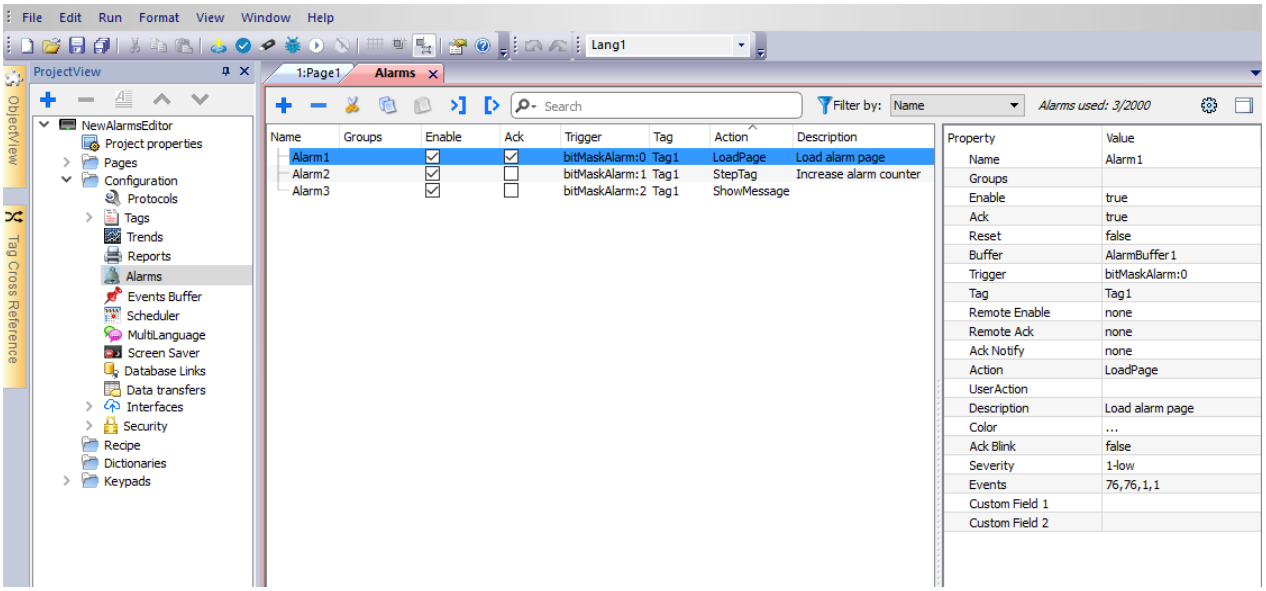
You can define how an alarm is displayed on the HMI device, if it requires user acknowledgment, and if and how it is logged into the event list.

Alarms are configured in the Alarms Configuration Editor and, thus, are available for all the pages of the project. An alarm widget can display more than one alarm at a time, if sized appropriately. You can trigger the opening or closing of the Alarm window with an event.

You work with alarms in the same way as you work with any other event. You may not want to display a dialog when an alarm is triggered and you can associate to it any other available action.

Alarms Editor

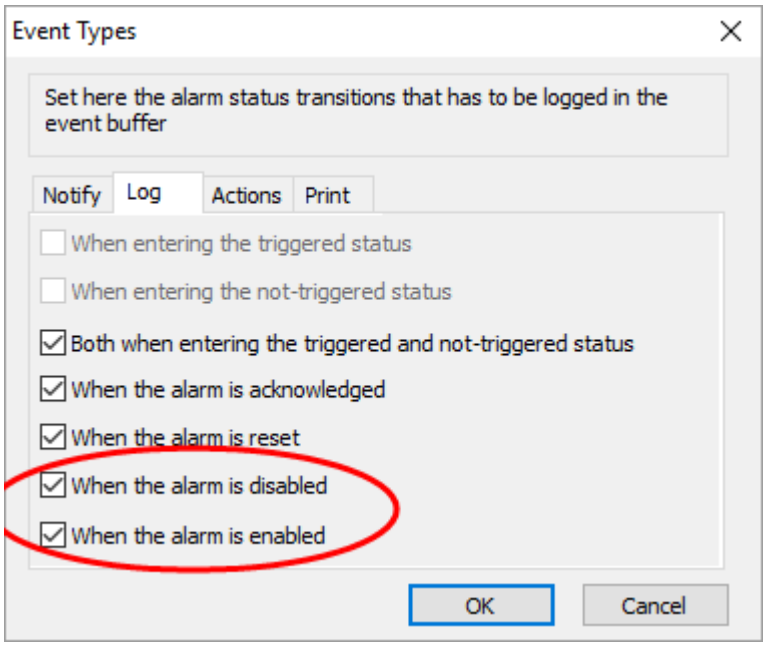
Path: ***ProjectView*** *>* ***Config*** *> double-click* ***Alarms***



Adding an alarm

Click **+** to add an alarm.

Parameter	Description
Name	Name of alarm
Groups	Groups associated with the alarm. They can be used in widgets display filters.
Enable	Enable/disable triggering of alarm.  Alarms can be enabled or disabled at runtime as well (see "Enable/disable alarms at runtime" on page 243 for details).
Ack	Enable/disable acknowledgment of alarm, if selected the operator must acknowledge the alarm once triggered to remove it from the **Active Alarm** widget.
Reset	Used with the **Ack** option, if selected, acknowledged alarms stay in the alarm list, labeled as **Not Triggered Acked**, until the operator presses the **Reset** button in the alarm widget.
Buffer	Buffer file where the alarm history will be saved.
Trigger	Triggering condition depending on alarm type: • **limitAlarm**: alarm triggered when tag value exceeds its limits. The alarm is not triggered if the value reaches the limits. • **valueAlarm** alarm is triggered when tag value is equal to the configured value

Parameter	Description
	• **bitMaskAlarm**: the bitwise AND operator compares each bit of the bitmask with the tag value corresponding to that Alarm. If both bits are on, the alarm is set to true. You can specify one or more bit positions (starting from 0) inside the tag. The Bit position must be given in decimal format; if more bits are specified, each position must be separated by a ",". • **deviationAlarm**: alarm triggered if the percentage of deviation of the tag value from the set point exceeds a set deviation. $\lvert Value_{now} - SetPoint \rvert > \left(\frac{deviation}{100} \times SetPoint\right)$
Tag	Tag whose value will trigger the alarm when it exceeds the set limits. The alarm can refer to the value of this tag, or to the state of a bit if **bitMaskAlarm** has been selected as trigger.
Remote Enable	Tag used by the PLC to enable/disable the alarm. • Changing the enable status from the Alarms Widget will change the tag value • When the tag cannot be read (e.g. communication error) the alarm is disabled • No tags related to the alarm are refreshed when alarm is disabled. Tip: It could be useful to enable the logging of the alarm's enable flag 
Remote Ack	Tag used by the PLC to acknowledge the alarm. A transition of this tag from 0 to a non zero value is considered an acknowledgment request. Leave empty if remote acknowledgment is not required. See "Remote alarms acknowledge" on page 233 for details.
Ack Notify	Tag used by the HMI device to notify when the alarm is acknowledged from the device or from the PLC.

Parameter	Description
	0 = set to this value when alarm is triggered 1 = set to this value when alarm is acknowledged.
Action	Actions executed when the alarm is triggered. Additional conditions can be specified in the **Events** column. See "Setting events" on page 234 for details. The macros added in the action field are executed on the server-side with the exception of the below macros that will be executed even on client-side (e.g. LRH SW). • loadPage • prevPage • nextPage • showDialog • showMessage • setLanguage • jsAction
User Action	Actions executed when user press the action button in the active alarm widget. See """Active Alarms widget" on page 236 for details.
Description	Alarm description. This text supports the multiple language features and can be a combination of static and dynamic parts, where the dynamic portion includes one or more tag values. See "Displaying live alarm data" on page 243 for details.
Custom Field #	It is an additional alarm description that can be used to show additional information inside the alarms widgets. For example, could be an index to use to show a picture related with the alarm.
Color	Foreground and background colors of alarm rows based on the status of alarm.
AckBlink	Blinking for triggered alarms. If selected the alarm rows blinks until acknowledged. Only effective if **Ack** is selected.
Severity	Severity of the alarm. If multiple alarms are triggered simultaneously, actions will be executed based on severity settings. 0 = not important 1 = low 2 = below normal 3 = normal 4 = above normal 5 = high 6 = critical
Events	Conditions in which the alarms are notified, logged or printed. See "Setting events" on page 234 for details.

Backup alarms events

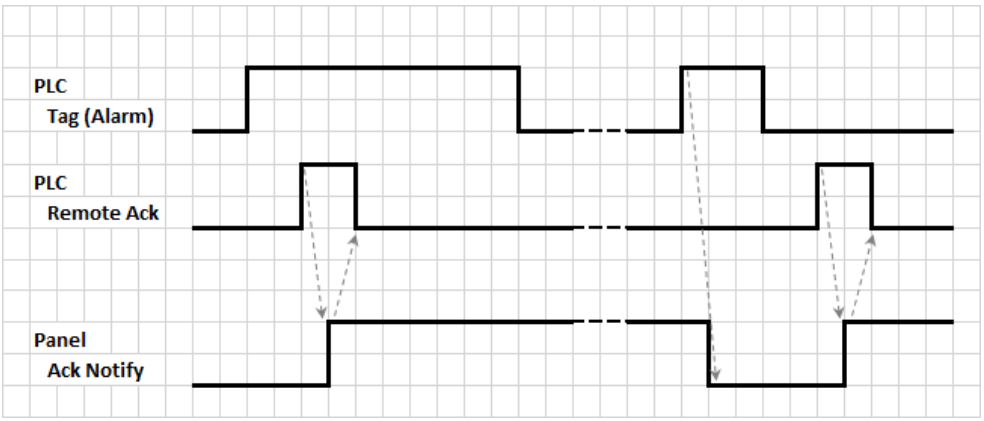
From the "Events Buffer" on page 249 you can configure the size of the alarms buffer and activate the backup of the alarms events when the buffer is full.

Remote alarms acknowledge

When the **Remote Ack** parameter is set, an alarm can be acknowledged from a PLC device setting a tag value to a nonzero value. The acknowledged status is notified to the PLC device by the **Ack Notify** flag.

Alarms acknowledgement process

Remote Ack tag is set/reset by the PLC to request the acknowledge, and **Ack Notify** is set/reset by HMI device to notify the execution of the acknowledge.



1. When an alarm condition is detected the HMI device set **Ack Notify** to 0 and all related actions are executed.
2. When the alarm is acknowledged (by HMI device or remotely), **Ack Notify** is set to 1
3. It's up to the controller to set **Remote Ack** to 1 to acknowledge the alarm or reset it to 0 when the HMI device send a notification that the alarm has been acknowledged (**Ack Notify** = 1)

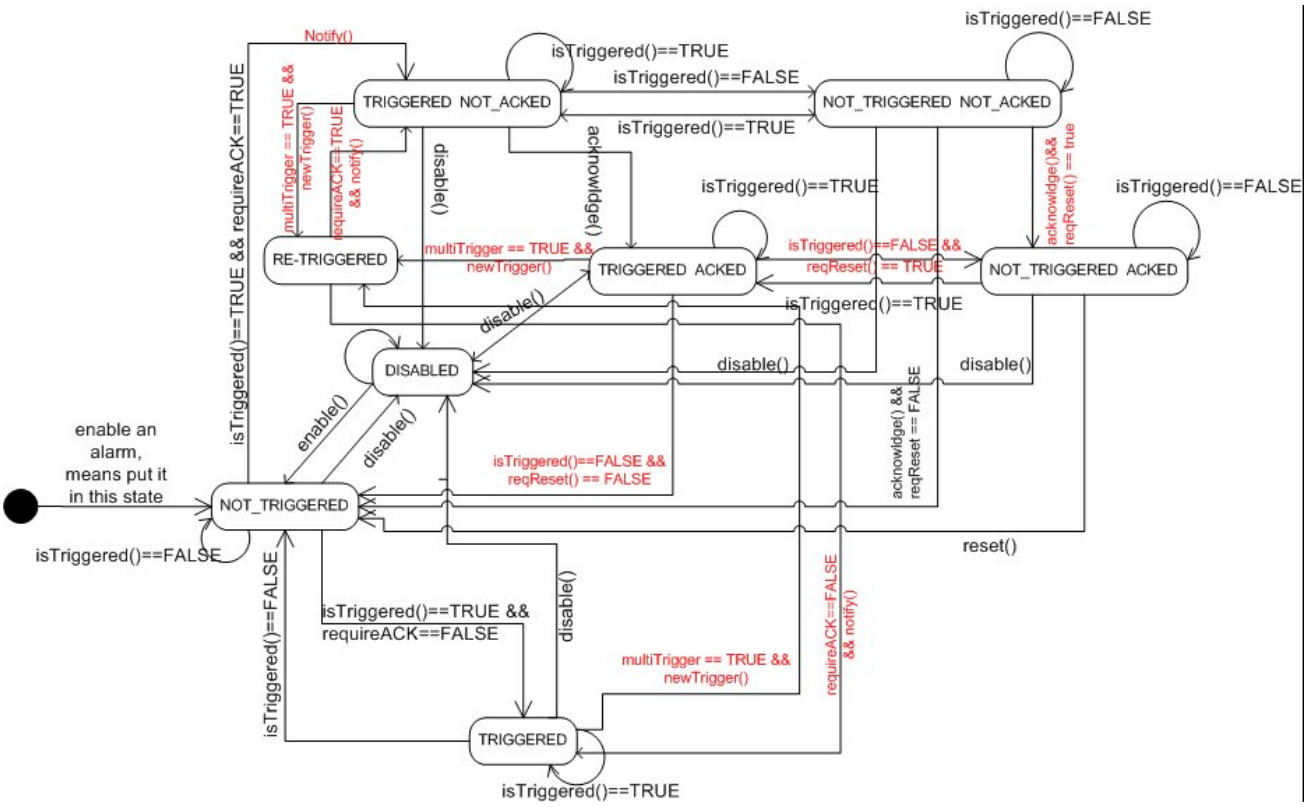
WARNING: When an alarm is triggered, some signals need to be update/communicated through the connected devices. We assume the Acknowledge to be a signal pushed from an operator and not released automatically from a controller device. This allows for time required to communicated the original signals.

Tip: Using the same tag both for **Remote Ack** and **Ack Notify** can connect more devices to the same controller and acknowledge the alarms from any HMI device.

Alarm state machine

The runtime implements the alarm state machine described in this diagram.

States and transitions between states are described according to the selected options and desired behavior.



Setting events

Path: **ProjectView> Config** > **Alarms** > **Events** property

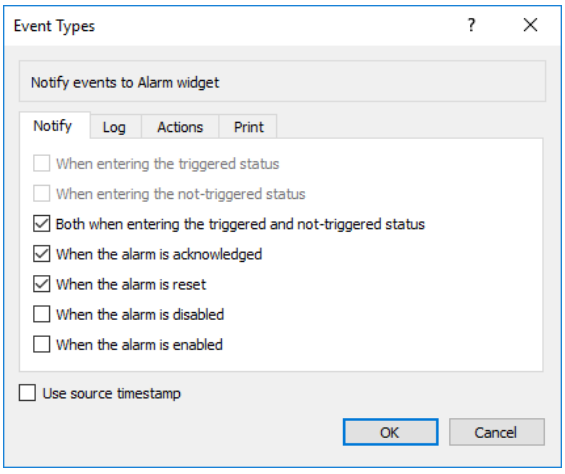
Events are defined using the Alarms Editor.

See "Alarms Editor" on page 230 for details.

Notifying events

Path: **ProjectView> Config** > **Alarms** > **Events** property > **Notify** tab

Set conditions under which the alarms will be posted in the alarm widget.



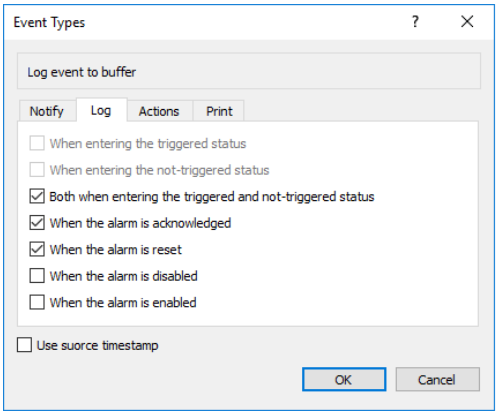
Here you define the behavior of the default alarm widget available in the Widget gallery and decide in which cases the widget is updated by a change in an alarm status.

CAUTION: Make only the adjustments required by the specific application while leaving all other settings as default.

Logging events

Path: **ProjectView**> **Config** > **Alarms** > **Events** property > **Log** tab

Set conditions for which you want to store the specific event in an alarm history buffer.

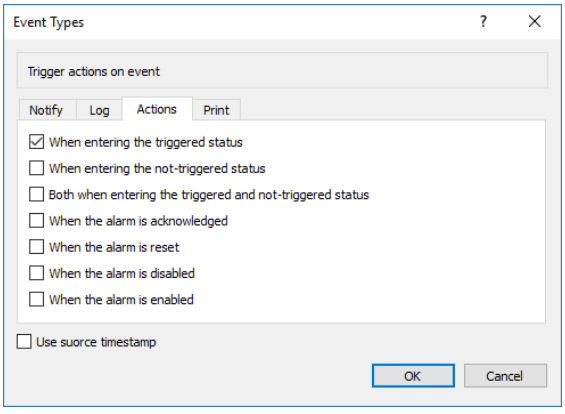


The alarm history is logged in the Event Buffer.

Executing actions

Path: **ProjectView**> **Config** > **Alarms** > **Events** property > **Actions** tab

Set conditions under which the action(s), configured for the specific alarm, must be executed.

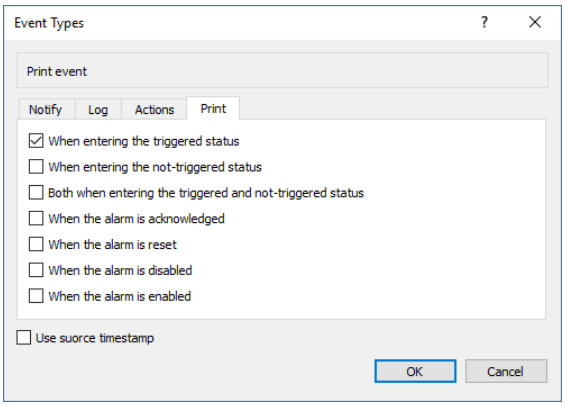


By default, actions are executed only when the alarm is triggered; other alarm states can also be set to execute actions.

Print events

Path: **ProjectView**> **Config** > **Alarms** > **Events** property > **Print** tab

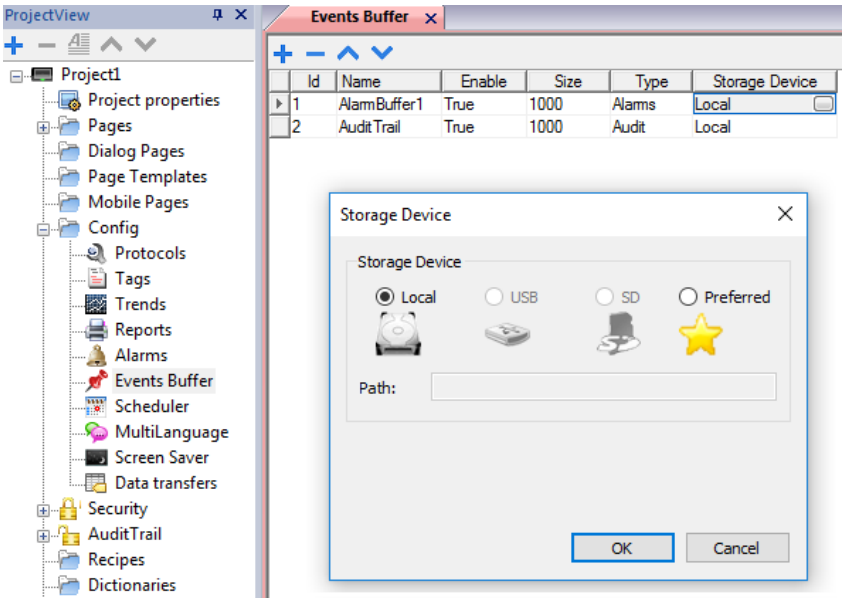
Set conditions for which you want to print the specific event



Setting storage device

Path: ***ProjectView*** *>* ***Config*** *>* ***Events Buffer*** *>* ***Storage Device*** *tab*

1. Open the **Storage Device** dialog.
2. Select a device for event data storage.



Data is automatically saved every five minutes except for alarm data which is saved immediately.

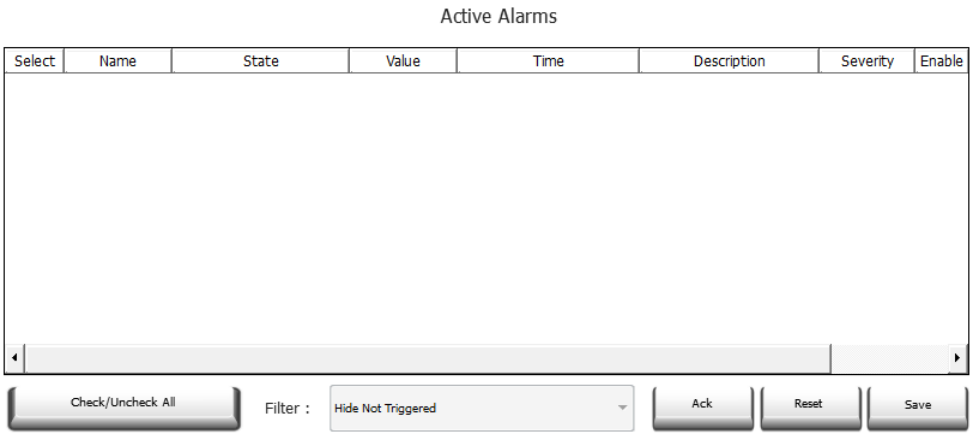
Use source timestamp

Events are stores with the timestamp of when the HMI device detect the event. When “Use source timestamp” is selected, the events are stored with the timestamp received from the remote device.

Available only for device’s protocols that support this feature (OPC UA Client)

Active Alarms widget

You can insert the **Active Alarms** widget in a page to display the alarms and to acknowledge, reset or enable/disable them.



Alarm filters

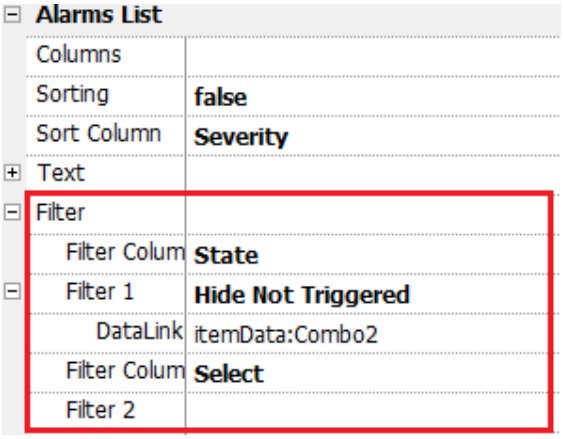
*Path: **ActiveAlarm** widget> **Properties** pane> **Filter***

Define filters used to display only some of the configured alarms. Filters are based on alarm fields, which means you can filter alarms according to name, severity, description and so on.

Filter 1 is the default filter. It's managed by the combo box **Filter 1**, and has two options: **Show all alarms** and **Hide Not Triggered** which, when selected, allows to display only active alarms.

Filter 2 is, by default, not configured and available for customization.

Filter's expressions make use of AWK language, the expressions are applied to the data contained in the selected **Filter** column of the Alarm widget.



Setting filters

*Path: **ActiveAlarm** widget> **Properties** pane> **Filter***

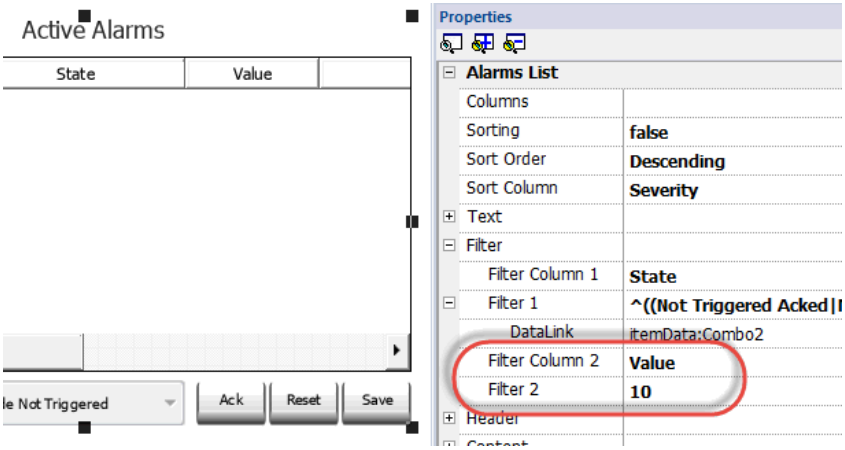
To set one of the two available filters:

1. Select **Filter Column 1** and choose the value to filter for (e.g.: Name, State, Time, Groups)
2. In **DataLink** attach a combo box widget. Use Shift+ left-click to select the combo box.
3. In the **Properties** pane select list property and open dialog to customize combo box values
4. In the combo box configuration dialog, specify **String List** and the regular expression to filter values.

See https://en.wikipedia.org/wiki/Regular_expression for additional details regarding regular expressions.

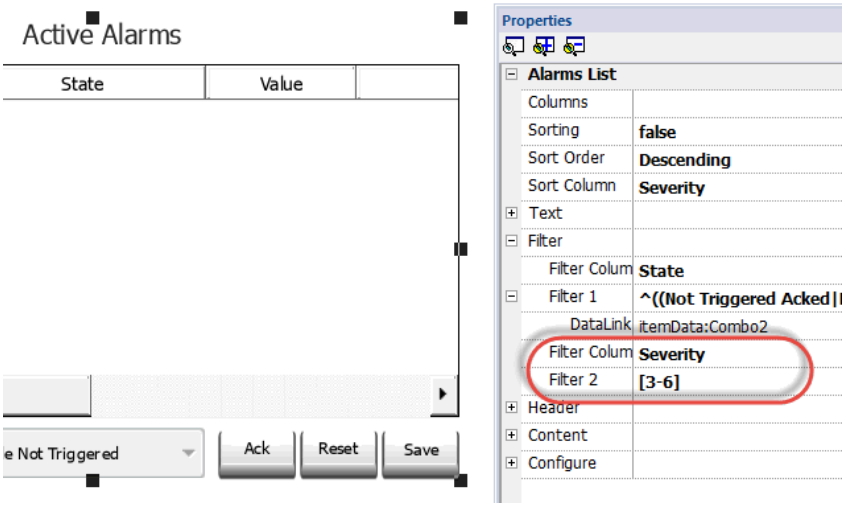
Filters first example

You want to show all alarms matching Filter 1 with value equal to 10. Then properties settings: **Filter column 2** = Value, **Filter 2** = 10



Filters second example

You want to show all alarms matching a Severity value from 3 to 6 (Normal to Critical). Then properties settings: **Filter column 2** = Severity, **Filter 2** = [3-6]



Filters third example

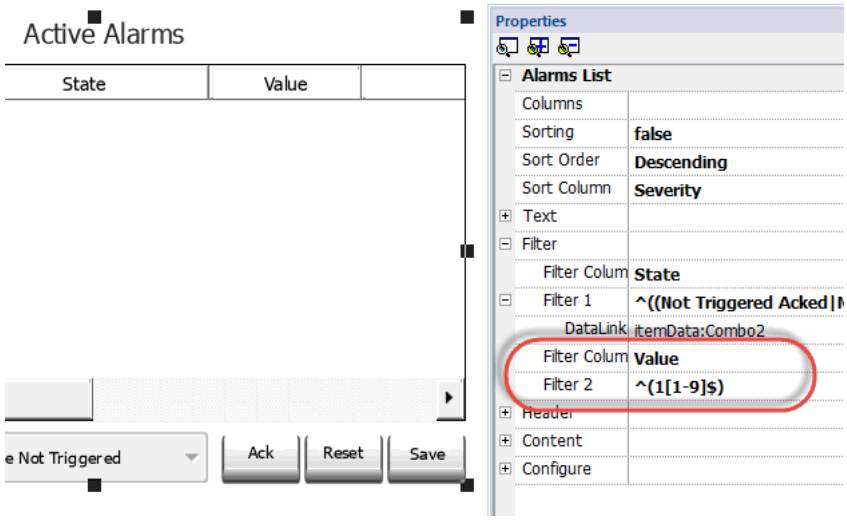
You want to show all alarms matching a value from 11 to 19. Then properties settings: **Filter column 2** = Severity, **Filter 2** = ^(1[1-9]$)

Meaning:

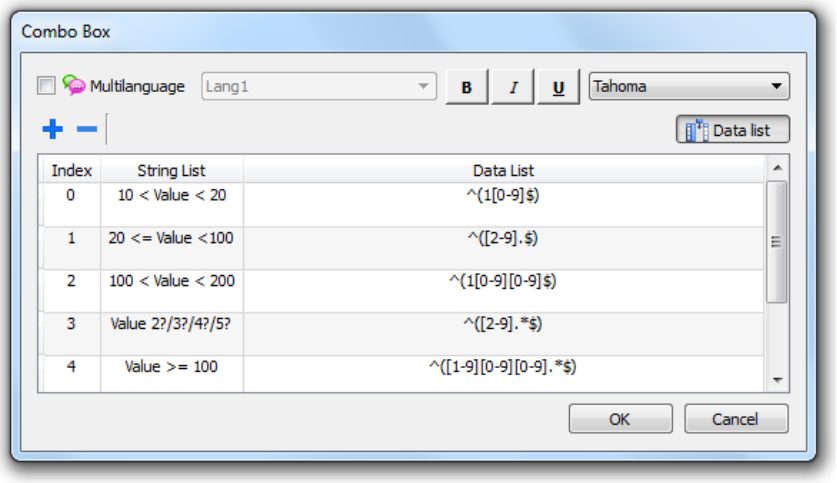
^ = match must starts from the beginning of the string

1[1-9] = first char must be 1 and the second char must be between 1 and 9

$ = end of the comparison.



Filters expression examples



Filter by	String list	Data list
State	Hide Not Triggered	^((Not Triggered Acked|Not Triggered Not Acked|Triggered).*$)
Value	10 < Value < 20	^(1[0-9]$)
Value	20 <= Value <100	^([2-9].$)
Value	100 < Value < 200	^(1[0-9][0-9]$)
Value	Value 2?/3?/4?/5?	^([2-9].*$)
Value	Value >= 100	^([1-9][0-9][0-9].*$)
Value	Value >= 20	^([2-9].*$\|[1-9][0-9][0-9].*$)

Sorting alarms

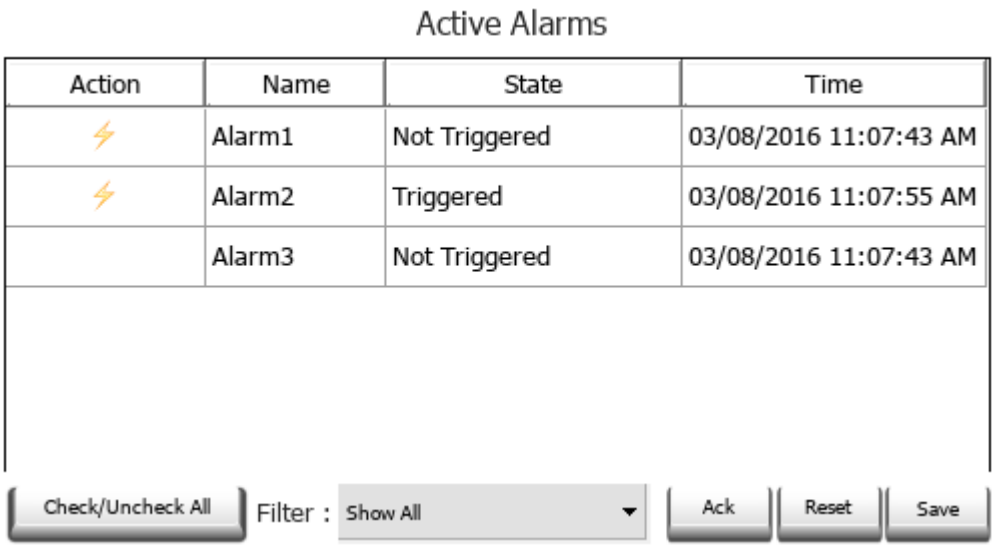
Path: ***ActiveAlarm*** *widget>* ***Properties*** *pane>* ***Sorting***

The sorting function allows you to sort alarms at runtime in the alarms widget by clicking on the column header.

Note: The severity value displayed here is set in the Alarm Editor.

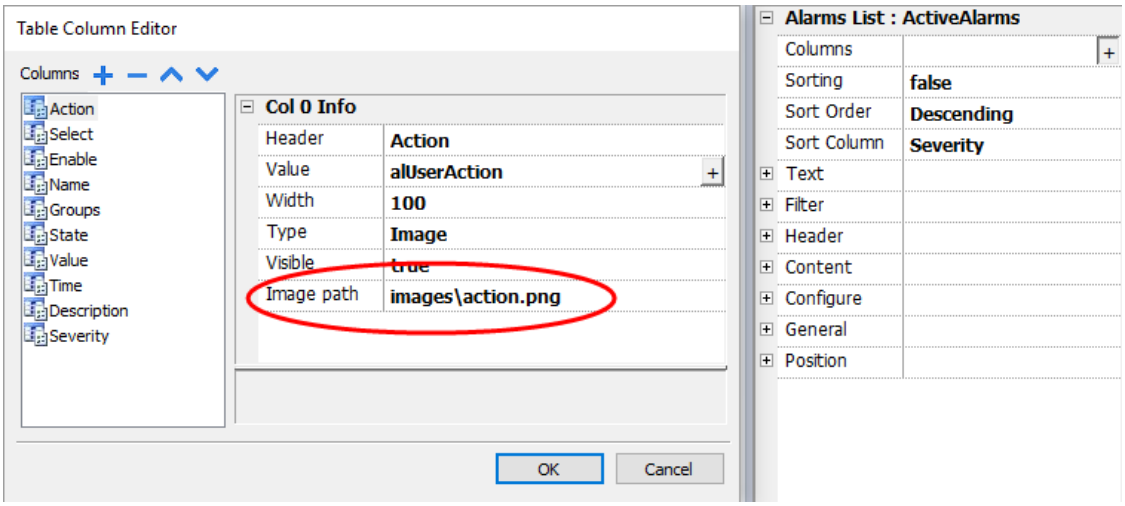
Action

When the "User Action" associate with the alarm (see ""Alarms Editor" on page 230 for details) contains valid actions, the Action icon is showed. Pressing the icon, the configured actions will be executed.



WARNING: If you are using an older converted project, you have to substitute the old Active Alarms Widget with the new one from the Widgets gallery

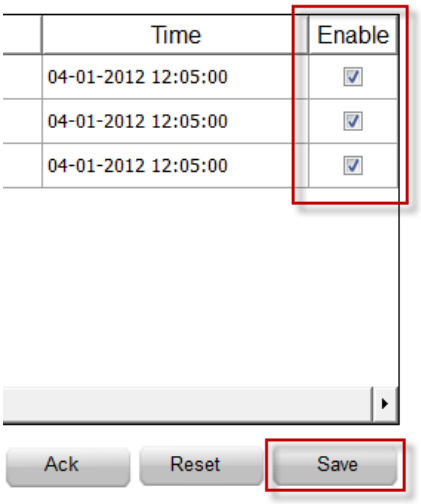
Note: The image can be modified from the Colums property of the Active Alarms widget



Enable/Disable Alarms

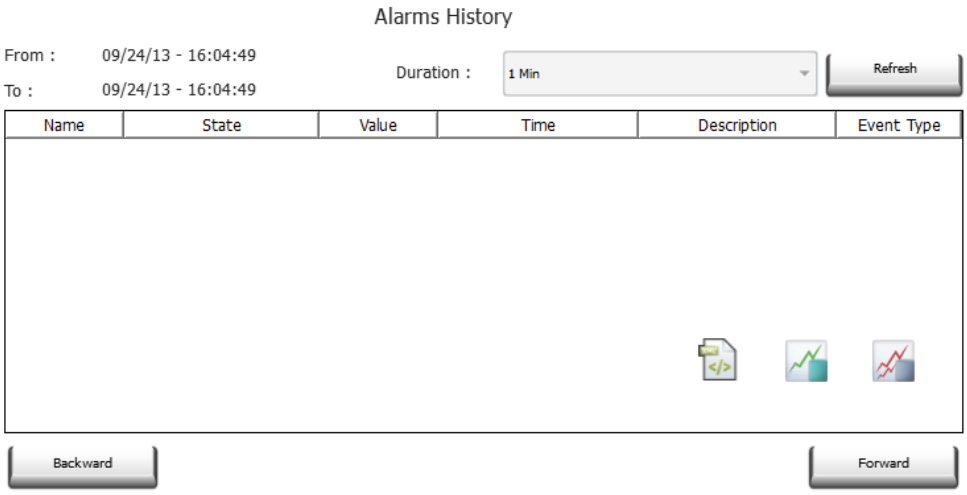
At runtime the Alarms Widget can be used to enable or disable the alarms.

Saves changes made in the **Enable** column in the alarm widget. This action is used with the **Save** button in the alarm widget.



Alarms History widget

Logs and display an alarm list if **Buffer** property in Alarms Configuration Editor is set.



Attaching widget to buffer

Path: **AlarmHistory** widget> **Properties** pane> **Buffer** > **EventBuffer**

In **Properties** pane > **Event** select the **Event Buffer** from which the alarm list is retrieved

Additional Alarms widgets

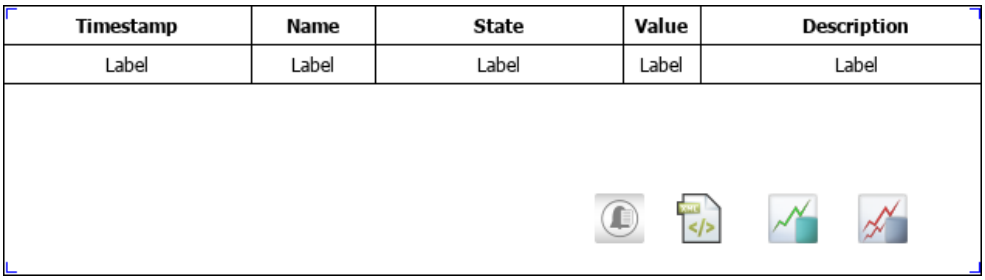
In addition to the two main “Active Alarms” and the “Alarms History” widgets, the Gallery contains some other alarms widgets with a slightly different look but basically similarly at the two main widgets. You are free to choose and use the widget that has the look that better meet your requirements.

Note that some widgets are available even inside the print report gallery.

Some widgets are based on the new table structure. For these widgets, in addition to the exposed properties, you can select the internal table and use the table capabilities to modify the widget as for your needs and taste (see "Table widget" on page 431 for additional details).

Printing the historical alarms list

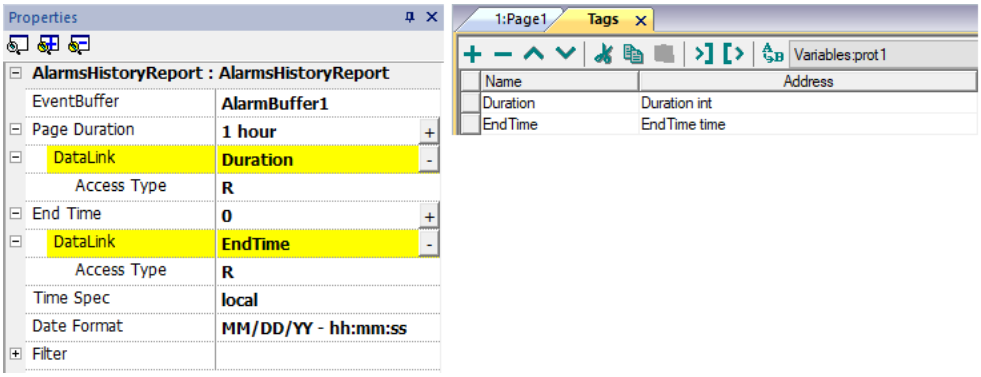
The print gallery contains historical alarms widgets, based on table structure, that can be used to generate an alarms report. The table can be drawn and enlarged to fill the entire page. If the number of lines to printed is greater of one page, the alarms table will be printed using additional pages.



Using the “attach to tag” feature is possible to use tags to define some properties of the historical alarms list to print at runtime:

- Page Duration
- End Time

"Page Duration" with "End Time" define the piece of the alarm buffer to print.



Managing alarms at runtime

When an alarm is triggered it is displayed in the Active Alarms widget where you can acknowledge and reset it. You can filter the alarms displayed using several filters, for example you can hide not triggered alarms or show all alarms.

See "Active Alarms widget" on page 236 for details.

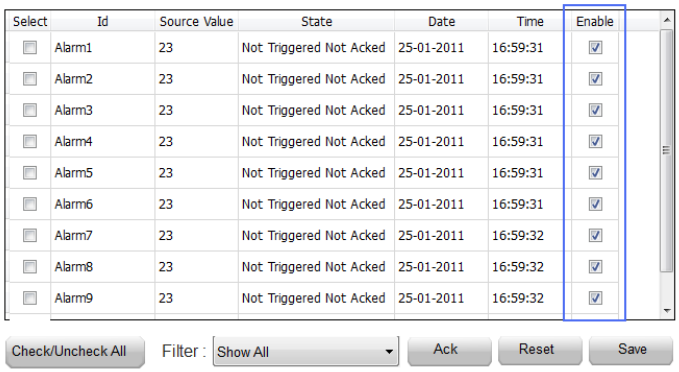
IMPORTANT: The Active Alarms widget is not displayed automatically. You must add a dedicated action that will open the page containing the alarm widget when the alarm is triggered.

Enable/disable alarms at runtime

You can enable or disable the alarms at runtime.

To enable an alarm select the **Enable** option in the alarm widget.

Disabled alarms are not triggered and therefore not displayed at runtime.



Note: Alarms can be configured to be enable/disable even from the PLC.
See Alarm Configuration Editor for details.

Displaying live alarm data

*Path: **ProjectView**> **Config** > double-click **Alarms***

Both in the Active Alarms widget and in the Alarms History widget it is possible to set the description of the alarm, or of the custom fields, to display the data of the live tags.

Id	Name	Enable	Ack	Reset	Tag	Buffer	Trigger	Action	Description
1	Alarm1	☑	☑	☑	Tag1	AlarmBuffer1	bitMask Alarm:	ShowDialog	Alarm 1 Tag Value is [Tag1]
2	Alarm2	☑	☑	☑	Tag1	AlarmBuffer1	bitMask Alarm:1	ShowDialog	Alarm 2 Tag Value is [Tag2]
3	Alarm3	☑	☑	☑	Tag1	AlarmBuffer1	bitMask Alarm:1	ShowDialog	Alarm 3 Tag Value is [Tag3]

To show the tag value, set a placeholder in **Description** entering the tag name in square brackets, for example "[Tag1]". At runtime, in **Description** column of Active Alarms widget the current value of the tag will be displayed.

Live Tags Placeholders

Tags

- [TagName]
  The tag value is read and continuously updated

Use '\' before '[ ]' if you want to show the '[ ]' in the description string, for example: *\[Tag\[1\]\]* will display the string *"[Tag[1]]"*.

Use '\', even when the tag label contains square brackets. For example, to display the live tag value of tag “TAG]3” or “TAG[3]” use:

- TAG\]3 = **[**TAG\]3**]**
- TAG\[3\] = **[**TAG\[3\]**]**

Array Tags

To reference the entire array (all elements will be shown):

- [TagName]
  All array elements will be displayed using a comma separate list.
- [TagName[-1]]
  All array elements will be displayed using a comma separate list.

To reference an element of the array:

- [TagName.Index]
  Example: [MyARRAY.5] will display the sixth element of the MyARRAY
- [TagName[TagIndex]]
  Example: [TagIndex] will display the sixth element of the MyARRAY when TagIndex is 5

Data Formats

Placeholder characters can be used to control how to display the tag value (see "Custom Formats" on page 28)

- [TagName|format("###")]

Example:

Live: [fCounter|format("#.00")] - Triggered: [!fCounter|format("#.00")]

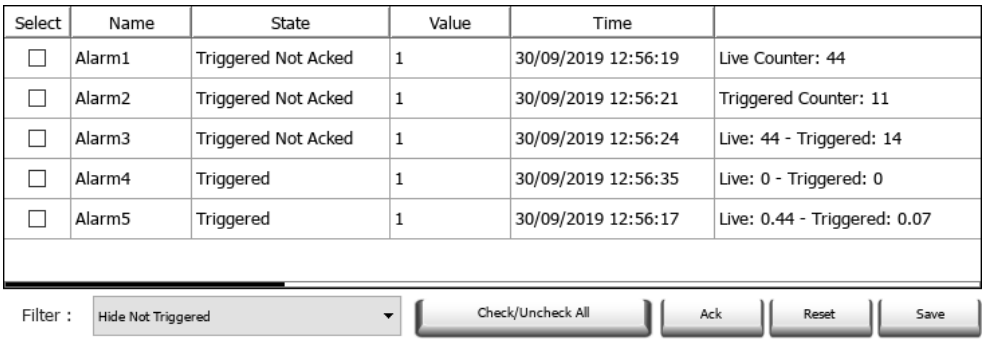
Note that by default, all tags are displayed as an integer. If you want to display a float number, you have to specify how to show the number adding the decimal digits.

To freeze a live tag value

Live tags are read and continuously updated. If you want to freeze the tag value at the instant the alarm is triggered, use the exclamation point as tag name prefix:

- [TagName]
  When alarm is triggered, tag value is read and continuously updated
- [!TagName]
  When alarm is triggered, tag value is read and frozen

Example of Alarm widget



In History Alarms widget or in .csv file, live tag values are the values taken when the alarm's status change (for both types of placeholders)

Length limit of the Description field

Number of live tags that can be used inside each alarm's description depends on size of used tags. LRH SW will check and show a warning message when too many tags are used.

The sum of the bytes that are calculated using the underlying algorithm must be less than or equal to 50

```
   4 bytes + (size of tag + 2) + (size of tag + 2) + (size of tag + 2) + . . .
<= 50
```

Example:

Alarm Description:
Tag1=**[TagInt]**, Tag2=**[TagBool]**, Tag3=**[TagStr8]**

Fixed	4	
Tag1	6	4 (sizeof-INT) +2
Tag2	3	1 (sizeof-BOOL) +2
Tag3	10	8 (sizeof-STR8) +2
Total:	23	

When arrays are used, e.g. Tag1 as an array of 8 integer:

- **[Tag1]** or **[Tag1[-1]]**
  The entire array is shown and the number of the necessary bytes is calculated as 4(size-INT) x 8(array elements) +2 = 34 Byte
- **[Tag1[Index]]**
  An element of the array is shown and the number of the necessary bytes is calculated as 4(size-INT) x 1(array elements) +2 = 6 Byte. In this case, if at runtime the Index assumes the value -1 some values could be lost

Exporting alarm buffers to .csv files

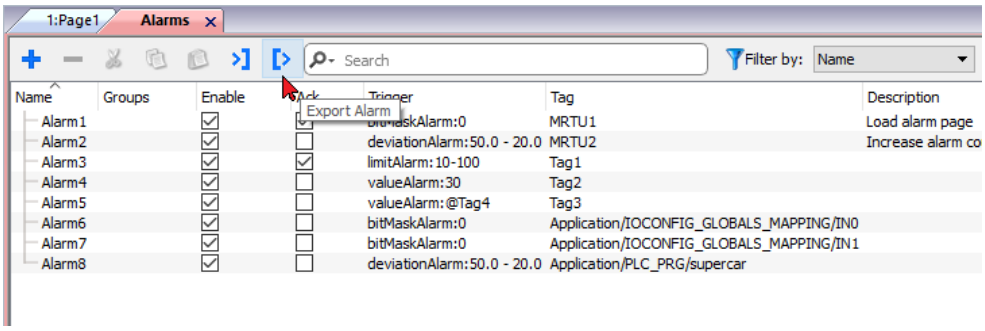
To export an event buffer containing an history alarms list, use the **DumpEventArchive** action.

See "System actions" on page 192 for details.

Note: Tag values displayed in the alarms description are also included in the buffer. Tags are sampled when the alarm is triggered and that value is logged and included in the description.

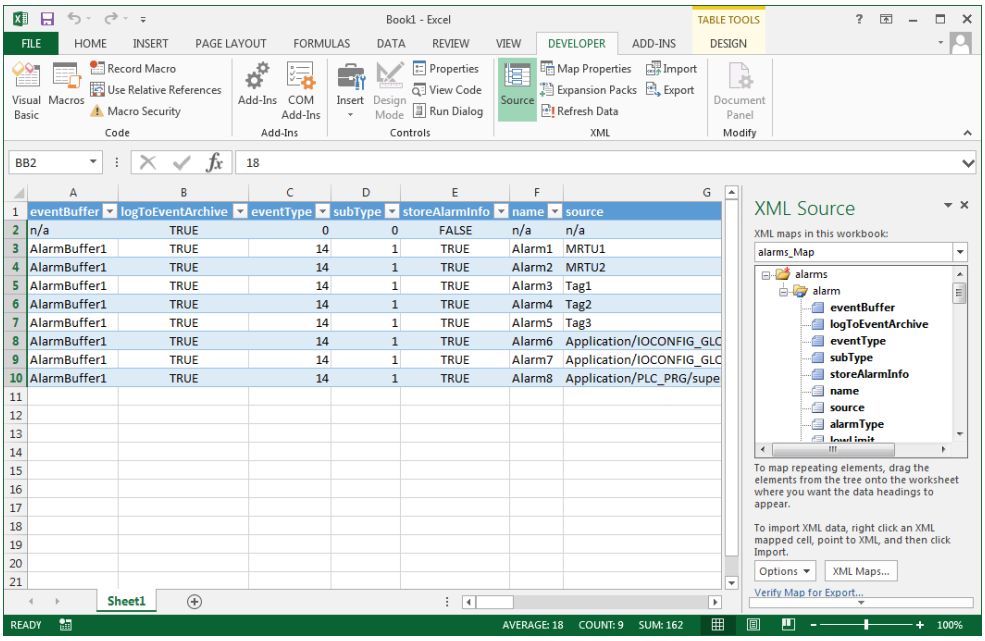
Exporting alarm configuration

Path: ***ProjectView*** *>* ***Config*** *> double-click* ***Alarms***



Click the **Export Alarms** button: the alarms configuration table is exported into an .xml file.

You can edit the resulting .xml file using third part tools (for example, Microsoft Excel) .

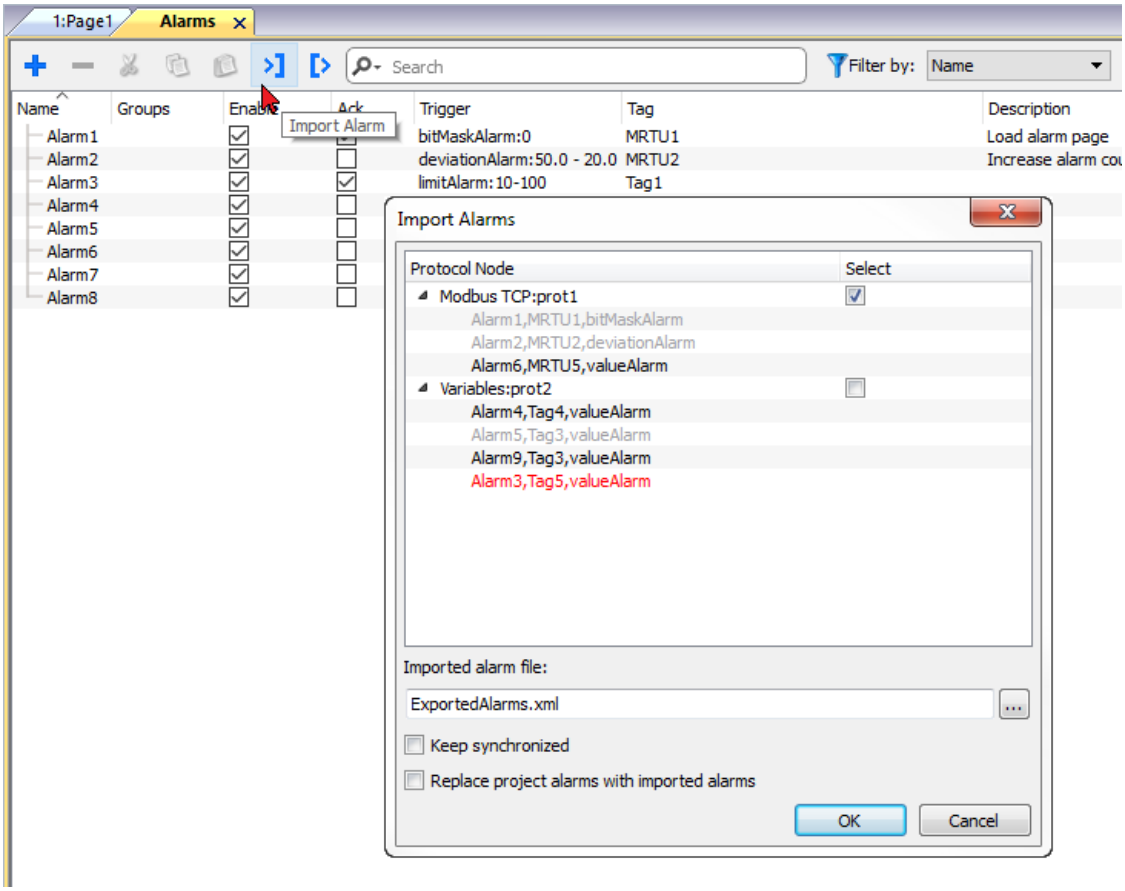


Warning: the bitMask values are reported as 2^BitPosition in Hexadecimal format.

alarmType	lowLimit	highLimit	value	bitMask	deviation	setPoint	enableTag
n/a	0	1000	0	1	50	20	n/a
bitMaskAlarm				1			
bitMaskAlarm				2			
bitMaskAlarm				4			
bitMaskAlarm				8			
bitMaskAlarm				10			
bitMaskAlarm				20			
bitMaskAlarm				40			
bitMaskAlarm				80			
bitMaskAlarm				100			
bitMaskAlarm				200			
bitMaskAlarm				400			
bitMaskAlarm				800			
bitMaskAlarm				1000			
bitMaskAlarm				2000			

Importing alarm configuration

Path: ***ProjectView*** *>* ***Config*** *> double-click* ***Alarms***



1. Click the **Import Alarms** button and select the .xml file from which to import the alarms configuration: the **Import Alarms** dialog is displayed.
2. Select the group of alarms to import and click **OK** to confirm.

Differences are highlighted in the **Import Alarms** dialog using different colors

Color	Description
Black	This is a new alarm and it will be imported
Red	This alarm has not been found and will be removed (only if check "Replace project alarms with imported alarms" is checked)
Blue	This alarm has been modified and will be updated.
Gray	This alarm is already part of the project and will be skipped.

Automatic synchronization

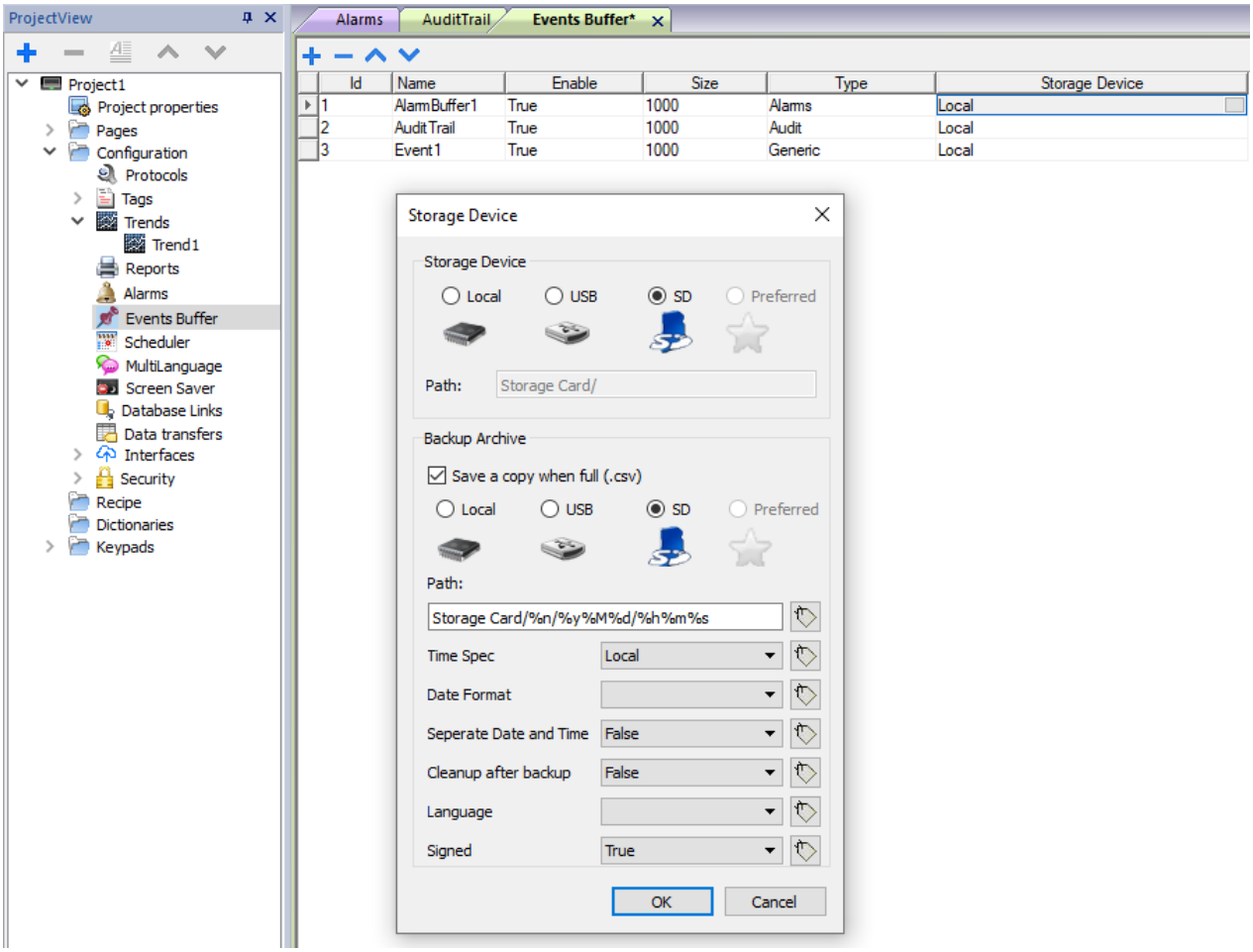
Select the **Keep synchronized** option in the **Import Alarms** dialog to enable the automatic synchronization of the alarm configuration file.

Whenever changes occur in the alarms configuration, the file will be automatically updated in silent mode.

Tip: Enable this function when the alarm file is managed by a different tool (for example, PLC programming software) as well as by LRH SW.

Events Buffer

The "Events Buffer" page gives you the possibility to configure the current events buffers (used for store alarms or audit trail information) or add additional events buffers.



Parameter	Description
Id	Buffer identification number
Name	Buffer name
Enable	Enable/disable logging
Size	Size of log file. Data is automatically saved to disk every 5 minutes.
Type	Type of events logged: • **Alarms** • **Audit** • **Generic**
Storage Device	Device where the data will be stored

Backup Archive

If **Save a copy when full** option is enabled, the HMI device will save a copy when the events buffer is full before it is overwritten by newer data.

Parameter	Description
Path	Where events buffer data will be copied. The below wild cards are supported • %n = Events buffer name • %y = Year • %M = Month • %d = Day • %h = Hour • %m = Minutes • %s = Seconds
Time Spec	Timestamp of events • Local Use the time of the HMI device where the project is running • Global Use global time (GMT)
Date Format	Time and Date format. Placeholders can be used (see "Time and Date placeholders" on page 413)
Separate Date and Time	When "true", the date and the time are placed into two different fields
Cleanup after backup	When "true", the event buffer is clean up after completing the backup. When "false", the older events are removed when new events are incoming (circular buffer)
Language	Language to use
Signed	When "true", the additional file with the signature is added (see "Signed CSV files" on page 318)

20 Recipes

Recipes are collections of tag values organized in sets that satisfy specific application requirements.

For example, if you have to control room variables (temperature and humidity) in the morning, afternoon and evening. You will create three sets (morning, afternoon and evening) in which you will set the proper tag values.

Each element of the recipe is associated to a tag and can be indexed into sets for a more effective use. This feature allows you to extend the capabilities of controllers that have limited memory.

You can add controller data to a page using a recipe widget. Recipe data contains all the controller data items; however data is no longer read directly from the controller but rather from the associated recipe element in the HMI device.

Recipe data is configured in LRH SW workspace; the user can specify default values for each element of the data records. In LRH SW HMI Runtime, data can be edited and saved to a new data file, any change to recipe data is therefore stored to disk. With the use of a separate data file LRH SW HMI Runtime ensures that modified recipe values are retained throughout different project updates. In other words, a subsequent project update does not influence the recipe data modified by the user in the LRH SW HMI Runtime.

See "Recipe actions" on page 187 for details on how to reset recipe data.

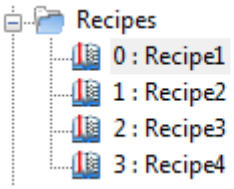
Note: Recipe data can be stored on a Flash memory, on a USB drive or on a SD card.

Managing recipes

Creating a recipe

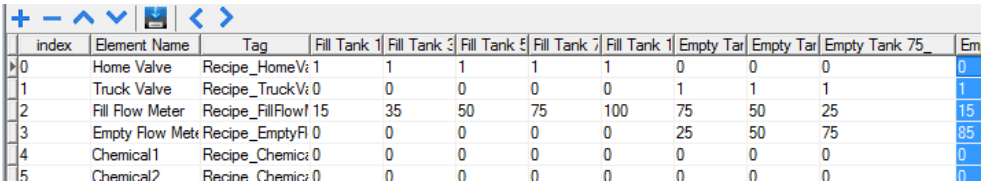
To create a recipe for your project:

1. In **ProjectView** right-click **Recipes** and select **Insert Recipe**: an empty recipe is added. You create and configure recipes using the Recipe Editor.



Recipe editor

Path: ***ProjectView*** *>* ***Recipes*** *> double-click* ***RecipeName***

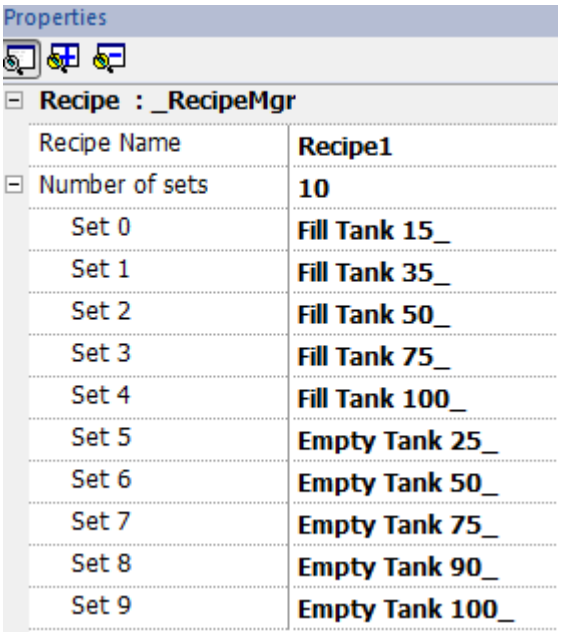


index	Element Name	Tag	Fill Tank 1	Fill Tank 3	Fill Tank 5	Fill Tank 7	Fill Tank 1	Empty Tar	Empty Tar	Empty Tank 75_	Em
0	Home Valve	Recipe_HomeVa	1	1	1	1	1	0	0	0	0
1	Truck Valve	Recipe_TruckVa	0	0	0	0	0	1	1	1	1
2	Fill Flow Meter	Recipe_FillFlowI	15	35	50	75	100	75	50	25	15
3	Empty Flow Met	Recipe_EmptyFl	0	0	0	0	0	25	50	75	85
4	Chemical1	Recipe_Chemica	0	0	0	0	0	0	0	0	0
5	Chemical2	Recipe_Chemica	0	0	0	0	0	0	0	0	0

Configuring recipe properties

In the **Properties** pane of each recipe you set the following parameters:

Parameter	Description
Recipe Name	Name of the recipe
Number of sets	Number of values sets for each recipe element. Each set has a different configurable name.

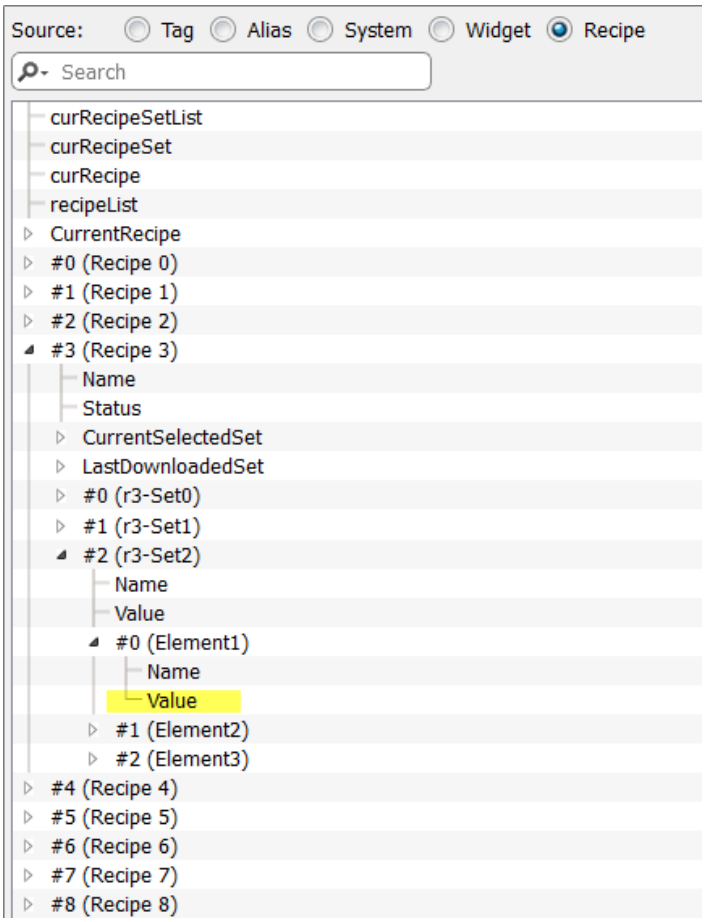


Setting up a recipe

1. Click **+** to add an element of the recipe.
2. Link the tags to each recipe element.

Defining recipe fields

Create a recipe field in the page using a numeric widget and attaching it to a recipe item after selecting Recipe as the Source.



In the **Attach to** dialog you have the choice of all the different recipe variables, such as:

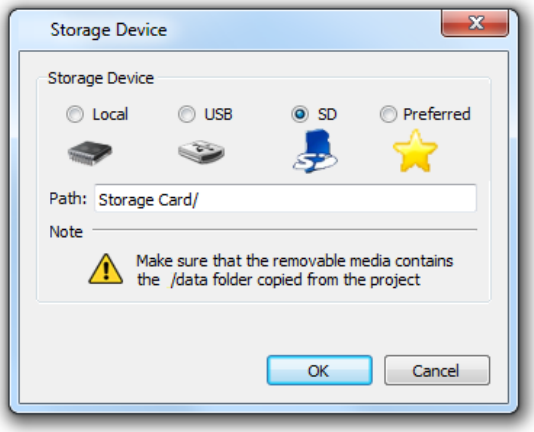
- Current Recipe >Current Selected Recipe Set> Element > Value
- Selected Recipe > Selected Set0 > Element > Value
- recipeList

When numeric widgets are defined as read/write, the default recipe data can be edited at runtime. These new values are stored in a separate file as modified recipe data.

Note: Since JavaScript API functions are used, the recipe elements and sets can be referenced by name or by position. To avoid ambiguity between names and index, the names of the recipe elements and sets must include at least one alphanumeric character.

Storing recipe data

In the Recipe Editor click the storage type icon to select where to store recipe data: the **Storage Device** dialog is displayed.



For USB drive and SD card storage you can provide the folder location.

WARNING: Recipe configuration files are created automatically when the project is saved and stored in the data subfolder of the project. To use external storage devices, you need to copy this folder into the external device. Note that you have the responsibility to manage the data folder inside external devices. Even dynamic files are not deleted when project is updated using the "Delete dynamic file" option.

Important: You can add a subfolder but you must not rename the "data" subfolder.

Import/Export recipes

To import/export the recipes configuration of your project:

In **ProjectView** right-click **Recipes** and select **Export Recipe** or **Import Recipe**

The following formats are supported for import:

- Comma Separated Values (.csv)
- Unicode Text (.txt)

Note: Use the Unicode Text file format when you import a file modified using Microsoft® Excel®.

Configuring a recipe widget

You can choose one of the two recipe widgets available in the **Widget Gallery**:

- **Recipe set**: allows you to select a recipe set for upload or download. See "Uploading/downloading a recipe" on page 256
- **Recipe menu**: when more recipes have been created for a project, use this widget to manage all recipes and select the desired sets for each of them.

Configuring the Recipe Set widget

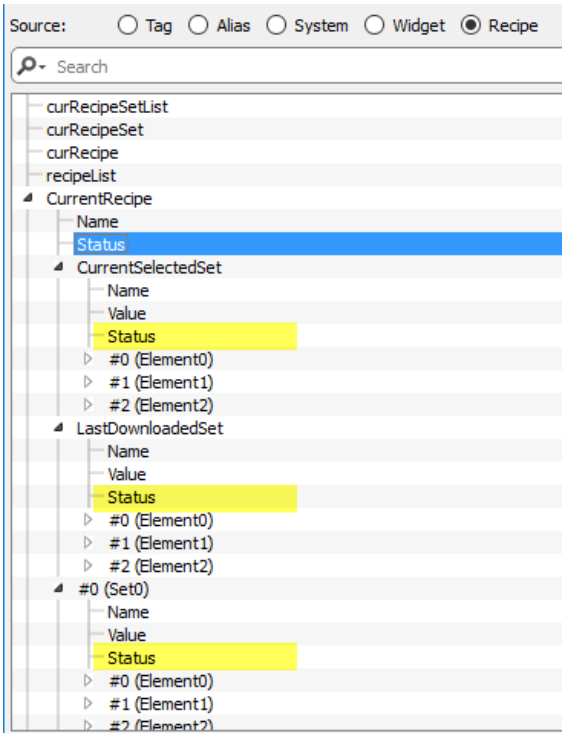
In the **Properties** pane of each **Recipe Set** widget set the following parameter:

Parameter	Description
Recipe Name	Name of the recipe

Recipe status

Each recipe contains two kinds of status parameters

- Recipe Status (blue in the below picture)
  Give information regarding the last download or upload operation
- DataSet Status (yellow in the below picture)
  Give information of modified datasets



Recipe Status

After every recipe upload or download, or recipe set modification, the **Recipe Status** parameters contain a value with the result of the operation.

Code	Function	Description
0	Set modified	Selected set changed
1	Download triggered	Download request triggered
2	Download Done	Download action completed
3	Download Error	Error during download (for example, unknown set, unknown recipe, controller not ready, Tags write failed etc.)
4	Upload triggered	Upload request triggered
5	Upload done	Upload action completed
6	Upload Error	Error during upload - same as for download
7	General Error	General error (for example, data not available)

DataSet Status

The status of each data set indicates that it has been changed. This information may be useful to not forget to download the recipe to synchronize the PLC. Both download or upload operations will reset the **DataSet Status** to 0.

Code	Function	Description
0	Syncronized	User synchronized PLC with the dataset values
1	Modified	User modified some values of the dataset

Note: After a device startup or a recipe reset/restore, all status values will reset to 0.

Uploading/downloading a recipe

Uploading a recipe

You upload a recipe to an HMI device using a recipe widget and the **UpLoadRecipe**, **UpLoadCurRecipe** action in one of the following ways:

- attach the action to an event of a button or a switch (see ""Attach to" parameters" on page 39 for details)
- configure the action in an alarm action list (see "Alarm actions" on page 170 for details)
- configure the action in a scheduler action list (see "Scheduling events at runtime" on page 308 for details)

Downloading a recipe

You download a recipe from an HMI device using a recipe widget and the **DownloadRecipe**, **DownLoadCurRecipe** action. See "Recipe actions" on page 187

Backup and restore recipes data

The recipe data stored in an HMI device can be exported for backup and later restored. This is done using the **DumpRecipeData** or the **RestoreRecipeData** actions.

See "Recipe actions" on page 187 for details.

21 Trends

Trends allow you to sample and record the values of specified tags according to specific sampling conditions. The trend function includes trend acquisition and trend display.

Trend acquisition parameters are set in the Trend editor so that data can be stored. Stored data can then be displayed in a graphical format using a trend widget.

Data logging

Data can be logged and stored to HMI memory. Data logging allows you to store the values of a group of tags all at the same time to a buffer. Data logging can be triggered by a timer or by a dedicated tag. Logged data can be exported to a .csv file or displayed using the historical trend widget. Logged data can be saved locally on a USB device or SD card, or on any available custom network folder.

WARNING: The operation with removable memory devices (USB Flash drives, SD memory cards) containing a very large number of files may result in a decrease of system performance.

WARNING: The max number of files inside a SD memory card depends on the type of formatting (e.g. FAT32 max 65536 files; FAT max 513 files).

WARNING: Flash cards support a limited number of write operations. We suggest to use only good quality memory cards; in the case your application use intensively the memory card consider a regular substitution of the memory card.

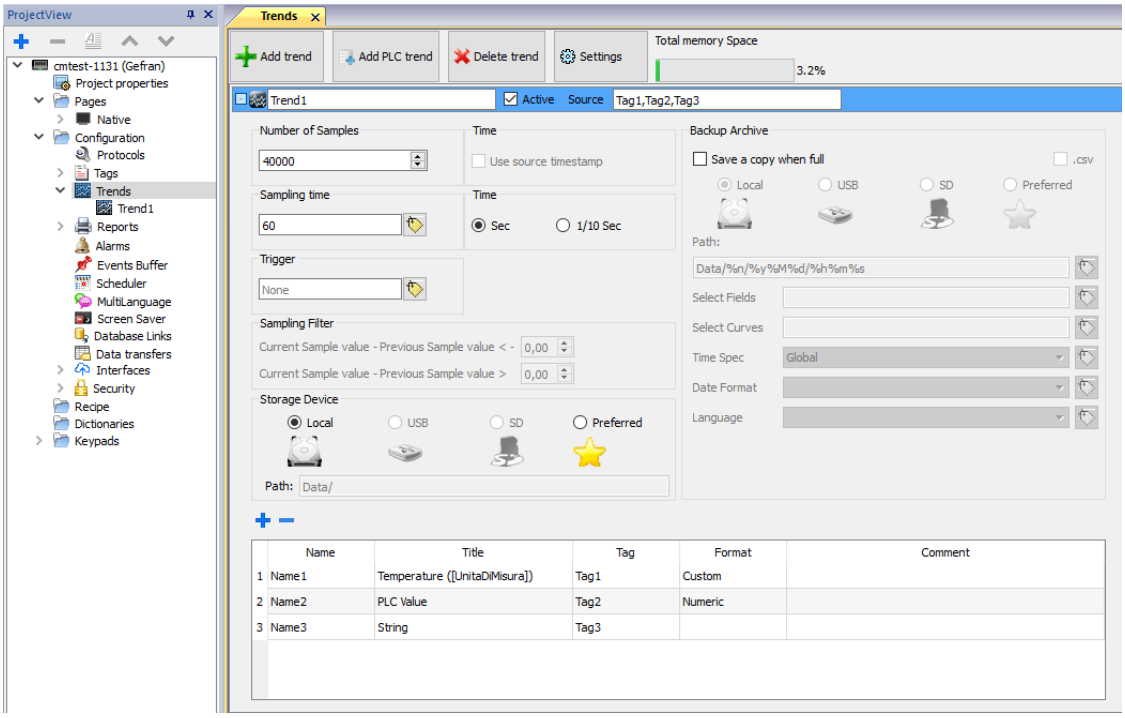
WARNING: If the data/time is moved back, the samples with invalid date/time are removed from the trend buffer. When system detects that data/time is invalid (e.g. battery low), a popup is shown to advise the user and the date/time of the last sample is used to avoid losing data.

Storage is based on trend buffers. Trend buffers are organized as a FIFO queue: when the buffer is full, the oldest values are discarded unless you configure your trend to create a backup copy of the buffer.

Adding a trend buffer

*Path: **ProjectView*** > ***Config*** > *double-click **Trends***

1. Click **Add** to add a new buffer.
2. Click **+** next to each trend buffer to display all configuration parameters.



Toolbar Element	Description
Add trend	Add a trend that will be sampled from the HMI device.
Add PLC trend	Add a trend that will be managed and sampled from the external device, instead of from the HMI device. You need a device that supports this feature to use it. The parameters depend on the used device, refer to the manual of the selected device.
Delete trend	Remove the select trend.
Settings	Offers the ability to customize the labels that appear in the trend dump header and trend table widgets. • Timestamp • Date • Time • Quality The below placeholders can be used: • **" \n "** (space + \n + space) can be used to split the label into two or more lines • **[TagName]** (tag name enclosed in square brackets) can be used to display a tag value
Total memory Space	Memory used by the defined trend buffers.

Trend Header	Description
Trend Name	Name of the trend collection (set of tags sampled at the same time)
Active	When enabled, the trend runs by default at system startup. Note: Trends cannot be activated at runtime.
Source	List of the tags sampled by the trend.

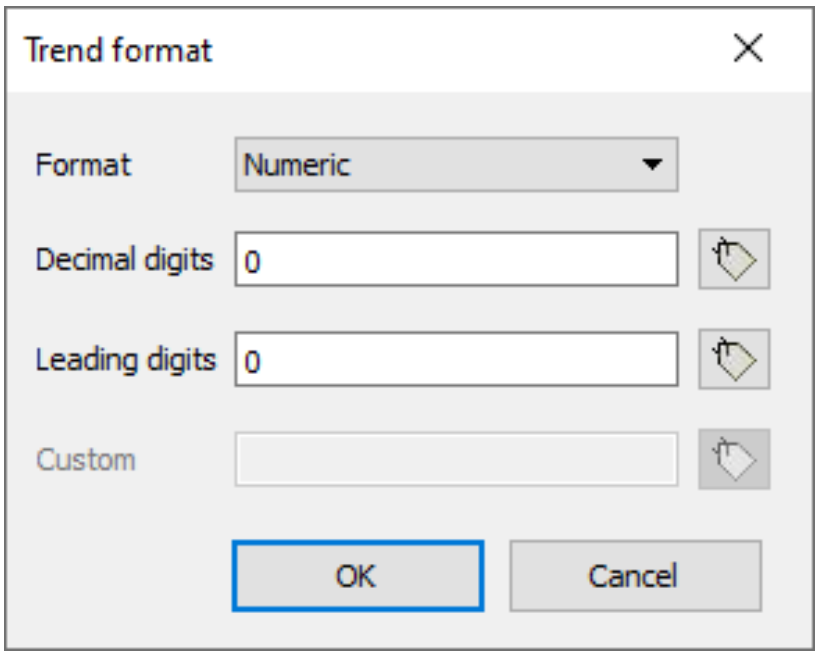
Trend Element	Description
Number of Samples	Trend buffer size (see "Number of Samples" on page 264 for additional information)
Sampling Time	Sampling interval. Note that instead of a constant, you can use a Tag to define/change the sample time at runtime. When sample time is 0, or negative, sampling is suspended.
Time	Time unit for the sample time. Could be 1 second (default) or 1/10 seconds Be aware that increasing the sampling rate could impact global HMI device performances.

<table>
<tr><th>Trend Element</th><th>Description</th></tr>
<tr><td>Timestamp</td><td>When checked, samples are stored using the timestamp provided from the remote device.
Available only:
for device’s protocols that support this feature (OPC UA Client)
when trend buffer is configured to with a single tag</td></tr>
<tr><td>Trigger</td><td>Tag triggering the sample.
If used, when the value of this tag changes, a sample is collected.
  Note: Trigger and Source can refer to the same tag.</td></tr>
<tr><td>Storage Device</td><td>Where trend buffer data will be stored.</td></tr>
<tr><td>Backup Archive</td><td>If Save a copy when full option is enabled, a backup copy of the buffer data is created before it is overwritten by newer data.
<table>
<tr><td>.csv</td><td>Backup data using text CSV format.</td></tr>
<tr><td>Path</td><td>Where trend buffer data will be copied.
The below wild cards are supported
%n = Trend name
%y = Year
%M = Month
%d = Day
%h = Hour
%m = Minutes
%s = Seconds</td></tr>
<tr><td>Select Fields</td><td>Fields that will be inside the dump file
  Note that you can use a string tag to define the fields to dump at runtime.</td></tr>
<tr><td>Select Curves</td><td>Curves that will be inside the dump file
  Note that you can use a string tag to define the fields to dump at runtime.</td></tr>
<tr><td>Time Spec</td><td>Timestamp of samples
Local
Use the time of the HMI device where the project is running</td></tr>
</table></td></tr>
</table>

Trend Element	Description	
		• Global Use global time (GMT)
	Data Format	Time and Date format. Placeholders can be used (see "Time and Date placeholders" on page 413)
	Language	Language to use
Sampling Filter / Trigger Filter	This parameter allows to specify a dynamic filter if required. When sampling is done on time basis the offset is applied to the sampled Tag value. If the new value exceeds the specified limits the new value is considered valid and stored, otherwise the new record will retain the previous saved value. When sampling is done on trigger the offset is applied to the trigger Tag value. If the trigger Tag value change exceeds the specified limits a new sample is taken and stored, otherwise no sampling will be done.	

Use the add/remove buttons to add the tags to sample

Samples	Description
Name	Trend name
Title	Title that has to appears inside the trend table or the trend dump. • The placeholder **" \n "** (space + \n + space) can be used to split the label into two or more lines • The placeholder **[TagName]** (tag name enclosed in square brackets) can be used to display a tag value Example: "Temperature ([UnitaDiMisura])" will be shown as "Temperature (°C)" when the tag UnitaDiMisura = "°C"
Tag	Tag that must be sampled **Tags string are supported until 8 bytes. If tag size is greater than 8 bytes, only the first 8 bytes are stored in trend. Unicode chars are not supported.**
Format	Display format to use. Note that even the custom format can be used (see "Custom Formats" on page 28).

Samples	Description
	
Comment	You can write whatever you want here

Number of Samples

The number of samples that you can have is dependent on the memory size reserved for trend buffers and from the size of each sample.

```
Number of available samples = Available Memory / Size of sample
```

Where the size of each sample is dependent on how many tags are used and can be calculated using the below formula:

```
Size of sample = TAGS*9 + 11
```

You are free to use the entire available memory for a unique trend buffer or split the available memory over several trends.

See also:

- Trend limits on "Table of functions and limits" on page 540
- Reserved memory for trend buffer on "HMI devices capabilities" on page 541

Exporting trend buffer data

Use the **DumpTrend** action to export trend buffer data to a .csv file. See "DumpTrend" on page 192 for the macro parameters details.

The exported .csv file could have different formats defined from the Dump Trend macro parameters. The different formats are maintained mainly for compatibilities reasons.

FileFormat: **Compatibility CSV**

	A	B	C	D	E	F	G	H	I	J	K
1	Type	Value	Time Stamp	Refresh Time	Quality	Type	Value	Quality	Type	Value	Quality
2	4	0	2015-09-18T14:42:22.000Z	1000	192	8	0.00E+00	192	3	0	192
3	4	0	2015-09-18T14:42:23.000Z	1000	192	8	0.00E+00	192	3	0	192
4	4	0	2015-09-18T14:42:24.000Z	1000	192	8	0.00E+00	192	3	0	192
5	4	40	2015-09-18T14:42:25.000Z	1000	192	8	0.00E+00	192	3	0	192
6	4	40	2015-09-18T14:42:26.000Z	1000	192	8	0.00E+00	192	3	0	192
7	4	40	2015-09-18T14:42:27.000Z	1000	192	8	0.00E+00	192	3	0	192
8	4	40	2015-09-18T14:42:28.000Z	1000	192	8	5.00E+01	192	3	0	192
9	4	40	2015-09-18T14:42:29.000Z	1000	192	8	5.00E+01	192	3	0	192
10	4	40	2015-09-18T14:42:30.000Z	1000	192	8	5.00E+01	192	3	0	192

FileFormat: **Compact CSV**

	A	B	C	D	E	F	G
1	Timestamp	Tag1	4	Tag2	8	Tag3	3
2		Value	Quality	Value	Quality	Value	Quality
3	2015-09-18T14:42:22.000Z	0	192	0.00E+00	192	0	192
4	2015-09-18T14:42:23.000Z	0	192	0.00E+00	192	0	192
5	2015-09-18T14:42:24.000Z	0	192	0.00E+00	192	0	192
6	2015-09-18T14:42:25.000Z	40	192	0.00E+00	192	0	192
7	2015-09-18T14:42:26.000Z	40	192	0.00E+00	192	0	192
8	2015-09-18T14:42:27.000Z	40	192	0.00E+00	192	0	192
9	2015-09-18T14:42:28.000Z	40	192	5.00E+01	192	0	192
10	2015-09-18T14:42:29.000Z	40	192	5.00E+01	192	0	192

Note: The first row of the header contains the tags names and tags data types

FileFormat: **Compact CSV** with columns' selection

	A	B	C	D	E	F	G	H
1	Date	Time	Name1(int)	Quality	Name2(int)	Quality	Name3(boolean)	Quality
2	17/04/2018	07:24:29	0	192	10	192	0	192
3	17/04/2018	07:24:30	1	192	11	192	1	192
4	17/04/2018	07:24:31	2	192	12	192	0	192
5	17/04/2018	07:24:32	3	192	13	192	1	192
6	17/04/2018	07:24:33	4	192	14	192	0	192
7	17/04/2018	07:24:34	5	192	15	192	1	192
8	17/04/2018	07:24:35	6	192	16	192	0	192
9	17/04/2018	07:24:36	7	192	17	192	1	192
10	17/04/2018	07:24:37	8	192	18	192	0	192

The time required to dump a trend buffer depends on the number of samples present in the buffer, the memory type, and the HMI device type.

Example in the worst conditions

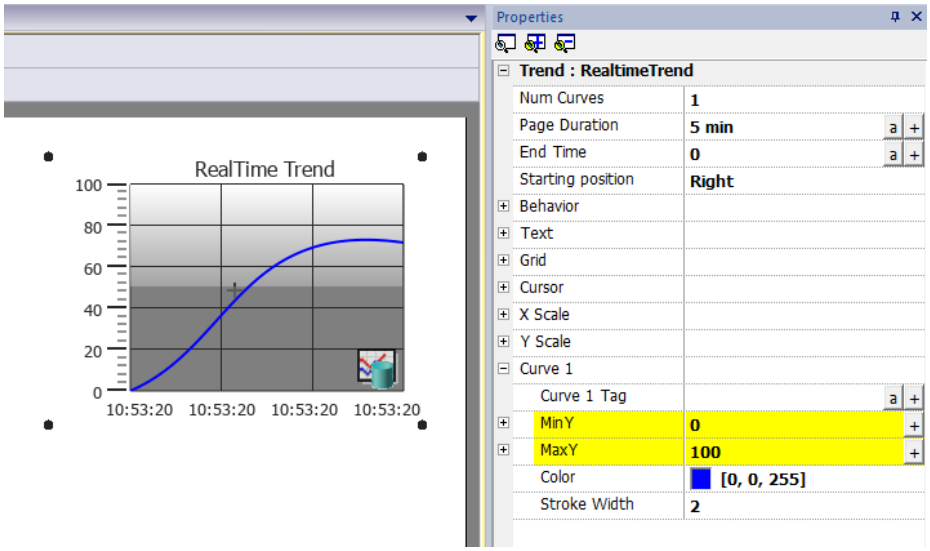
HMI Type	Buffer Size	Samples	Time
Linux	50 Mb	1.807.800 samples (2 tags)	4 Min

Realtime trend widget

The real- time trend widget can be used to display the changes of value of a tag. Data is not stored in a trend buffer and cannot be retrieved for later analysis.

To display a real-time trend:

1. Drag and drop the **RealTime Trend** widget from the widget gallery to the page.



2. Attach the tag that you want to sample to the **Curve *n* Value**. Data is always plotted against time.

RealTime trend widget properties

Property	Description
Num Curves	Number of trend curves to be displayed
Page Duration	Time window to show
End Time	End time of the time window This parameter can be used to scroll the time window. When zero, the end time is the current system time.
Starting Position	Specifies where the curve begin to be drawn when the page is opened (can be left, center or right).
Behavior	Definition of: • Min/Max of Y axis • Number of tickets to draw on the axes • Background image
Text	Trend title and font properties (font size, label, etc.)

Property	Description
Grid	Properties of grid presentation (colors)
Cursor	Properties of cursor presentation (enable and color)
X Scale	Properties of X Scale presentation
Y Scale	Properties of Y Scale presentation
Curve "n"	Tag that will be plotted in the trend widget.

Tag values can be scaled using the X Forms in the **Attach to** dialog. See ""Attach to" parameters" on page 39 for details.

History trend widget

The data collected and stored from the data logger can be analyzed using the History Trend widget.

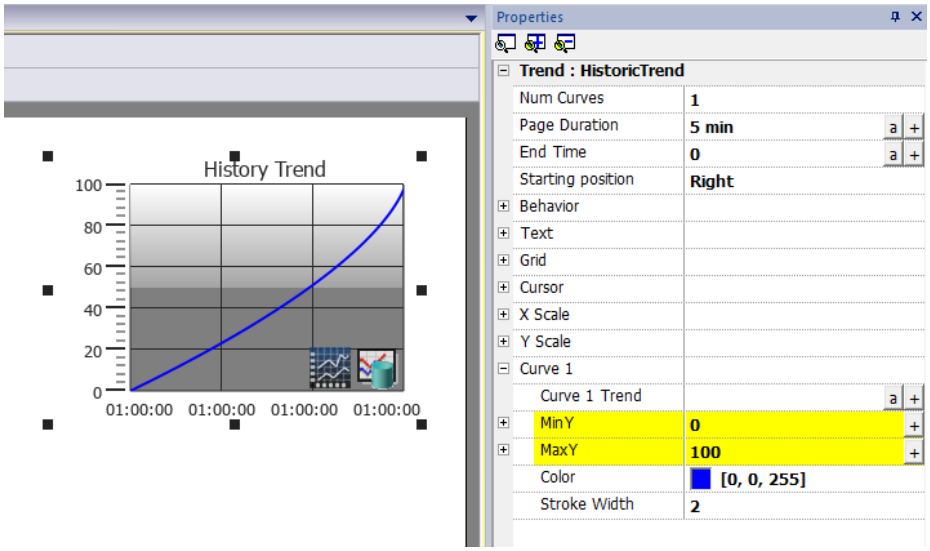
This is a two-step process:

- first you create a trend buffer to collect data for specified tags at specific points in time,
- then you configure a History Trend widget to display the collected data in a graphical format.

See "Data logging" on page 260 for details on how to create a trend buffer

To display a history trend:

1. Drag and drop the **History Trend** widget from the widget gallery to the page.



2. Attach the trend buffer that you want to draw to the **Curve *n* Value**. Data is always plotted against time.

History trend widget properties

Property	Description
Num Curves	Number of trend curves to be displayed
Page Duration	Time window to show
End Time	End time of the time window This parameter can be used to scroll the time window. When zero, the end time is the current system time.
Starting Position	Specifies where the curve begin to be drawn when the page is opened (can be left, center or right).
Behavior	Definition of: • Min/Max of Y axis • Number of tickets to draw on the axes • Background image
Text	Trend title and font properties (font size, label, etc.)
Grid	Properties of grid presentation (colors)
Cursor	Properties of cursor presentation (enable and color)
X Scale	Properties of X Scale presentation
Y Scale	Properties of Y Scale presentation
Curve "n"	Buffer that contains the tag's values to plotted in the trend widget.

Tag values can be scaled using the X Forms in the **Attach to** dialog. See ""Attach to" parameters" on page 39 for details.

Printing historical trend widget

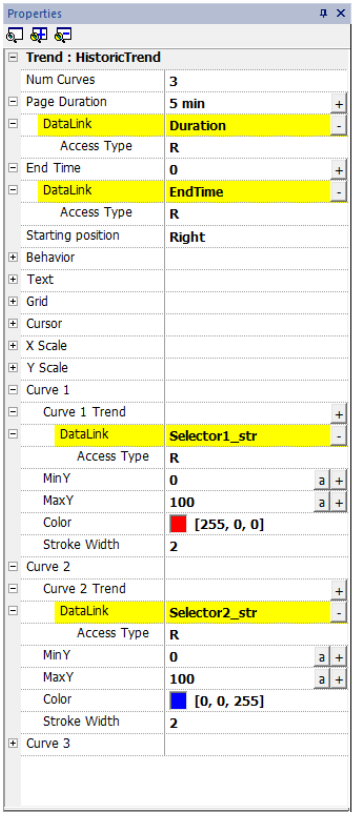
The historical trend widget can be found and used from the print report gallery.

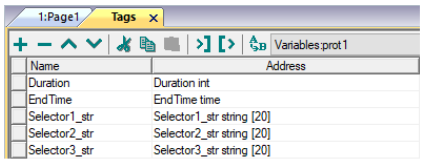
Using the “attach to tag” feature is possible to use tags to define some properties of the historical trend to print at runtime:

- Page Duration
- End Time
- Curve Name

"Page Duration" with "End Time" define the piece of the trend buffer to print. "Curve Name" can be used to select the curve to show. An empty string means no curve to show.

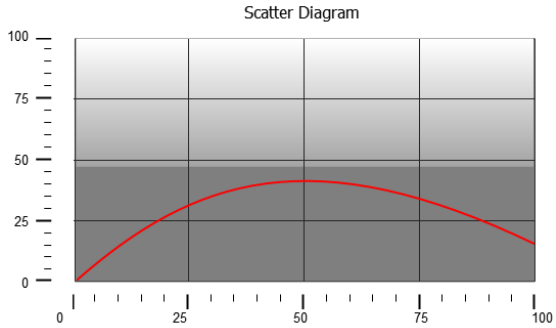
SetTrendView() and ScrollTrendToTime() are display macros and have no effect on report printing.





Scatter diagram widget

A scatter diagram is a type of diagram to display values for two variables from a set of data using Cartesian coordinates. The data is displayed as a collection of points, each having the value of one variable determining the position on the horizontal axis and the value of the other variable determining the position on the vertical axis. For this reason it is often called *XY graph*.



Scatter diagram curves are obtained by a linear interpolation of points. To create a new scatter diagram:

1. Add a **Scatter Diagram** widget to the page.
2. Select the number of curves to show: each curve is named as Graph1, Graph2,...
3. Customize the general graph properties such as **X Min**, **X Max**, **Grid** details.
4. Define the max number of samples/values for each curve by setting the **Max Samples** parameter.

   Here you set the max number of values to be displayed in the graph starting from first element in the array.

For example: Tag1[20] and Max Samples = 10 will show just first 10 elements of the Tag1 array.

5. Define for each curve the two tags of type array to be displayed (**X-Tag** and **Y-Tag**).

When the array tags change, you can force a refresh with the **RefreshTrend** action .

Note: Scatter diagrams support only the **RefreshTrend** action.

Printing scatter diagram widget

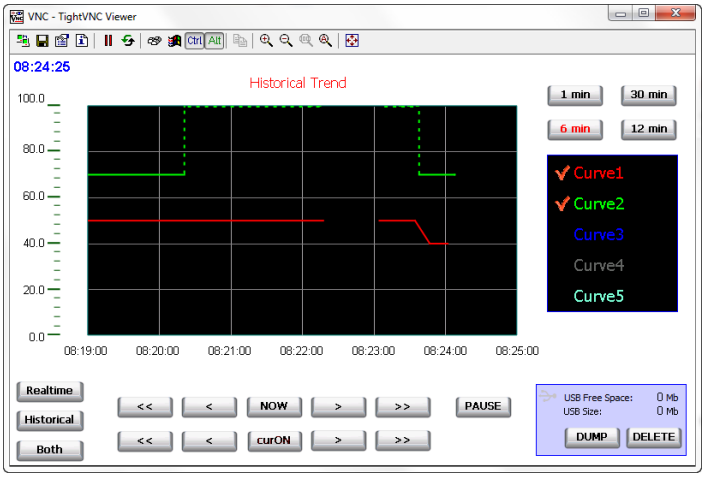
The scatter diagram widget can be found and used from the print report gallery. Note that using the attach to tag feature is possible to use tags to define some properties of the scatter diagram to print at runtime.

Trend widget tips

Be aware that some properties are only available when the Properties pane is in Advanced view.

Values outside range or invalid

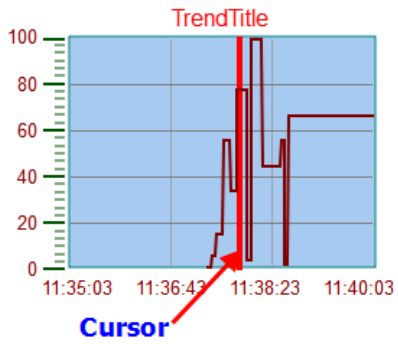
When trend value goes beyond the limits set for the trend widget, a dotted line is displayed. When the value of the tag is not available, for example the controller device is offline, no curve is drawn.



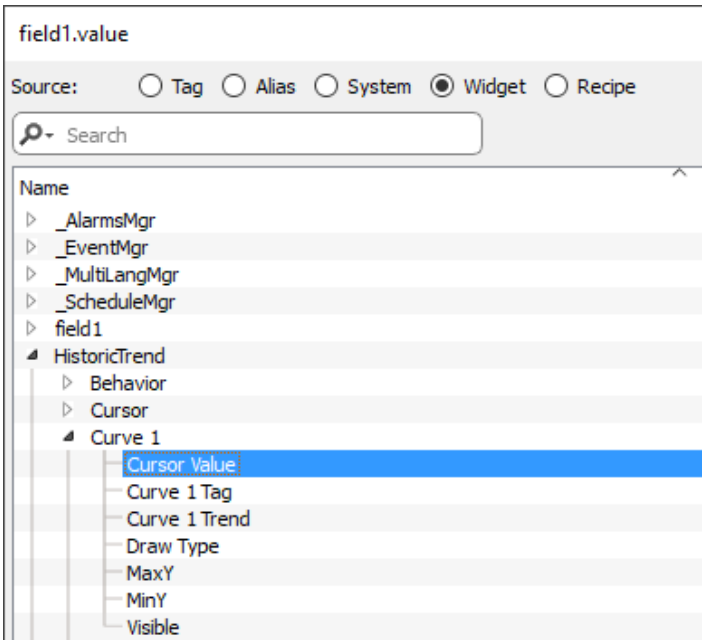
Showing trend values (cursor)

Trend cursor displays the trend value at a specific point.

Use the actions **ShowTrendCursor** and **ScrollTrendCursor** to enable the trend cursor and move it to the required point to get the value of the curve at that particular point in time.

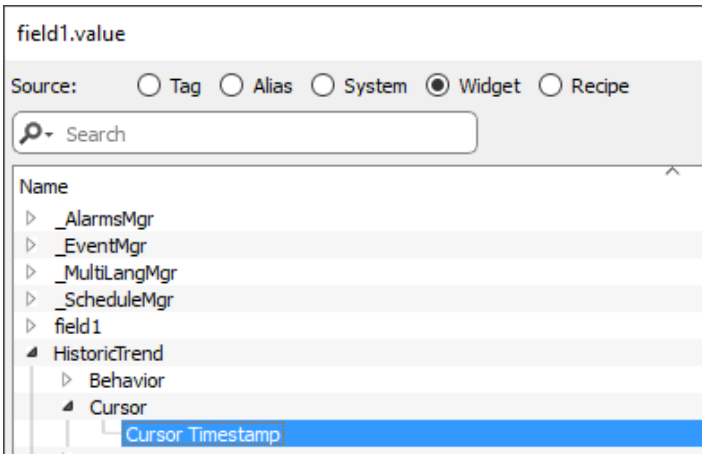


To display the value of the trend cursor on the page, define a numeric field and attach it to the **Cursor Value** widget tag.



In this example the Y axis value of the cursor is displayed.

To display the trend time stamp at the position of the cursor, use a Time/Data widget (available inside Basic->Controls category) and attach the widget's value to the **Cursor Timestamp** property of the trend widget.



Modify trend widget properties at runtime

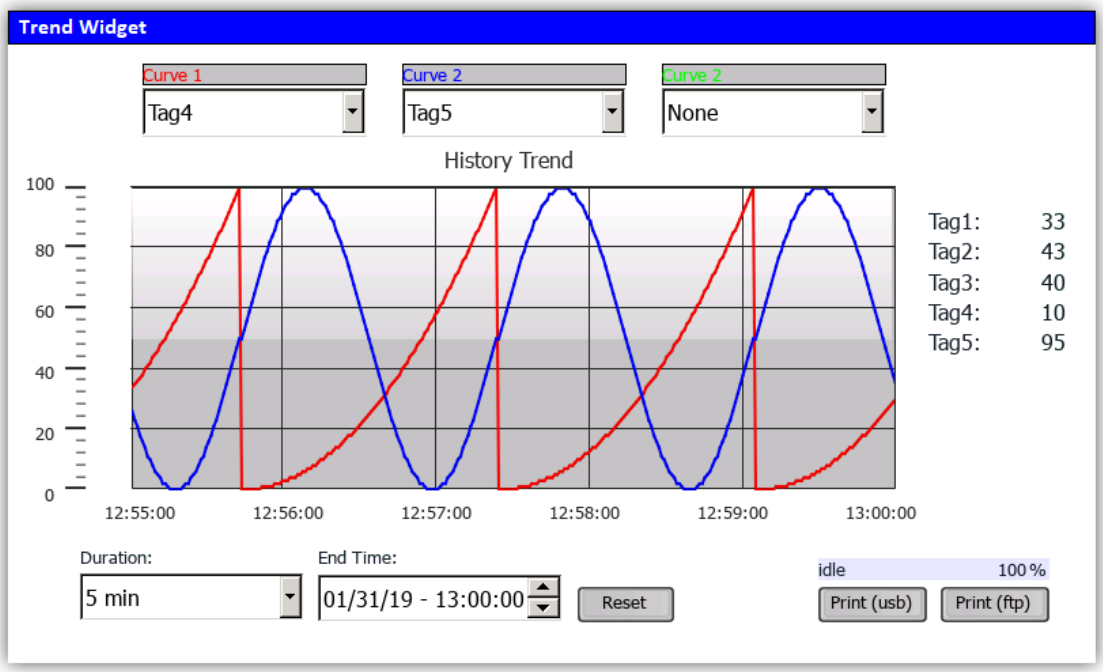
Using the attach to tag feature is possible to use tags to modifies some properties of the trend widgets at runtime.

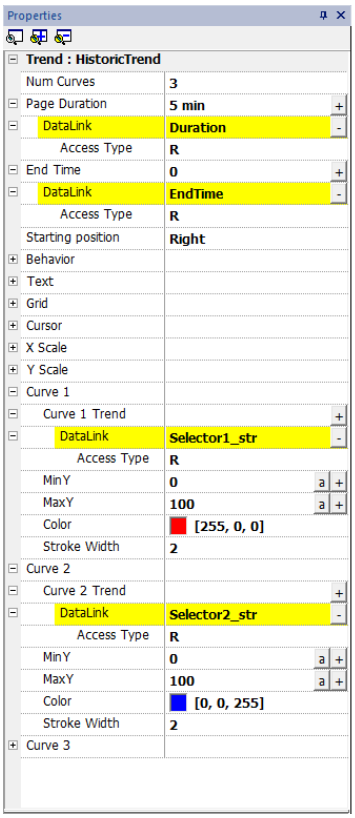
Example 1

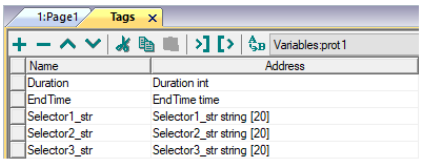
Using :

- Page Duration
- End Time
- Curve Name

is possible to modify from the runtime application the zoom factor, the window period and the curve to draw.

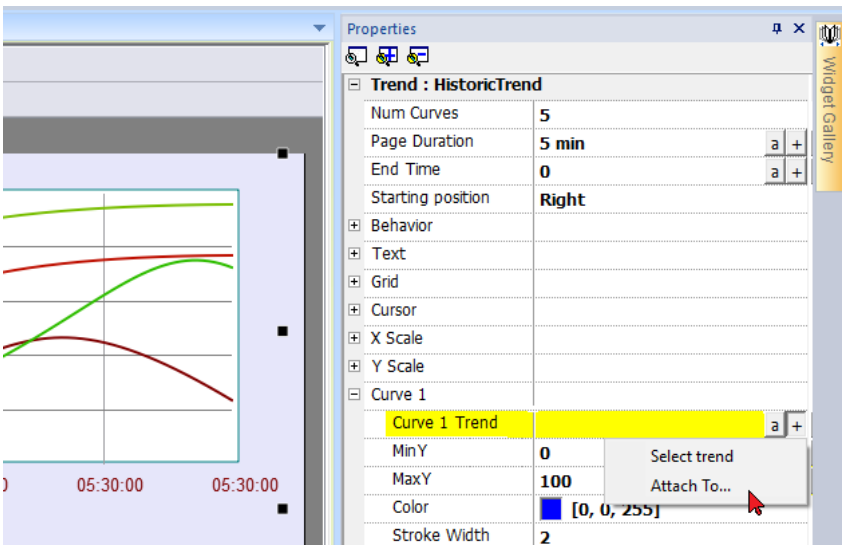


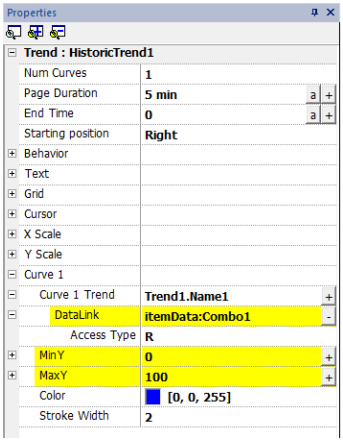


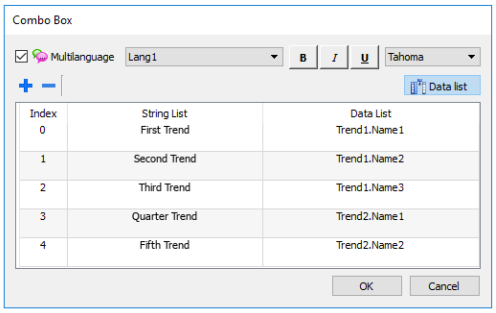


Example 2

Curve property can be attached to a Combo Box to select the curve to draw







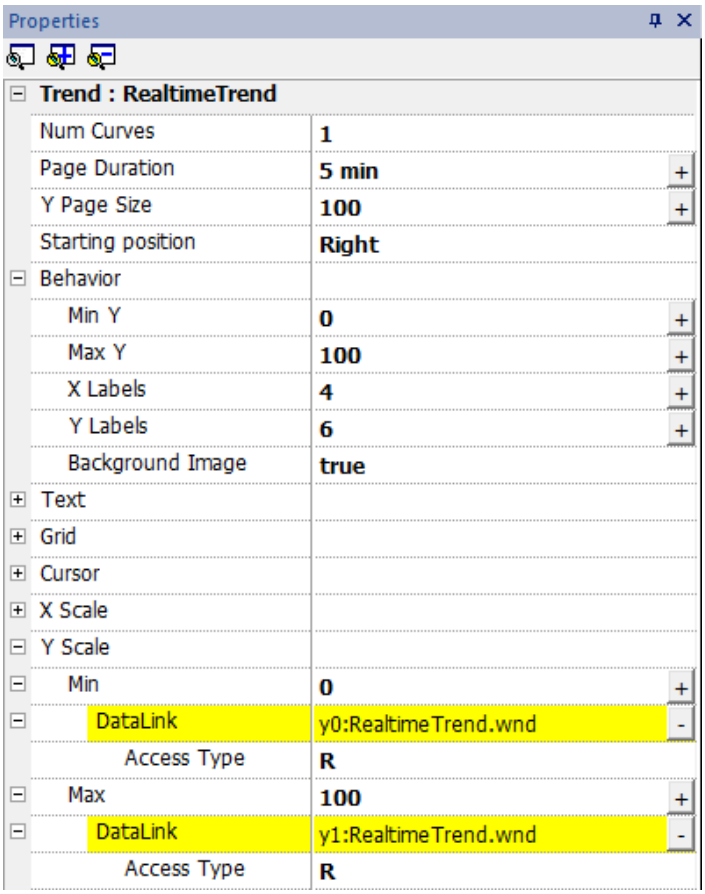
Trend widget gestures

Trend widgets support gesture commands:

Gesture	Description
pan	Touch the widget to scroll the curve within the widget area
pinch	Use two fingers to pinch the curve and perform zoom operations

WARNING: Only multi touch HMI devices can generate pinch events

Note: In order to support gestures on Y axis, Min/Max properties of the trend widget must be linked to Min/Max values of Behavior parameters (default for new trends).



Request Samples

Request Sample property can be set for each curve and indicates the maximum numbers of samples read by the widget at one time from the trend buffer.

Tip: You normally do not need to modify the default value. Adjust it to fine tune performances in the trend widget refresh, especially when working with remote clients.

Color bands

Use the color bands configuration to customize your graphs background, for example to make certain days or hours stand out (weekends, night hours, etc.).

1. In the **Properties** pane, in **Color Bands** property click **+**: the **Configure Bands** window appears.
2. Click **+** to add as many colors you need.
3. Select multiple cells and click on a color band to assign the color to the selected range of cells.

Note: This feature only uses local time in the trend widget, not the global time option.

Calendar color bands example

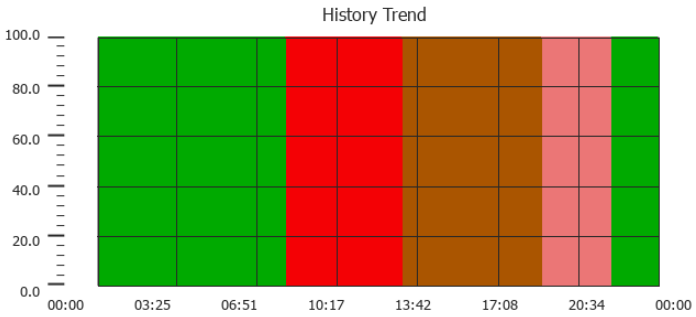
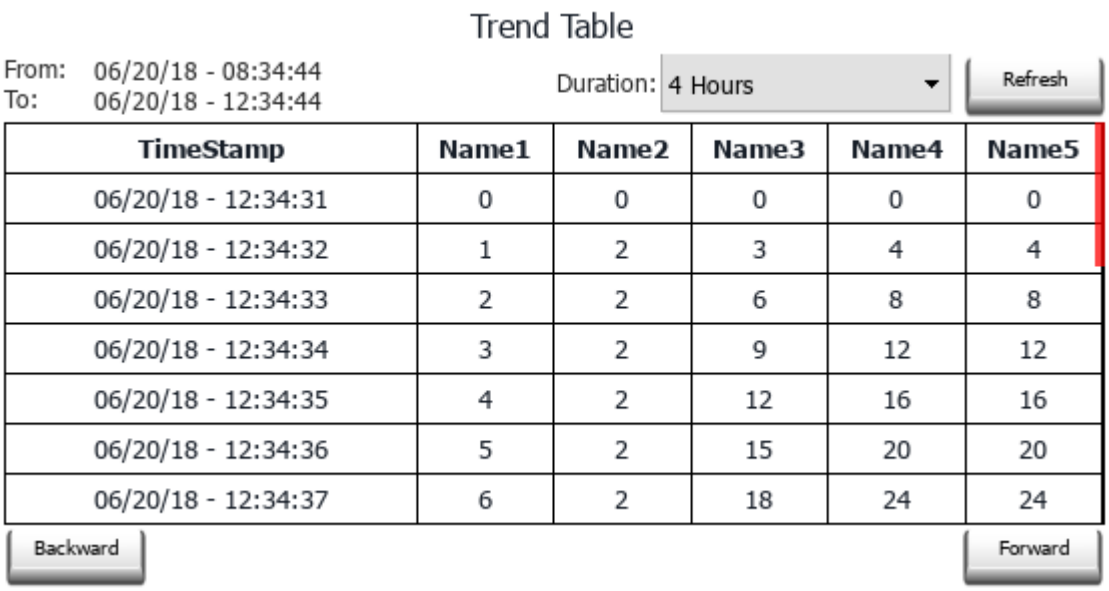


Table trend widget

Path: **Widget Gallery> Basic> Trends/Graphs**

Display contents of a trend buffer inside a widget



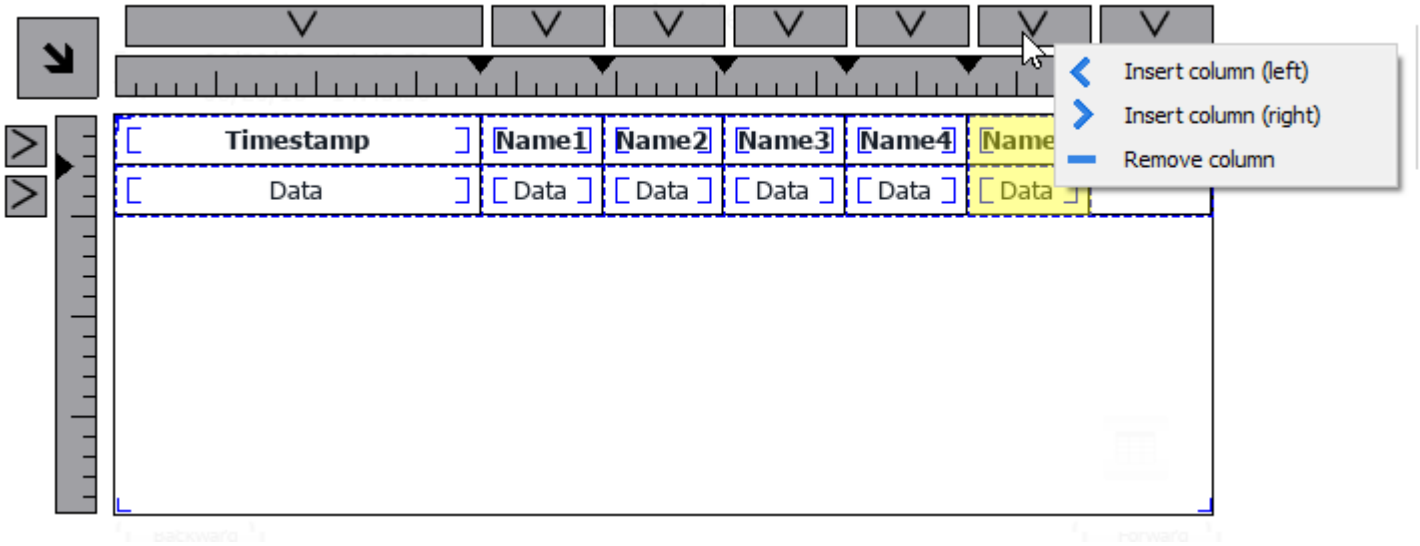
Buttons:

- REFRESH
  Retrieve trend data from internal buffer and refresh table view
- BACKWARD/FORWARD
  Move the display window forward or backward as specified in the duration parameter

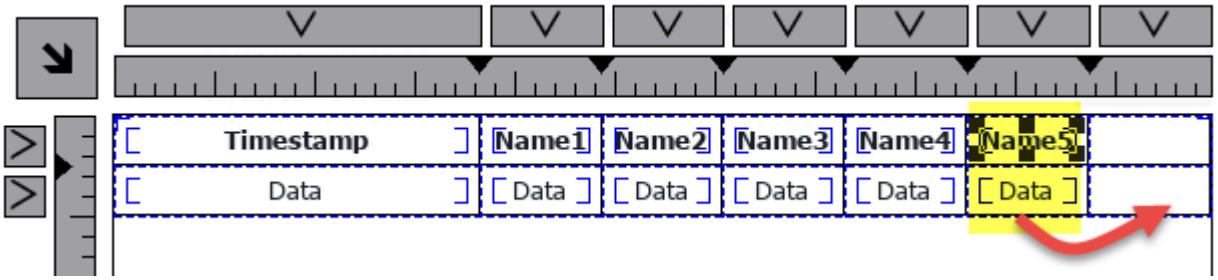
Parameter	Description
TrendName	Trend Buffer from which the samples are retrieved (see "Data logging" on page 260)
Heading	Heading labels The visible labels inside the LRH SWeditor are only placeholders, the actual labels that will be displayed are defined in the trend configuration (see "Data logging" on page 260)
Page Duration	Time window to show
End Time	End time of the time window This parameter can be used to scroll the time window. When zero, the end time is the current system time.
Time Spec	Time format: • **Local** = show the time values of the HMI device. • **Global** = show the time values using UTC format.
Date Format	Select the Date and Time format
Table Layout	Defines the characteristics of the scroll bar and allows to remove the header of the table

Adding or removing trend columns

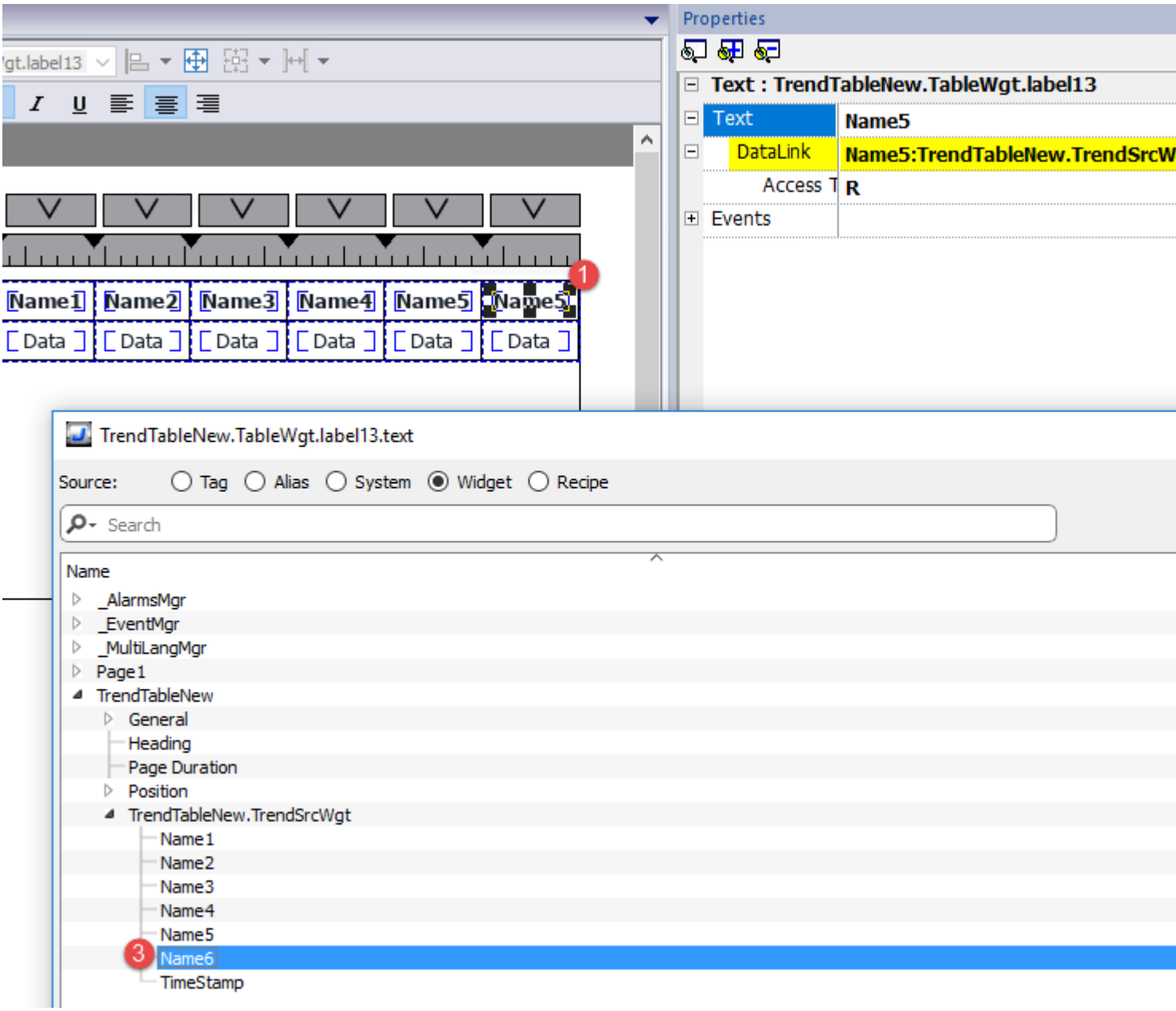
To add or remove a column, double-click on the grid to enter edit mode and right-click on the column selector to open the context menu from where to insert or remove a column.

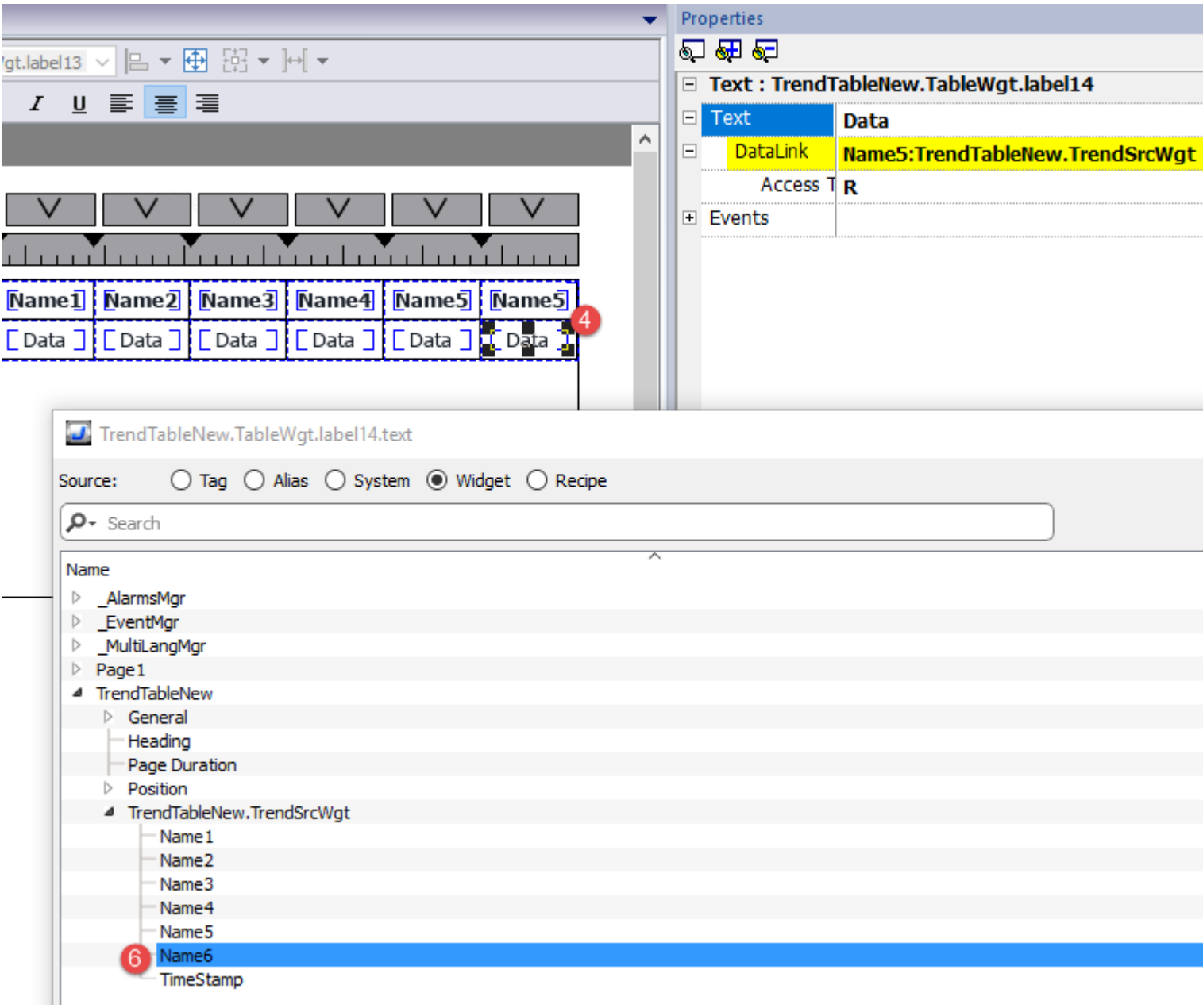


Copy and past fields from another column



Then use the properties panel to select the trend element to add to the new columns


Properties
Text : TrendTableNew.TableWgt.label13
Text
Name5
DataLink
Name5:TrendTableNew.TrendSrcW
Access T
R
Events
Name1
Name2
Name3
Name4
Name5
Name5
Data
Data
Data
Data
Data
Data
1
TrendTableNew.TableWgt.label13.text
Source:
Tag
Alias
System
Widget
Recipe
Search
Name
_AlarmsMgr
_EventMgr
_MultiLangMgr
Page1
TrendTableNew
General
Heading
Page Duration
Position
TrendTableNew.TrendSrcWgt
Name1
Name2
Name3
Name4
Name5
Name6
TimeStamp
3



Printing trend table

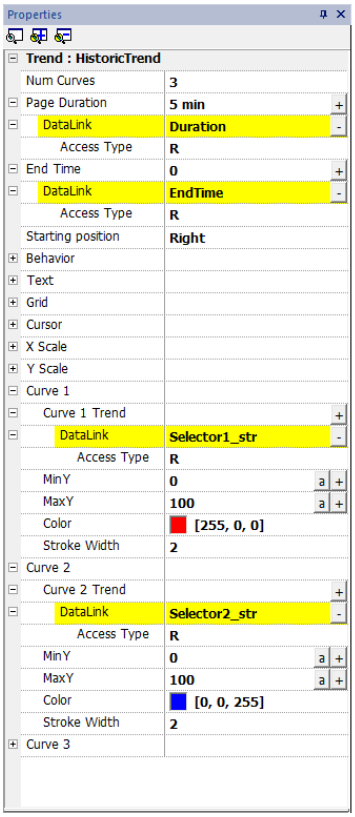
A trend table widget without buttons can be found and used from the print report gallery. The table can be drawn and enlarged to fill the entire page. If the number of lines to printed is greater of one page, the trend table will be printed using additional pages. See the "Table of functions and limits" on page 540 for the max number of printable rows.

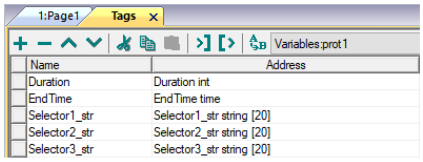
Using the “attach to tag” feature is possible to use tags to define some properties of the historical trend to print at runtime:

- Page Duration
- End Time
- Curve Name

"Page Duration" with "End Time" define the piece of the trend buffer to print. "Curve Name" can be used to select the curve to show. An empty string means no curve to show.

SetTrendView() and ScrollTrendToTime() are display macros and have no effect on report printing.





22 Data transfer

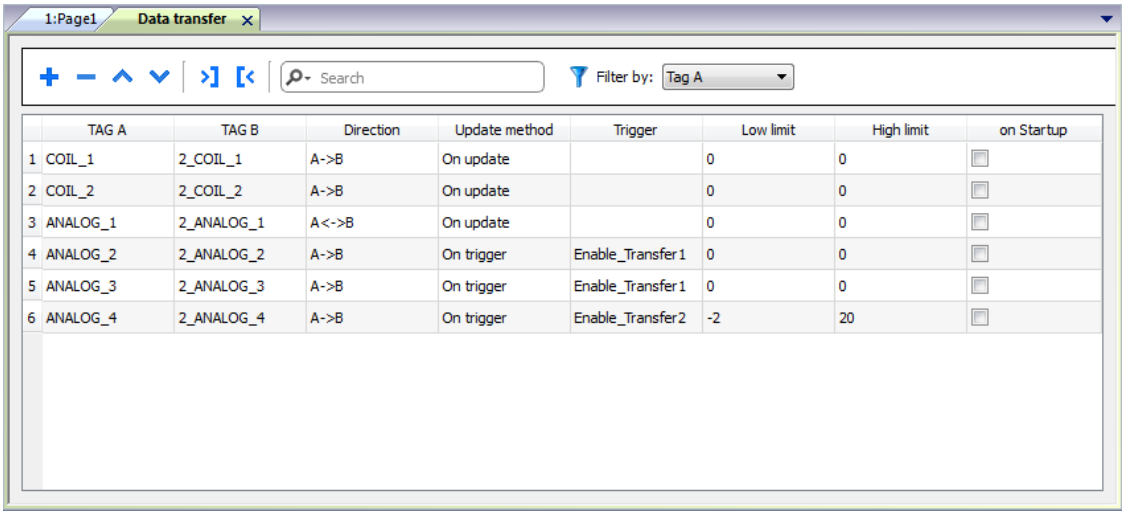
Data transfer allows you transferring variable data from one device to another. Using this feature an HMI device can operate as a gateway between two devices, even if they do not use the same communication protocol.

Data transfer editor

Path: ***ProjectView*** *>* ***Config*** *> double-click* ***Data transfer***

Use the Data transfer editor to map transfer rules.

Each line in the Data transfer editor defines a mapping rule between two tags. Define more mapping rules if you need different direction, update method or trigger.



	TAG A	TAG B	Direction	Update method	Trigger	Low limit	High limit	on Startup
1	COIL_1	2_COIL_1	A->B	On update		0	0	
2	COIL_2	2_COIL_2	A->B	On update		0	0	
3	ANALOG_1	2_ANALOG_1	A<->B	On update		0	0	
4	ANALOG_2	2_ANALOG_2	A->B	On trigger	Enable_Transfer1	0	0	
5	ANALOG_3	2_ANALOG_3	A->B	On trigger	Enable_Transfer1	0	0	
6	ANALOG_4	2_ANALOG_4	A->B	On trigger	Enable_Transfer2	-2	20	

To add a new rule, click **+**: a new tag line is added.

Data transfer toolbar

Prameter	Description
Import/ Export	Imports or exports data transfer settings from or to a .csv file.
Search	Displays only rows containing the search keyword.
Filter by	Display only rows matching filter and search field.

Data transfer parameters

Prameter	Description
TAG A/ TAG B	Pair of tags to be mapped for exchanging through the HMI device.
Direction	Transfer direction. **A->B** and **B->A**: Unidirectional transfers, values are always copied from one tag and sent to the other tag in the specified direction. **A<->B**: Bidirectional transfer, values are transferred to and from both tags.
Update Method	**On trigger**: Data transfer occurs when the value of the tag set as trigger changes above or below the values set as boundaries. Limits are recalculated on the previous tag value, the same that triggered the update.

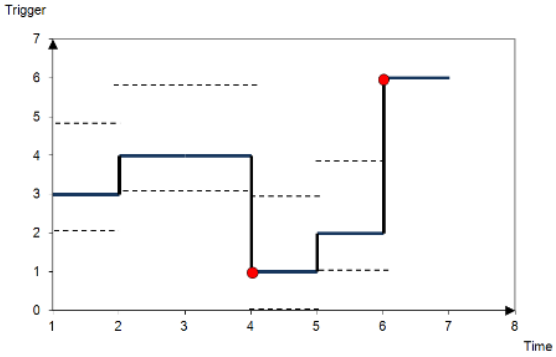
Prameter	Description
	Note: This method applies only to unidirectional transfers (A->B or B->A). **On Update**: Data transfer occurs whenever the value of the source tag changes. Note: This method applies both to unidirectional and to bidirectional transfers (A->B, B->A and A<->B). Note: The Runtime cyclically monitors source tags changes (trigger tag when using On Trigger or tags to transfer when using On Update) based on Tag editor **Rate** parameter. If **Rate** setting for source Tag is 500 ms (default), the system checks for updates every 500 ms. Note: Changes on source tags faster than **Rate** may be not detected .
Trigger, High limit, Low limit	Tag that triggers the data transfer process. When this tag changes its value outside the boundaries set as **High limit** and **Low limit**, data transfer is started. The range of tolerance is recalculated according to the specified limits on the tag value which triggered the previous update. No action is taken if the change falls within the limits. This mechanism allows triggering data transfers only when significant variations of the reference values occur. **Low limit** is less or equal to zero. Note: If both **Low limit** and **High limit** are set to "0", data transfer occurs whenever the value of the trigger tag changes.
on Startup	When selected, data transfer is forced: • on HMI startup if the quality of the source tag is good • after communication errors, when the associate device nodes return active See "Objects" on page 479 for details on quality. **Important: Data transfers executed on startup may have major impact on the HMI device boot time. Enable this option only when necessary.**

Example of limit setting

High limit = 1,9

Low limit = - 0,9

• = points where the data transfer is triggered



Exporting data to .csv files

Configuration information for data transfers can be exported to a .csv file.

Example of data transfer settings in .csv file

A	B	C	D	E	F	G	H	I	J
COIL_1	2_COIL_1	A->B	On update		0	0	data1	true	1
COIL_2	2_COIL_2	A->B	On update		0	0	data2	true	1
ANALOG_1	2_ANALOG_1	A<->B	On update		0	0	data3	true	1
ANALOG_2	2_ANALOG_2	A->B	On trigger	Enable_Transfer1	0	0	data4	true	1
ANALOG_3	2_ANALOG_3	B->A	On trigger	Enable_Transfer1	0	0	data5	true	1
ANALOG_4	2_ANALOG_4	A->B	On trigger	Enable_Transfer2	-10	20	data6	true	1

Column	Description
A to G	Same data as in the Data transfer editor
H	Unique identifier automatically associated to each line. **Important: When you edit the .csv file and you add any extra line, make sure you enter a unique identifier in this column.**
I and J	Reserved for future use.

Import/export use the separator character defined inside Windows Regional Settings.

Data transfer limitations and suggestions

Correct definition of data transfer rules is critical for the good performance of the HMI devices. To guarantee reliability of operation and performance, keep in mind the following rules.

On trigger method

The **On trigger** method allows only unidirectional transfers, (A->B or B->A)

Data transfer based on the **On Trigger** mode should be preferred since it allows you to force the transfer and monitors only the trigger tags and not all the tags involved in the transfer.

On update method

The **On update** method allows changing the values in accordance with the direction settings only when the source value changes.

Using the **On Update** method you force the system to continuously read all the defined source tags to check if there are changes that need to be transferred. The default value of the update rate of each tag is 500 ms and can be modified with Tag editor.

Performance observations

Data transfer performance depends on:

- number of data transfers defined,
- number of data transfers eventually occurring at the same time,
- frequency of the changes of the PLC variables that are monitored,

Important: Always test performance of operation during project development.

Important: If inappropriately set, data transfer tasks can lead to conditions where the tags involved create loops. Identify and avoid such conditions.

Tip: Use the scheduler to calibrate the update rate based on the performance of your entire project.

Tip: Use array type tags to optimize data transfer and reduce workload.

Tip: Reduce the number of data transfers to reduce page change time and boot time.

23 Offline node management

When one of the controllers communicating with the HMI device goes offline, communication performance of the system may eventually decrease.

The offline node management feature recognizes offline controllers and removes them from communication until they come back online.

Additionally, if you know that any of the controllers included in the installation is going to be offline for a certain time, you can manually disable it to maximize system performance.

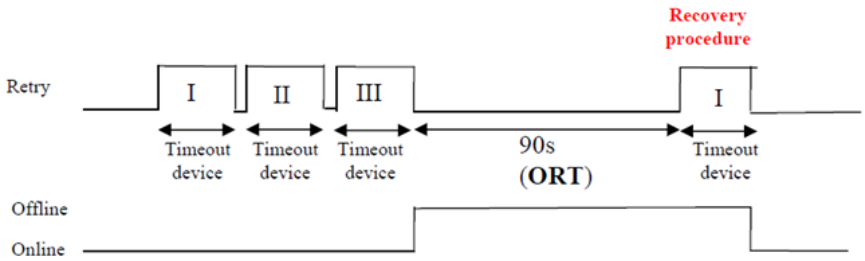
Note: This feature is not supported by all communication protocols. Check protocol documentation to know if it is supported or not.

Offline node management process

Steps of the process are:

- The system communicates normally with a certain device. When the device is not responding to a communication request, the system will repeat the request twice before declaring the device offline.
- When a device is offline, the system sends communication requests to the device with a longer interval, called Offline Retry Timeout. If the device answers to one of these requests, the system declares it online and restarts normal communication.

The diagram shows the three communication attempts and the recovery procedure that starts when the Offline Retry Timeout is elapsed.



Manual offline node management process

Offline node management can be done manually. When a specific device is online and it is communicating normally you can:

- use an action to declare the device offline: the system stops communication with the device.
- use an action to declare the device online: the system restarts normal communication with the device.

Manual offline configuration

When you know that some devices in communication with the HMI device are going to remain offline for a certain period of time, you can exclude them from communication using the **EnableNode** action.

WARNING: All disabled device nodes will remain disabled if the same project is downloaded on the device, on the other hand, if a different project is downloaded, all disabled devices will be re-enabled. The same happens with a package update.

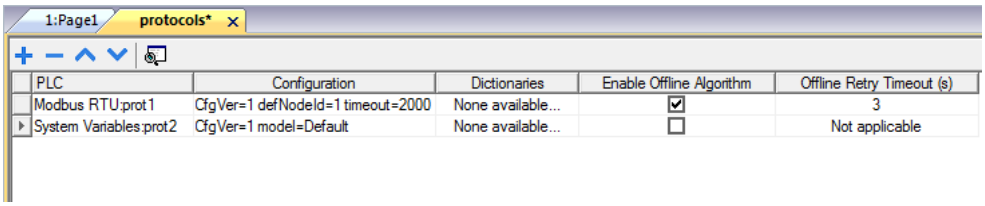
Tip: To make this feature more dynamic, you may decide not to indicate a specific **NodeID** but attach it to the value of a tag or to an internal variable created to identify different devices that might be installed in your network.

Note: When using the action **EnableNode** to force a device node back online, communication will start immediately.

Automatic offline node detection

When a device is not answering to communication requests, it is de-activated. The HMI device stops sending requests to this device. After three seconds, the HMI device sends a single command to check if device is available, if so the communication is restarted, otherwise it is disabled for another timeout interval.

Default settings can be modified in Protocol editor.



Note: Not all protocols support this feature.

Parameter	Description
Enable Offline Algorithm	Enables offline management for the protocol
Offline Retry Timeout	Interval in seconds for the retry cycle after a device has been deactivated. Range: 1–86.400 seconds (24h).

24 Multi-language

Multi-language feature has been designed for creating HMI applications that include texts in more than one language at the same time

Multi-language feature uses code pages support to handle the different languages. A code page (or a script file) is a collection of letter shapes used inside each language.

Multi-language feature can be used to define languages and character sets in a project. LRH SW also extends the TrueType Fonts provided by Windows systems to provide different font faces associated with different character sets.

LRH SW also allows you to provide strings for each of the languages supported.

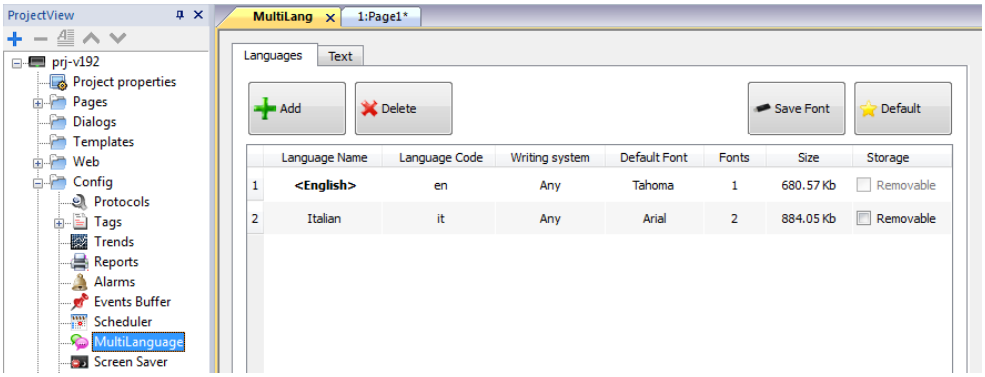
LRH SW also allows you to change the display language so that you can see the page look and feel during the design phase.

Appropriate fonts may need to be installed to manage the different languages. When adding font files, be aware that there may be license rights that need to be acquired in order to use them.

On the Internet, is easy to find several fonts provided with the open-source license as, e.g., the Noto family offered by Google (www.google.com/get/noto)

The Multi-language editor

Path: ***ProjectView*** *>* ***Config*** *> double-click* ***MultiLanguage***



Language settings

Parameter	Description
Language Name	Name identifying the language in the project.
Language Code	ISO 639 language code identifier, used to match language items when importing resources from external xml files.
Writing system	Select the set of fonts to be used with the language
Default Font	Default font for project's widgets. Note: When you choose a new font you are prompted to replace the font used in the widgets you already created.
Fonts	Number of fonts associated with the selected language.
Size	Memory used to store font files.
Storage	Location of file fonts is a removable external memory. Tip: Store large font files on removable memory to free memory requirements in the HMI device.

Adding a language

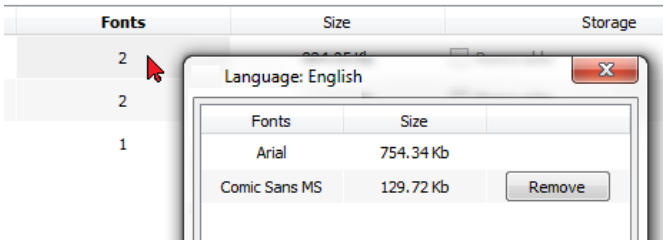
1. In the **Languages** tab, click **+**: a line is added to the table.
2. Enter all language settings.
3. Click **Default** to set the selected language as the default language when the Runtime starts.
4. Click **Save Font** to copy the fonts you marked as **Removable** on an external memory.

Important: Font files configured to be stored on removable memory must be provided to the final user to complete font installation on the HMI device.

Removing fonts

To remove fonts no longer needed:

1. Click on the font number in the Multi-language editor: a dialog with the list of the used fonts is displayed.

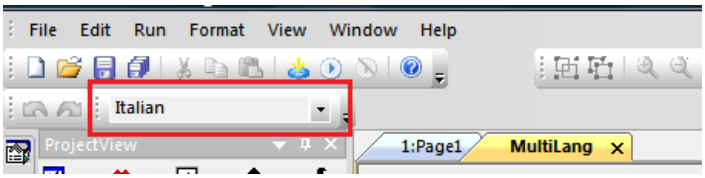


2. Select the fonts to be removed and click **Remove**: removed fonts are replaced with the default font.

Changing language

Changing language during page design

A combo box is available for changing language during page design. If no texts appears, please check **Text** tab in the Multilanguage editor and insert missing string.

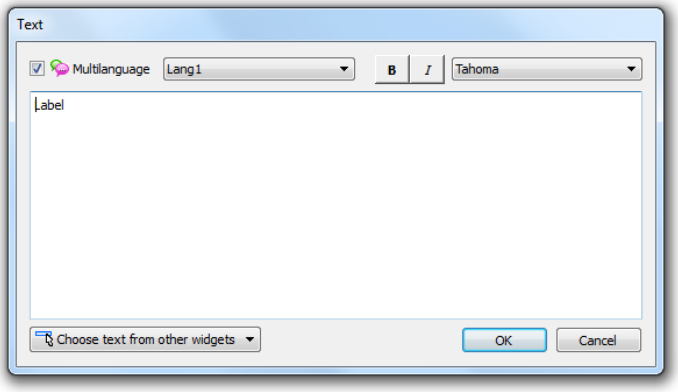


Multi-language widgets

Multi-language support is available for objects such as buttons, static text, messages, alarm descriptions and pop-up messages.

Multi-language for label widgets

Double-click on a text widget in a page to open the **Text** dialog.



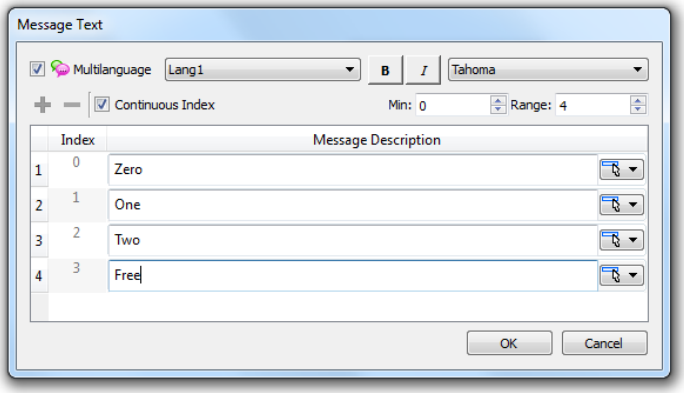
Enable/disable multi-language function, edit the text for the selected language and choose the font.

Note: Bold, italic and color properties set here for the widget are applied to all languages .

Parameter	Description
Multilanguage	Enable/disable multi-language function for the widget.
Choose text from other widget	Click on button to browse existing message strings in project to pick text for the widget.

Multi-language for message widgets

Double-click on a message widget in a page to open the **Message Text** dialog.

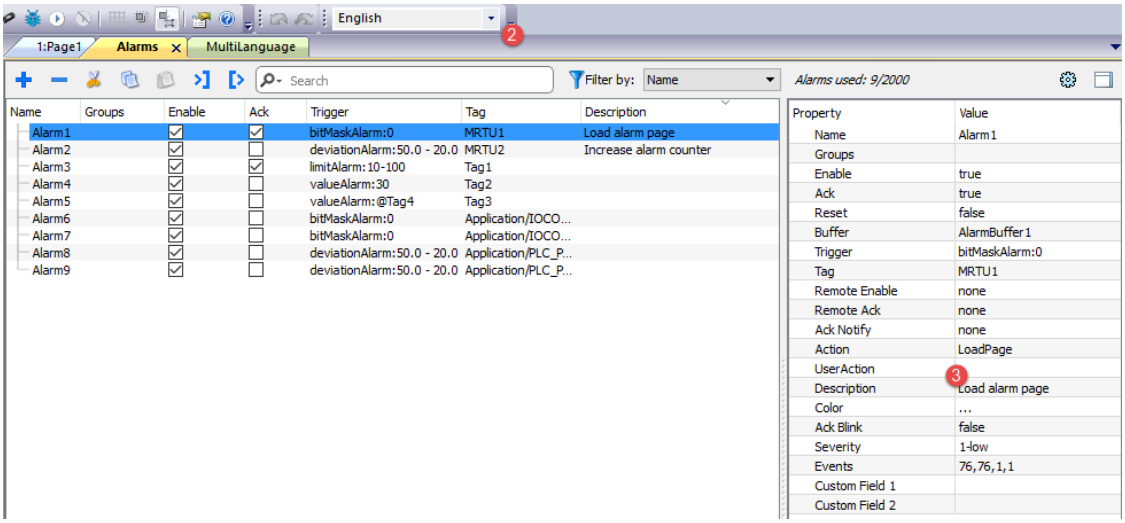


Parameter	Description
Multilanguage	Enable/disable multi-language function for the widget.
Continuous Index	Index for the widget is set of contiguous numbers (example 3, 4,5,6)
Min	Starting number for index
Range	Number of messages
Choose text from other widget	Click on button to browse existing message strings in project to pick text for the widget.

Multi-language for alarm messages

To add a multi-language strings for alarm messages:

1. Open the Alarm editor.
2. Select a language using the language combo box.
3. Enter the text for the alarm in the **Description** column.

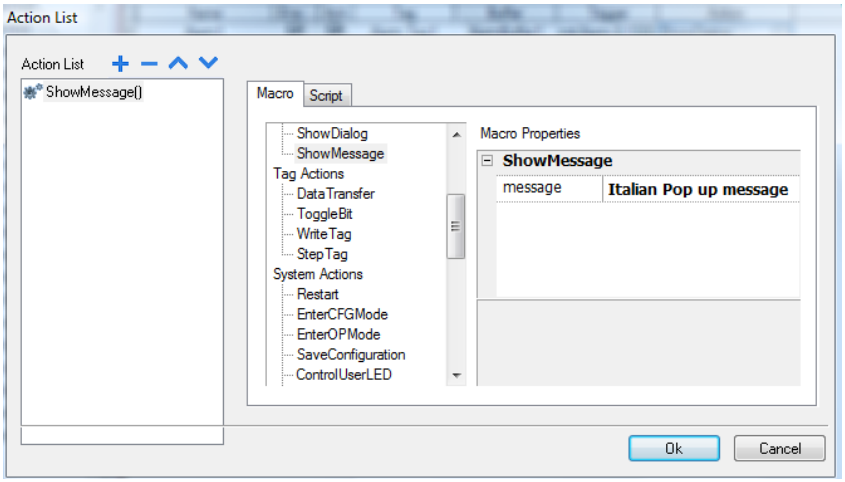


Tip: Text labels with alarm states displayed by alarms widgets can be translated or personalized through the Multilanguage text editor.

Multi-Language for pop-up messages

To add a multi-language pop-up message:

1. Select a language from the language combo box.
2. Add the Page action **ShowMessage** and enter the text in the selected language.



Exporting/importing multi-language strings

The easiest way to translate a project into multiple languages is to export all texts to a .csv file, translate the resulting document and then import the translated text back into the project.

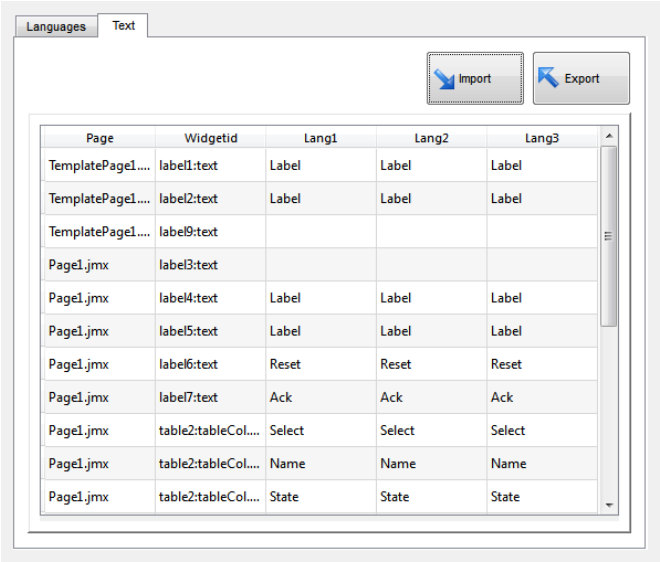
Important: The .csv file exported by LRH SW is coded in Unicode, to edit it you need a specific tool supporting Unicode encoded .csv files.

Exporting and reimporting strings

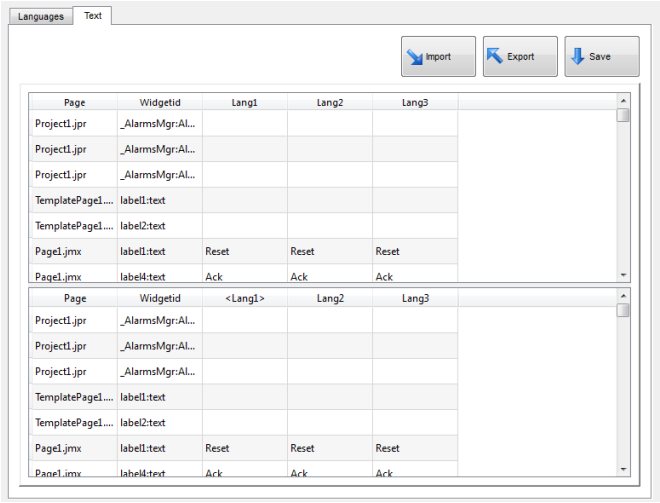
Path: ***ProjectView*** *>* ***Config*** *>* *double-click* ***MultiLanguage***

To export and re-import multi-language strings:

1. In the **Text** tab, click **Export**: all multi-language strings are exported to a .csv file.



Important: Set all languages that will be used in the project before exporting the file. This will guarantee that the exported file will contain all columns and language definitions.



2. Once the strings have been translated, click **Import** to re-import them into the project: strings are imported matching the widget ID and the page number of each widget.
3. Click **Save** to save the new widget data.

Note: To change the separator used in the exported file, change the regional settings of your computer. When importing, the separator information is retrieved from the file; if not found, the default character "," is used.

Import constraints

The following formats are supported for import:

- Comma Separated Values (.csv)
- Unicode Text (.txt)

Note: Use the Unicode Text file format when you import a file modified using Microsoft® Excel®.

Changing language at runtime

Changing language with an action

After the project download, the HMI Runtime will start using the language set as default. You can change the language using the **SetLanguage** action. See "MultiLanguage actions" on page 174.

Note: Once the language has been changed, it will be used also in future sessions.

The active language code is available from JavaScript API. See "curLangCode" on page 497 for additional details.

Missing fonts

When you change language, if the required fonts are not available in the device memory, a pop-up message prompts you to insert the memory card containing the missing fonts. At the end of the operation you can remove the memory card.

Limitations in Unicode support

LRH SW has been designed for working with Unicode text. However, for compatibility issues with some platforms, Unicode is supported only in a subset of properties.

Area	Property	Charset Accepted	Reserved Chars/Strings
Protocol editor	Alias	ASCII [32..126]	(space) , ; : . < * >'
Tag editor	Name	ASCII [32..126]	. \ / * ? : > < | " & # %;=
	Group	ASCII [32..126]	<New> \ / * ? : > < | " & # % ;
	Comment	Unicode	
Trends	Name	ASCII [32..126]	\ / * ? : > < | " & # % ;
Printing Reports	Name	ASCII [32..126]	\ / * ? : > < | " & # % ;
Alarms	Name	ASCII [36..126]	\ / * ? : > < | " & # % ;
	Description	Unicode	[] - for live tags, \ escape seq for [and \
Events	Buffer Name	ASCII [32..126]	\ / * ? : > < | " & # % ;
Scheduler	Name	ASCII [32..126]	\ / * ? : > < | " & # % ;
Languages	Language Name	ASCII [32..126]	\ / * ? : > < | " & # % ;
	Texts in widgets	Unicode	-
	Texts from import files	Unicode	-
User Group	Group Name	a-z A-Z _	admin,guest,unauthorized
	Comments	Unicode	-
User	Name	ASCII [32..126]	\ / * ? : > < | " & # % ;
	Password	Unicode	-
	Comment	Unicode	-
Recipes	Name	ASCII [32..126]	\ / * ? : . > < | " & % ; ,
	Set Name	ASCII [32..126]	\ / * ? : . > < | " & % ; ,
	Element name	ASCII [32..126]	\ / * ? : . > < | " & % ; ,
General	Project Name	A-Z,a-z,0-9,-,_	“PUBLIC”, “readme”, “index.html”
	Page Name	A-Z,a-z,0-9,-,_	-
	Dialog Page Name	A-Z,a-z,0-9,-,_	-
	Template Page Name	A-Z,a-z,0-9,-,_	-

Area	Property	Charset Accepted	Reserved Chars/Strings
	Keypad Name	A-Z,a-z,0-9,-,_	-
	Files (Images/Video/etc..)	A-Z,a-z,0-9,-,_	-
	Widgets ID	A-Z,a-z,0-9,-,_	-
Runtime	PLC Communication	UTF-8, Latin1, UCS-2BE, UCS-2LE, UTF-16BE, UTF-16LE	-

25 Scheduler

LRH SW provides a scheduler engine that can execute specific actions at set intervals, or on a time basis.

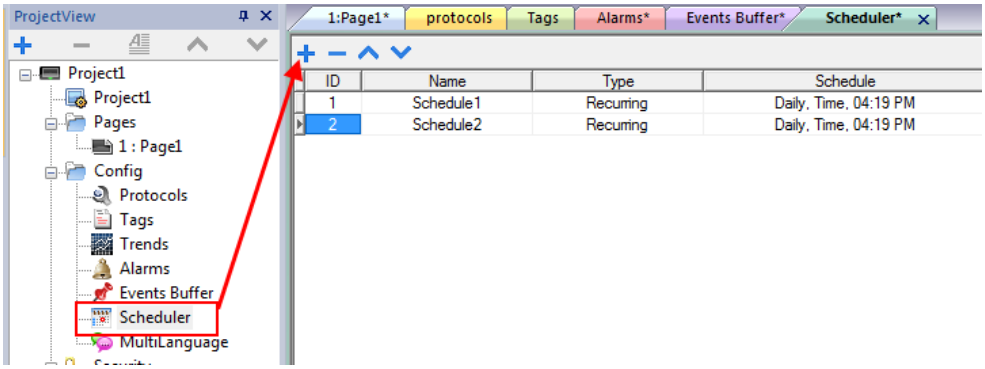
Creating a schedule is typically a two-step process:

1. You create a schedule with a list of actions to be executed when the scheduled event occurs. You do this in the Scheduler editor
2. You create a runtime user interface that allows the end-user to change settings for each schedule. You do this adding a **Scheduler** widget to a page of your project and configuring it to fit user scheduling needs.

Creating a schedule

Path: ***ProjectView*** *>* ***Config*** *> double-click* ***Scheduler***

- Click **+** to add a schedule.



Schedule parameters

Parameter	Description
ID	Unique code assigned automatically to the schedule
Name	Name of schedule
Type	Type of schedule: • **Recurring**, see "Recurring schedule" on the facing page for details. • **HighResolution**, see "HighResolution schedule" on the facing page for details
Schedule	Scheduler settings and options. See "Recurring schedule" on the facing page for details.
Action	Actions to be executed at the scheduled time The macros added in the action field are executed on the server-side with the exception of the below macros that will be executed even on client-side (e.g. LRH SW). • loadPage • prevPage • nextPage • showDialog • showMessage • setLanguage • jsAction
Priority	Priority level for the event. If two schedules occur at the same time, the event with the higher priority will be executed first.

HighResolution schedule

The **HighResolution** schedule is used to perform actions that need to be repeated at specified intervals. The interval between executions is set in milliseconds in the **Schedule** column.

Note: You cannot change at runtime the settings of this type of schedule. If you need to change the action time settings at runtime, choose **Recurring** schedule and set **Type** to **Every**. See "Recurring schedule" below for details.

Recurring schedule

The Recurring schedule is used to perform actions at specified points in time. Settings can be modified at runtime.

Recurring scheduler parameters

Parameter	Description
Type	Frequency of the scheduled actions
Mode	Specific settings required by each scheduler type
Condition	Boolean tag (true/false) to activate the specified actions at the moment the timer is triggered. Actions will be executed if tag = true. By default, actions are executed when the timer is triggered. Note: Only tags attached to the Boolean data type are shown.
Actions	Actions to be executed by the schedule. **Important: Actions and schedule parameters cannot be modified at runtime**
Date	Date when the scheduled actions will be executed
Time/Offset	This field display one of the following: **Time** = when the scheduled actions will be executed **Offset**= delay or advance with respect to the selected mode.
Location	Reference location to calculate sunset/sunrise time.
weekdays	Days of the week in which the scheduled actions will be executed.
On startup	Executes schedule at start up
Enable schedule	Enables/disables the schedule
Execute only at startup	Executes the schedule only once at start up

Schedule type options

Option	Description
By Date	Actions are executed on the specified date and time.
Daily	Actions are executed daily at the specified time.
Every	Actions are executed with the specified interval (Range: 1 s–1 day)
Hourly	Actions are executed every hour at the specified minute.
Monthly	Actions are executed every month at the specified date and time.
Weekly	Actions are executed every week on the specified weekday(s) and time.
Yearly	Actions are executed every year at the specified date and time.

Schedule mode options

Option	Description
Time	Depends on the schedule type. Allows you to specify date/time/week data.
Random10	Actions are executed in the time interval of 10 minutes before or after the set time. For example, if set time is 10:30, actions are executed any time between 10:20 and 10:40.
Random20	Actions are executed in the time interval of 20 minutes before or after the set time. For example, if set time is 10:30, actions are executed any time between 10:10 and 10:50.
Sunrise+	Actions are executed with a specified delay after sunrise. The delay is set in minutes/hours and sunrise time is location specific.
Sunrise-	Actions are executed with a specified advance before sunrise. The advance is set in minutes/hours and sunrise time is location specific.
Sunset+	Actions are executed with a specified delay after sunset. The delay is set in minutes/hours and sunset time is location specific.
Sunset-	Actions are executed with a specified advance before sunset. The advance is set in minutes/hours and sunset time is location specific.

See "Configuring location for schedules" below for details on sunset and sunrise settings.

Note: **Mode** options are not available for all schedule types.

Configuring location for schedules

Scheduled actions can be configured to be executed at a specific time with respect to sunrise and/or sunset. To do this you need to define the correct location, based on UTC information. The system will the automatically calculate the sunrise and sunset time.

Only a few locations are available by default. If your location is not listed, you can add it by entering latitude, longitude and UTC information in the Target_Location.xml file.

Important: Each platform has its own Target_Location.xml file.

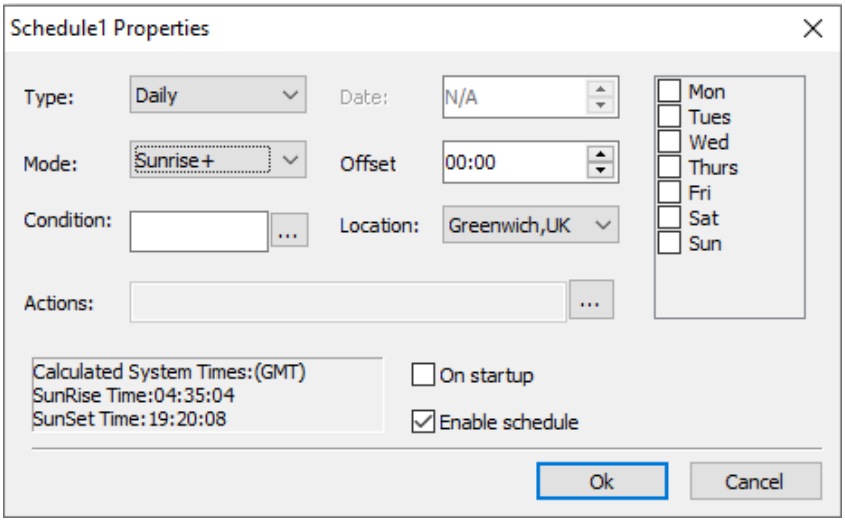
Location files position

Application	Location file path
LRH SW	*Lovato Electric\LRH SW\languages\shared\studio\config\Target_Location.xml*
HMI Devices	*Lovato Electric\LRH SW\runtime\<HW Platform>\config\Target_Location.xml*
Simulator	*Lovato Electric\LRH SW\simulator\config\Target_Location.xml*

For example, the information for Greenwich (UK) is shown below:

```
<file city="Greenwich,UK" latitude="51.47" longitude="0" utc="0"/>
```

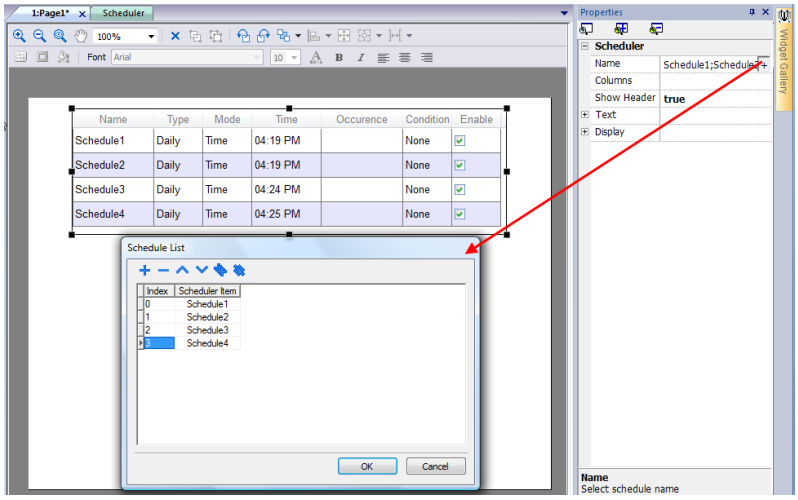
Location information is also displayed in the dialog together with sunset and sunrise times.



Configuring the Scheduler widget

To display scheduler data on a page:

1. Drag and drop a **Scheduler** widget from the widget gallery into the page.
2. In the **Properties** pane, click **+** for the **Name** parameter: the **Schedule List** dialog is displayed.
3. Add all the schedules you want to display in the page.



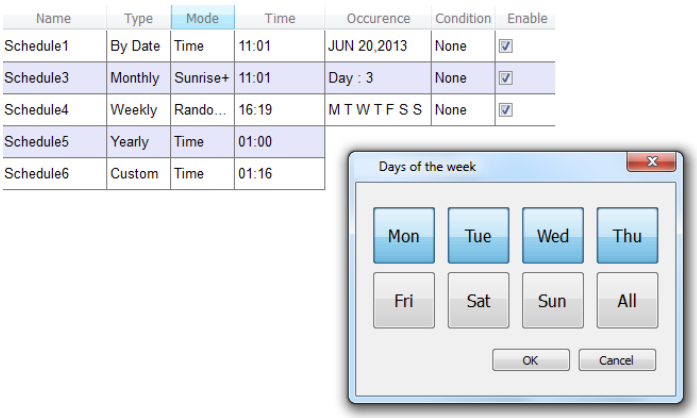
4. In the **Properties** pane, customize all settings.

Scheduler settings

Parameter	Description
Name	Schedule to be displayed
Columns	Columns to be displayed and their characteristics
Show Header	Shows/hides column headers
Time Spec	Time to be displayed at runtime
Text	Font used for text
Display	Table styles

Scheduling events at runtime

At runtime you can modify the following scheduling parameters.



Name	Type	Mode	Time	Occurence	Condition	Enable
Schedule1	By Date	Time	11:01	JUN 20,2013	None	☑
Schedule3	Monthly	Sunrise+	11:01	Day : 3	None	☑
Schedule4	Weekly	Rando...	16:19	M T W T F S S	None	☑
Schedule5	Yearly	Time	01:00			
Schedule6	Custom	Time	01:16			

Parameter	Description
Occurrence	Information on the type of schedule and time of execution
Condition	Condition applied to action execution
Enable	Enabels/disables the execution of the scheduled actions without deleting the schedule.

See "Recurring schedule" on page 305 for details on schedule parameters.

26 21 CFR Part 11 Compliance

LRH SW includes a set of functions for responding to the requirements specified in FDA 21 CFR Part 11. The standard is intended to provide a solution for securely handling electronic records and electronic signatures in industrial applications.

The table lists all the requirements specified by the regulation and reports the functions available in LRH SW for compliance.

FDA 21 CFR Part 11 compliance is optional during application development and the application developer is responsible to configure the application in the proper way.

Chapter	Description	LRH SW compliance level (v2.8)
11.10(a)	(a) Validation of systems to ensure accuracy, reliability, consistent intended performance, and the ability to discern invalid or altered records.	Reports generated by LRH SW can be signed using x.509 Certificates. A certificate that includes the public key, necessary to verify the signature of reports, will be exported with the report. References: • "SaveEventArchive" on page 201 • "PrintGraphicReport" on page 186
11.10(b)	The ability to generate accurate and complete copies of records in both human readable and electronic form suitable for inspection, review, and copying by the agency. Persons should contact the agency if there are any questions regarding the ability of the agency to perform such review and copying of the electronic records.	Application developer can select the resources (process values, alarms, etc.) whose changes will be tracked to the audit trail. Each change of the selected resources will be recorded with the name of the operator doing the change. The audit trail reports can be exported to .csv or .pdf files. References: • "Enable/disable audit trail" on page 338 • "Exporting audit trail as .csv files" on page 344 • "SaveEventArchive" on page 201 • "Printing audit table" on page 344 • "PrintGraphicReport" on page 186
11.10(c)	Protection of records to enable their accurate and ready retrieval throughout the records retention period.	Applications can be developed to self-generate signed reports to external memory or network folders at predefined interval (e.g. at the end of the day) or when circular buffer is full. User is responsible to keep these reports saved for the retention period. References: • "SaveEventArchive" on page 201 • "PrintGraphicReport" on page 186

Chapter	Description	LRH SW compliance level (v2.8)
		• "Scheduler" on page 303
11.10(d)	Limiting system access to authorized individuals.	Application developer is responsible for the appropriate security configuration of the application. References: • "User management and passwords" on page 325
11.10(e)	Use of secure, computer-generated, time-stamped audit trails to independently record the date and time of operator entries and actions that create, modify, or delete electronic records. Record changes shall not obscure previously recorded information. Such audit trail documentation shall be retained for a period at least as long as that required for the subject electronic records and shall be available for agency review and copying.	Audit trail records are stored using a circular buffer (this is to ensure that the device will not run out of memory). Audit trails cannot be modified by the operator. Each record contains a sequential number to easily check the presence of all records. The application can be developed to save/export a copy of the data at regular intervals (e.g. at the end of each day); operator is responsible for storing copy of reports in a safe place. References: • "Exporting audit trail as .csv files" on page 344 • "SaveEventArchive" on page 201 • "Printing audit table" on page 344 • "PrintGraphicReport" on page 186 • "Scheduler" on page 303
11.10(f)	Use of operational system checks to enforce permitted sequencing of steps and events, as appropriate.	Macros or JavaScript can be used to configure command sequences in the application.
11.10(g)	Use of authority checks to ensure that only authorized individuals can use the system, electronically sign a record, access the operation or computer system input or output device, alter a record, or perform the operation at hand.	The HMI application can be configured • to be accessible only after user sign in with its own password • objects can be configured to be available or not available depending on the user who logged in to the system • resources can be configured to require a password confirmation before be modified References: • "User management and passwords" on page 325 • "Electronic Signature" on page 339
11.10(h)	Use of device (e.g., terminal) checks to determine, as appropriate, the validity of the source of data input	Resources can be configured to be accessible only from selected user groups. List of allowed IP address

Chapter	Description	LRH SW compliance level (v2.8)
	or operational instruction.	can be configured from the User Management settings. References: • "Modifying access permissions" on page 327
11.10(i)	Determination that persons who develop, maintain, or use electronic record/electronic signature systems have the education, training, and experience to perform their assigned tasks.	Application developer is responsible to define and assign the appropriate user rights to each user that have access at the HMI device
11.10(j)	The establishment of, and adherence to, written policies that hold individuals accountable and responsible for actions initiated under their electronic signatures, in order to deter record and signature falsification.	Application developer is responsible for establishing appropriate procedures.
11.10(k)	Use of appropriate controls over systems documentation including: (1) Adequate controls over the distribution of, access to, and use of documentation for system operation and maintenance. (2) Revision and change control procedures to maintain an audit trail that documents time-sequenced development and modification of systems documentation.	Application developer is responsible for establishing appropriate procedures.
11.30	Persons who use open systems to create, modify, maintain, or transmit electronic records shall employ procedures and controls designed to ensure the authenticity, integrity, and, as appropriate, the confidentiality of electronic records from the point of their creation to the point of their receipt. Such procedures and controls shall include those identified in 11.10, as appropriate, and additional measures such as document encryption and use of appropriate digital signature standards to ensure, as necessary under the circumstances, record authenticity, integrity, and confidentiality.	LRH SW has been designed for operation in closed systems.
11.50(a)	Signed electronic records shall contain information associated with the signing that clearly indicates all of the following: (1) The printed name of the signer; (2) The date and time when the signature was executed; and (3) The meaning (such as review, approval,	All records will be added to the audit trail with time stamp and user id of logged user. References: • "Exporting audit trail as .csv files" on page 344 • "Table audit widget" on page 343

Chapter	Description	LRH SW compliance level (v2.8)
	responsibility, or authorship) associated with the signature.	
11.50(b)	The items identified in paragraphs (a)(1), (a)(2), and (a)(3) of this section shall be subject to the same controls as for electronic records and shall be included as part of any human readable form of the electronic record (such as electronic display or printout).	
11.70	Electronic signatures and handwritten signatures executed to electronic records shall be linked to their respective electronic records to ensure that the signatures cannot be excised, copied, or otherwise transferred to falsify an electronic record by ordinary means.	Application developer is responsible for avoiding using the macros that permit the import/export of user passwords.
11.100(a)	Each electronic signature shall be unique to one individual and shall not be reused by, or reassigned to, anyone else.	System will ensure that two users with the same id cannot be defined. It is user responsibility to avoid removal and reassignment of the same user id to a different user.
11.100(b)	Before an organization establishes, assigns, certifies, or otherwise sanctions an individual's electronic signature, or any element of such electronic signature, the organization shall verify the identity of the individual.	User responsibility.
11.100(c)	Persons using electronic signatures shall, prior to or at the time of such use, certify to the agency that the electronic signatures in their system, used on or after August 20, 1997, are intended to be the legally binding equivalent of traditional handwritten signatures. (1) The certification shall be submitted in paper form and signed with a traditional handwritten signature, to the Office of Regional Operations (HFC-100), 5600 Fishers Lane, Rockville, MD 20857. (2) Persons using electronic signatures shall, upon agency request, provide additional certification or testimony that a specific electronic signature is the legally binding equivalent of the signer's handwritten signature.	User responsibility.
11.200(a)	(a) Electronic signatures that are not based upon biometrics shall: (1) Employ at least two distinct identification components such as an identification code and password.	LRH SW Security functions are based on the combination Username/ Password.

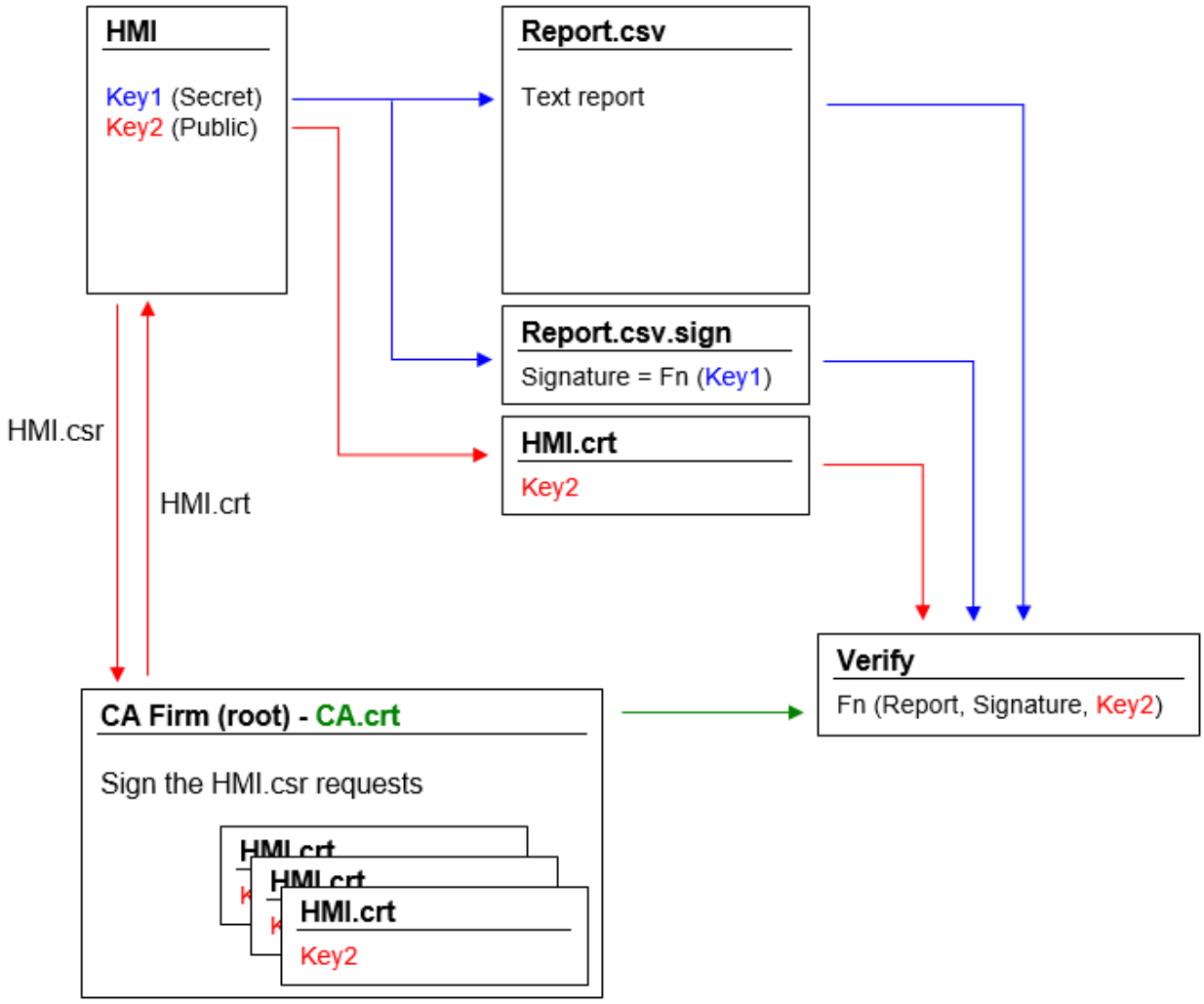
Chapter	Description	LRH SW compliance level (v2.8)
	(i) When an individual executes a series of signings during a single, continuous period of controlled system access, the first signing shall be executed using all electronic signature components; subsequent signings shall be executed using at least one electronic signature component that is only executable by, and designed to be used only by, the individual. (ii) When an individual executes one or more signings not performed during a single, continuous period of controlled system access, each signing shall be executed using all of the electronic signature components.	Users must enter name and password to access the system. Critical actions can be configured to require entering again the password before execution is started. References: • "User management and passwords" on page 325 • "Electronic Signature" on page 339
	(2) Be used only by their genuine owners; and (3) Be administered and executed to ensure that attempted use of an individual's electronic signature by anyone other than its genuine owner requires collaboration of two or more individuals.	Each user is responsible to not divulge own password. Passwords defined by administrator for first access can be forced to be redefined at first use. References: • "Configuring users" on page 333
11.200(b)	Electronic signatures based upon biometrics shall be designed to ensure that they cannot be used by anyone other than their genuine owners.	LRH SW does not support biometrics.
11.300(a)	Maintaining the uniqueness of each combined identification code and password, such that no two individuals have the same combination of identification code and password.	It is not possible to define to define two users with the same User ID
11.300(b)	Ensuring that identification code and password issuances are periodically checked, recalled, or revised (e.g., to cover such events as password aging).	System can be configured to force each users to define a new and different password after a configurable number of days References: • "Configuring users" on page 333
11.300(c)	Following loss management procedures to electronically deauthorize lost, stolen, missing, or otherwise potentially compromised tokens, cards, and other devices that bear or generate identification code or password information, and to issue temporary or permanent replacements using suitable, rigorous controls.	Users can change their password at any time. Administration can redefine each user's password and force them to redefine at the first login. References: • "User management actions" on page 210 • "Configuring users" on page 333
11.300(d)	Use of transaction safeguards to prevent unauthorized use of passwords and/or identification codes, and to detect and report in an immediate and urgent manner any attempts at their unauthorized	Failed logging attempts are logged to audit trail.

Chapter	Description	LRH SW compliance level (v2.8)
	use to the system security unit, and, as appropriate, to organizational management.	
11.300(e)	Initial and periodic testing of devices, such as tokens or cards, that bear or generate identification code or password information to ensure that they function properly and have not been altered in an unauthorized manner.	User is responsible for ensuring appropriate measures.

x.509 Certificate

To ensure authenticity of reports generated by HMI devices, LRH SW HMI Runtime can generate reports with signed files to verify the authenticity and the integrity of the generated reports.

LRH SW HMI Runtime uses asymmetric cryptography keys to sign files and x.509 standard to manage public key certificates. The picture shows the architecture.



The public key can be signed by a Certificate Authority (CA) that guarantees its authenticity.

Workflow

1. Each HMI device contains two keys:
   - Key1 is the secret key, that is used to sign the reports generated by the HMI device. This key is securely stored inside the HMI device.
   - Key2 is the public key that anyone can use to verify the authenticity of the reports signed by the HMI device.
2. The macros "*SaveEventArchive*" or "*PrintGraficReport*" can be used to generate signed reports (see "SaveEventArchive" on page 201 or "PrintGraphicReport" on page 186 for additional details)
3. For the .csv file, you can use the public key and the signed file to verify the report is authentic and not tampered. (See "Signed CSV files" on the next page)
4. For the .pdf file, you can use a PDF reader to verify the report is authentic and not tampered. (See "Signed PDF files" on page 320)

The internal x.509 certificate files

Each HMI devices already have a self-signed certificate. You are free to use it, ask a Certificate Authority to sign it, create a new one using the information that you prefer or to upload and use your own certificate. All operations are available from the device "*System Settings*" (see the x.509 Certificate section inside the "System Settings" on page 543).

Note that you can never retrieve the private key from the HMI device. You can instead provide a certificate with both private and public keys.

Use the self-signed certificate

To use the self-signed certificate you have to do nothing. Simply, use the macros that generate signed reports. Even if the certificate will be provided from the macros, you can use the "*System settings*" to retrieve your copy of the certificate (just to be sure of the originality of the certificate).

Use a certificate signed from a Certificate Authority

To use your signed HMI certificate from a certificate authority you must download the certificate signing request file from the "*System settings*" panel. Sending and asking a certificate authority to sign the certificate (generally this is a pay operation) and then upload the signed certificate to the HMI device.

After retrieving the "certificate signed request" file to send to the certificate authority, be sure to never regenerate a new certificate otherwise the internal private key associated with the certificate send to the authority will be lost.

Use your own certificate

If you have your own Certificate and you like to use it, you can upload it inside the HMI device from the "*System Settings*" panel. Note that you must provide both private and public keys.

When the certificate contains a private key, the current private key will be substituted with the key found in the certificate and it will not be possible to recover it.

Example of a certificate with both public and private keys (certificates are encoded base64).

ssl-certificate.crt

```
-----BEGIN CERTIFICATE-----
MIIDBDCCAewCCQDcBYW7PYwJsDANBgkqhkiG9w0BAQsFADBEMQswCQYDVQQGEwJJ
VDEPMA0GA1UEBwwGVmVyb25hMRMwEQYDVQQKDApUZXN0T2ZmaWNlMQ8wDQYDVQQD
DAZITUktMDQwHhcNMTcwNjI2MDgwOTQ1WhcNMTgwNjI2MDgwOTQ1WjBEMQswCQYD
VQQGEwJJVDEPMA0GA1UEBwwGVmVyb25hMRMwEQYDVQQKDApUZXN0T2ZmaWNlMQ8w
DQYDVQQDDAZITUktMDQwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDc
Nlp2kswcbLh4IxS6eeCgQ4EAUHCRpaZ5YPfQ/un9/s0tejaa3Si3Pcqv3JqddJM8
mJEZaPF/+HhAEhtC+rv57TbgullUQJdoQpfoGChofpULforXZt2BfdWNx67plNoa
YM3ElaNtAKIW2o6S9HGEv1kf09XFLGkFgeMgC59+SejgguCNT0m99m6fNa5910I7
UDJFINkC3bxtONj+WiL/iEZYkHXacaN9q06fx+2NfmiSsXGPnmSys5mocqo89tMa
TjyeF7jYpDccCpJ9pY4xRjRpcIkDCM7PabVoG/ascSMUU6XPE2R0W4UJ6bPAygD6
QLKCCq0BUi6/eUj0pnanAgMBAAEwDQYJKoZIhvcNAQELBQADggEBAMLfIEXQOEjS
OpwVkzNxXmL/A6PLU5BK1hVYHb7ofb2Z37zN69vCn8ESglAFYK7QhkhJu3zAD+jH
fYBVKVdxfd3HS8EmcDWxpC6F21fgqsSqepMRTbKbsSaO53a7JsXtwnHVNfG6EBZV
8tqSlGc4RwtJeVZJelUdmWSBD4Fc7asFeBCKqLrHJinz7buj3I4fLcyscTaMTBI9
fsE7poEpWvKc7NWtKYZglGG3AG6xONu3sEahcJ5k+UVdh/QQdAiCt3vG+JJ/owYU
sd30WIZ4pNzG/GUH9MbJyvI4ftA8IvEhGxHvi3xt7slJnvYQDaghOEDhdtGvilOr
nJZ2FZOBCEI=
-----END CERTIFICATE-----
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEA3DZadpLMHGy4eCMUunngoEOBAFBwkaWmeWD30P7p/f7NLXo2
mt0otz3Kr9yanXSTPJiRGWjxf/h4QBIbQvq7+e024LtZVECXaEKX6BgoaH6VC36K
l2bdgX3Vjceu6dTaGmDNxJWjbQCiFtqOkvRxhL5ZH9PVxSxpBYHjIAuffkno4ILg
jU9JvfZunzWufZdCO1AyRSDZAt28bTjY/loi/4hGWJBl2nGjfatOn8ftjX5okrFx
j55ksrOZqHKqPPbTGk48nhe42KQ3HAqSfaWOMUY0aXCJAwjOz2mlaBv2rHEjFFOl
zxNkdFuFCemzwMoA+kCyggqtAVIuv3lI9KZ2pwIDAQABAoIBAGnamsuqrwDu5hGh
02H8GhUPvd/3ytTISujHyvgkwTf+FoTI3Zy9uMe0pUy5/3y2v9v9/qm3P3djafJq
gb5Fprxx4dJPXJZaYi2U7U585lesmVqoHneCk/GeGlyH4zWlwo2xgNgBkkhgaIoR
zz0m0bachVz+SCD6wxUJpbMOw0FBw54oPL0XS/gD+76S9ET7xmqZAS5xV/w8Khht
PtjPfT58GKhqVIC9cMrrBrkuGQPrNrDaJMPsQDxrFp7POQm4+GivrUJ0FA9Vtx46
C5QhXqVps/BODo3mjeOcj2b/FqsvG7WCc5PWOAcCqStmDxl+DQZOIVFSTrE4kdpg
mNn/80kCgYEA88Xfmqg0ta83lpe9b6U0BaLvvslgxgXmCmkyvK7Ru+iKyPUMzxB+
BjGWeeiZuigmIhXfFu3eBs5xOgDrUxf9j55sJAFamljG4LTyun378RnOdA87fflq
rpF4oPKVfTrfXXzZkeIg0eX2tD6Lsn3+MJwYqpefovxmyJA3kPgcGv0CgYEA50H0
HQififZZ2nApgPf/jJpU7hBLC45cSXvE2MX2I3rd3ptGwzKRo/lZkslbvQutqRln
slyEF+c9LCz6g7FYhJoewChLqCVfeZ9GxBzHeJloxZwmxDXi8L4vmEDphwlcV8b3
ExHqUlMGuINHGelPIRlLKeEsbTQU+OVHuNv443MCgYEA7rMKYhllC6bYCsjowSMG
TqKembX84cqyl+zstp+EVbi99Usm0Lc4f/4cd6EQrplTwbqi6YPgDdAmRQLTalkp
e3FIOPVub4aQr0XgDEcC5bI8W57yxUrZJLjjYs5HHQoB4Dw5m0TOmFnS+enoxs3i
kly3Nowjz+fRCYFWN8kLVE0CgYEA43CLLK7ZcW9XKa2cNBo0PE1g8A4YMJJfk2nl
zKjNjlF9ujyO2NV4RYOsI+RSsFe3ARdJcS6xP20OTc8ixrh57VhCnAxFdGblQpFy
oNgJGkf9zjPoMJsqykjSOHTG+CctqaqmPxwkkLScbIW4PPSn/U6KDPNHpVNOuQeO
hXHak58CgYBLW1719vgYhUiSWc9Gd3mCSxpAb6y8RcyTgqF76K8v4MalLPqFkEtD
0BaFtlA+PtMLk2ODTRH4XU18oc9eV+7VDFkPJ8T0A2VwjzjMgNAd+vKlm4nOEBTt
UhegY0k8yLxSlZvuYiVnHvKBIoF/G2ckwrxjO9KVE+SA45Ex0Px5qQ==
-----END RSA PRIVATE KEY-----
```

You can import inside each HMI device the same certificate file to have a unique public certificate file for all your HMI devices.

Signed CSV files

Reports generated in CSV format using the **SaveEventArchive** macro can be signed using the x.509 certificate included inside the HMI device. The signature makes sure that nobody tampered with the content of the document since it was

signed.

See also:

- The SaveEventArchive parameters ("SaveEventArchive" on page 201)
- How to provide an x.509 Certificate to Linux devices ("x.509 Certificate" on page 559)
- How to provide an x.509 Certificate to WinCE devices ("System Settings" on page 1)

When required, using Signed=True, the SaveEventArchive macro in addition of the [ReportName].csv generate other two files:

- [ReportName].csv.sign
- ssl-[CertificateName].crt

Where the [ReportName].csv.sign is the signature of the report and the ssl-[CertificateName].crt is a copy of the x.509 certificate of the HMI device. Note that you can retrieve the certificate of the HMI device even from the System Setting of the HMI device.

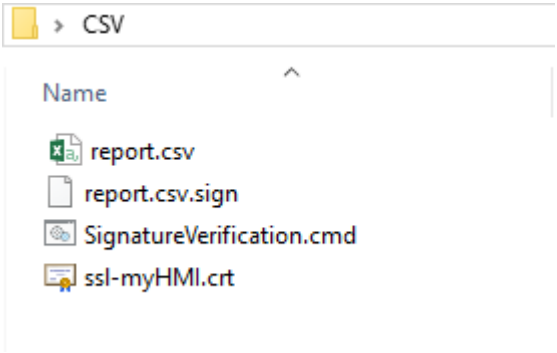
How to verify the report's signature using the public OpenSSL library

To verify that nobody has tampered the content of the report you need

- be sure the ssl-[CertificateName].crt is coming from the HMI device
- use a tool to verify the signature (e.g. OpenSSL-Win32)

Reference.: https://www.openssl.org/

To verify that the .csv report generate from HMI device has not tampered you can install a public OpenSSL library, copy all files generated from the macro inside the same folder and use the below batch file

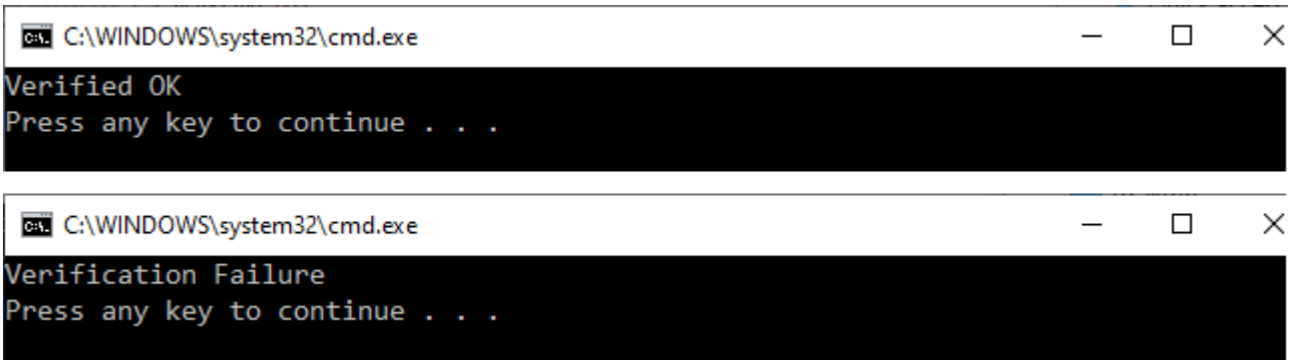


File: *SignatureVerification.cmd*

```
@echo off
set OpenSSL="C:\Program Files (x86)\OpenSSL-Win32\bin\openssl.exe"
set FileToCheck=Report.csv
set hmiCertificate=ssl-myHMI.crt

rem Extract public key from the certificate
%OpenSSL% x509 -in %hmiCertificate% -pubkey -noout > publicKey.pem

rem Verify Signature
%OpenSSL% dgst -sha256 -verify publicKey.pem -signature %FileToCheck%.sign
%FileToCheck%
```

```
rem Remove public key
del publicKey.pem

pause
```

The below pictures are showing the possible outputs of the batch file



On Linux devices, the BSP v1.0.239 or greater is required

Signed PDF files

Reports generated in PDF format using the **PrintGraphicReport** macro can be signed using the x.509 certificate included inside the HMI device. The signature makes sure that nobody tampered with the content of the document since it was signed.

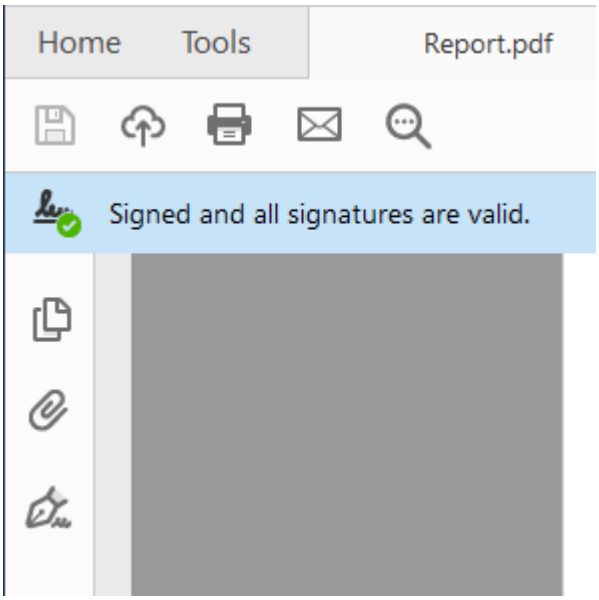
See also:

- The PrintGraficReport parameters ("PrintGraphicReport" on page 186)
- How to provide an x.509 Certificate to Linux devices ("x.509 Certificate" on page 559)
- How to provide an x.509 Certificate to WinCE devices ("System Settings" on page 1)

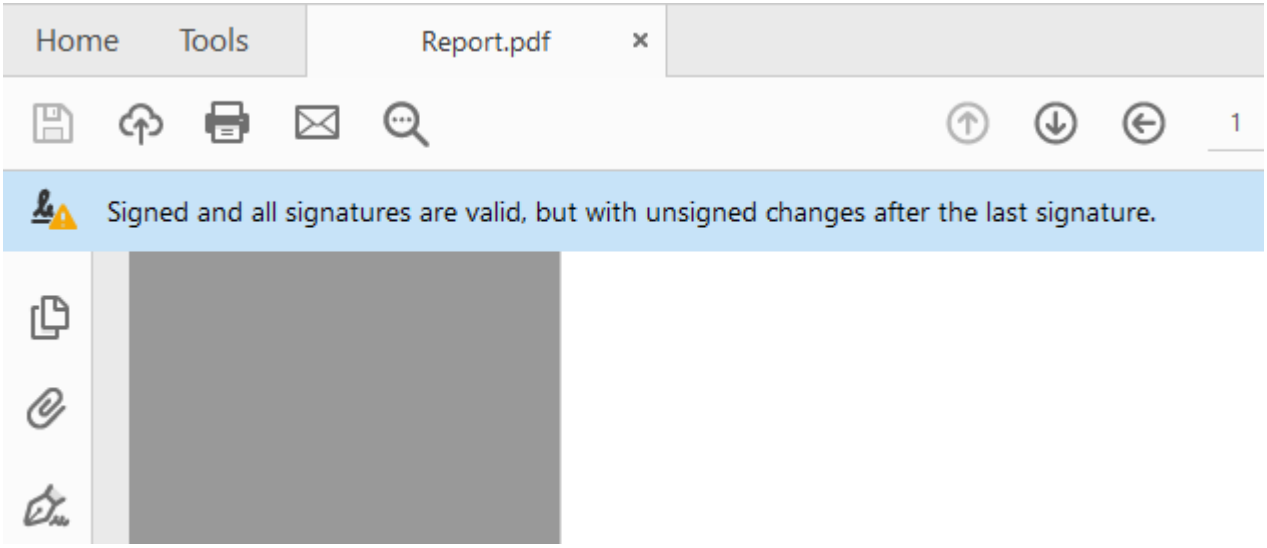
When you open the file, the PDF reader tries to decide if the signature is valid then it looks at the certificate used to sign the document.

x.509 certificate signed from a Certificate Authority

If you have uploaded to the operator panel a valid x.509 certificate, signed by a Certification Authority, when you open the generated PDF file you will get a message that highlights the document is valid.



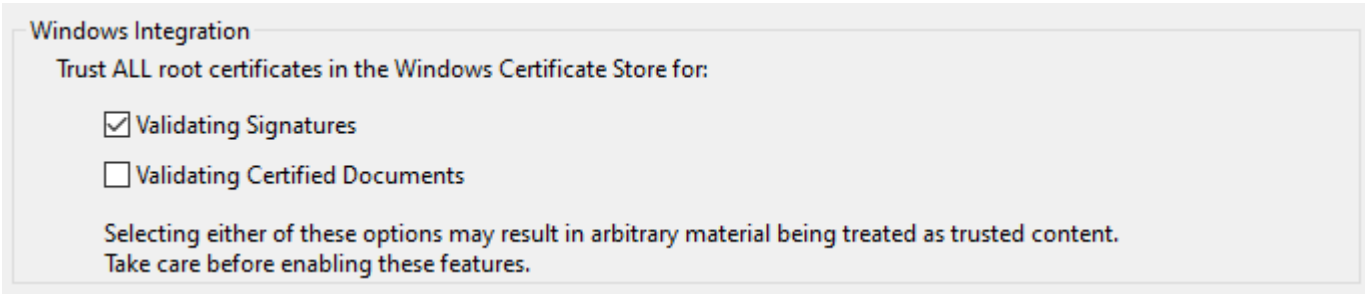
If the document has been modified, it will be highlighted with a different message.



Certificate Trust and Authenticity

Trust of signed certificates depends on the issuer of the certificate. The PDF reader will trust a certificate if you have told it to trust the issuer of that particular certificate. By default the Adobe Reader only trust certificates issued by Adobe or one of their partners. This means that it will show a warning if the certificate wasn't issued by one of these authorities. Microsoft Windows also uses certificates for validating software vendors and content providers. You can configure your Adobe Reader to trust these issuers in addition to the Adobe partners.

Check inside the preferences of the PDF reader if you want to enable the PDF reader to use even the Microsoft Windows certificates

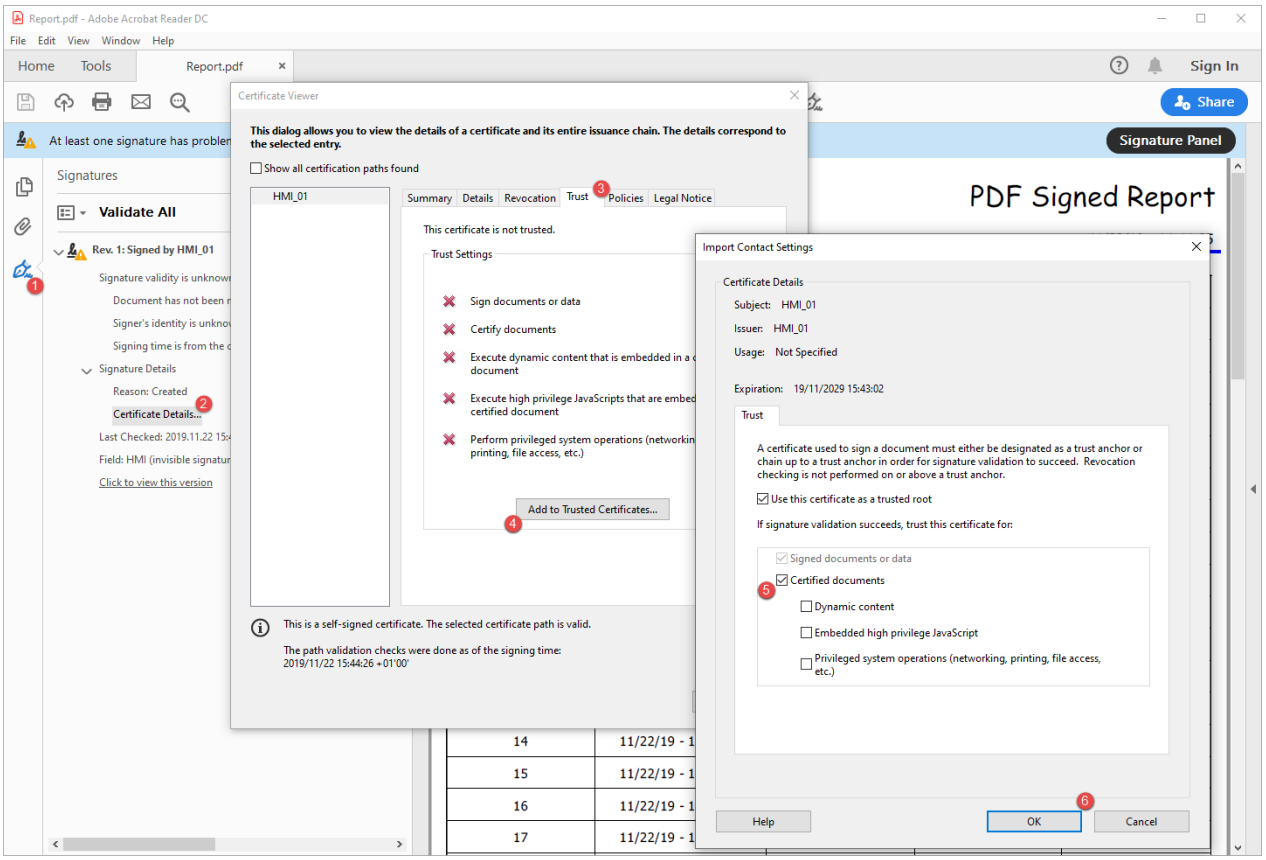


x.509 self signed certificate

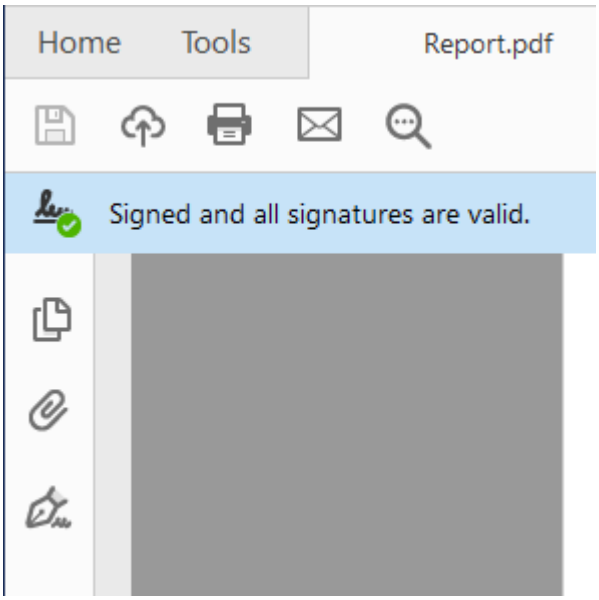
A self-signed certificate is a certificate that is not signed by a certificate authority (CA).

This means that PDF Reader can confirm the file is signed and not tampered, but cannot confirm the signature (alias the certificate) is authentic. Is the user have to take care to verify the certificate is authentic (for example, making sure that the document was actually produced by the panel) and confirm to the PDF reader that the certificate included in the document is valid and that can be considerate valid even for the next reports.

Steps to manual confirm that the certificate is authentic:



Now, if you close and reopen the PDF document you will get the valid signature. Moreover, even all other documents produced from the same HMI device will be shown with the correct signature because the information that the certificate is authentic has been stored inside settings of the PDF Reader.



On Linux devices, the BSP v1.0.507 or greater is required

Compliant applications

Suggestions to development a CFR11 compliant applications

User management macros

User management macros that could be use from any user

- Login
- Logout
- SwitchUser
- ChangePassword

User management macros that could be used from administrator only

- ResetPassword
- AddUser
- EditUsers
- ExportUsers,

Deprecated macros that must not be used inside CFR 21 part 11 compliance applications

- ImportUsers
- DeleteUser
- DeleteUMDynamicFiles,

27 User management and passwords

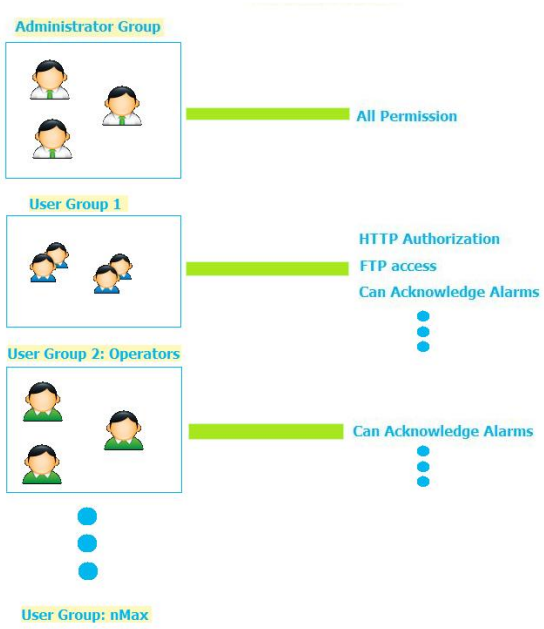
You can restrict access to various widgets and operations by configuring users, users groups and assigning specific authorizations to each group.

Each user must be member of one and only one group. Each group has specific authorizations and permissions.

Authorizations and permissions are divided in two categories:

- Widget permissions: hide, read only, full access
- Action permissions: allowed or not allowed.

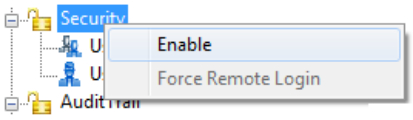
By organizing permissions and groups you can define the security options of a project.



Enable/disable security management

Path: ***ProjectView*** *>* *right-click* ***Security*** *>* ***Enable***

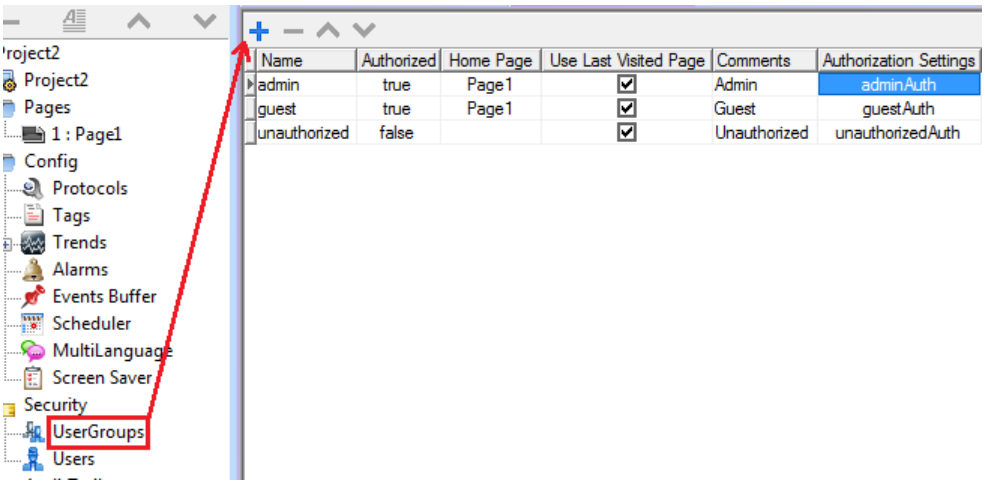
The padlock symbol indicates whether the function is enabled or disabled.



Important: Security settings are effective only if the security function is enabled.

Configuring groups and authorizations

Path: ***ProjectView*** *>* ***Security*** *>* *double-click* ***UserGroups***



Three predefined groups are available by default (**admin**, **guest** and **unauthorized**): they cannot be deleted nor renamed. You can, however, modify authorizations and other settings.

Adding a user group

Click **+** to add user group.

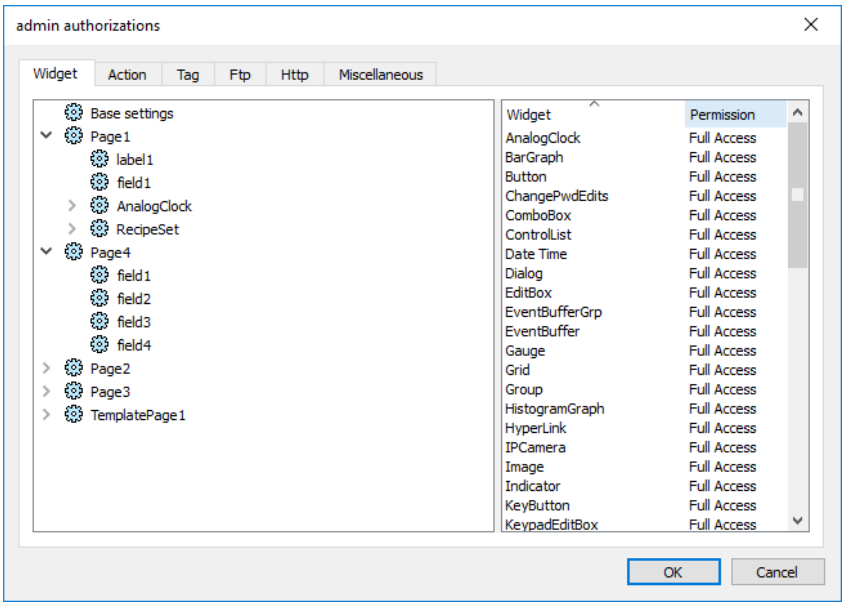
Parameter	Description
Name	Name of users group
Authorized	Authorization granted
Home Page	Page displayed when users belonging to this group log in
Use Last Visited Page	When selected, the last page displayed by the previous user will be displayed when users belonging to this group log in

Parameter	Description
Comments	Any comment or description for the group
Authorization Settings	Opens the Admin Authorization dialog to set access permissions. See "Modifying access permissions" below for details.

Modifying access permissions

Path: ***ProjectView*** *>* ***Security*** *> double-click* ***UserGroups*** *>* ***Authorization Settings*** *column*

Click the button: a dialog appears with a list of widgets and actions. You can modify access permissions for each one in the list.



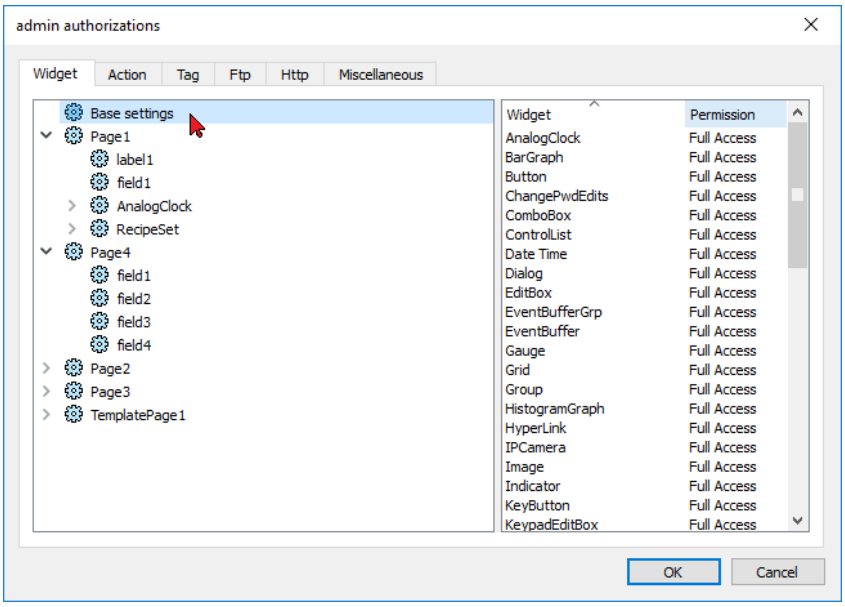
Widget permissions

In the **Widget** tab you can define widget access options at project level, at page level or at widget level for all the widgets used in the project. Lower levels permission (for example, widget level) overrides higher levels (that is, page and project levels).

Use **Base settings** to set default permissions at project level.

Possible settings are:

- **Full Access** to enable read/write access to the widget
- **Read Only** to enable readonly access to the widget
- **Hide** to hide widget for selected group



Changing a widget permission

To change access permission for an individual widget in a page of the project, navigate to that widget within its page on the right pane and customize its access options. Otherwise, all widgets take the permissions set at project or page level.

For example, if page permission for a widget is set at project level to **Read Only**, then all the same widgets will have permission **Read Only**. When you select a widget inside a page from the tree structure, permission is actually set to **Use Base Settings**. You can change this setting and modify access permissions only for this widget in this page.

Access priority

Widget permissions are considered with the following priority:

Permission level	Priority
Project level - Basic settings	Low
Page level	Medium
Widget level	High

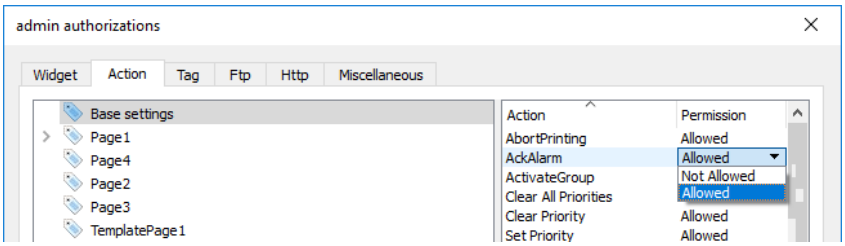
This allows you to specify exceptions for an action or a widget directly from the page view.

For example, if you set permissions for a widget at project level to Read Only and to Full Access at page level then the page level settings will prevail.

Access permissions can be modified directly from the project page. See "Assigning widget permissions from page view" on page 332 for details.

Action permissions

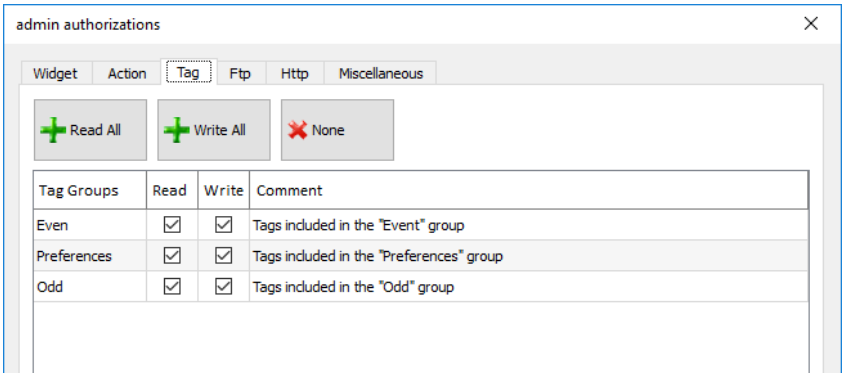
In the **Action** tab you can define action authorizations at project level, at page level or at widget level. Actions can be either **Allowed** or **Not Allowed**.



Action permissions can be modified directly from the project page. See "Assigning widget permissions from page view" on page 332 for details.

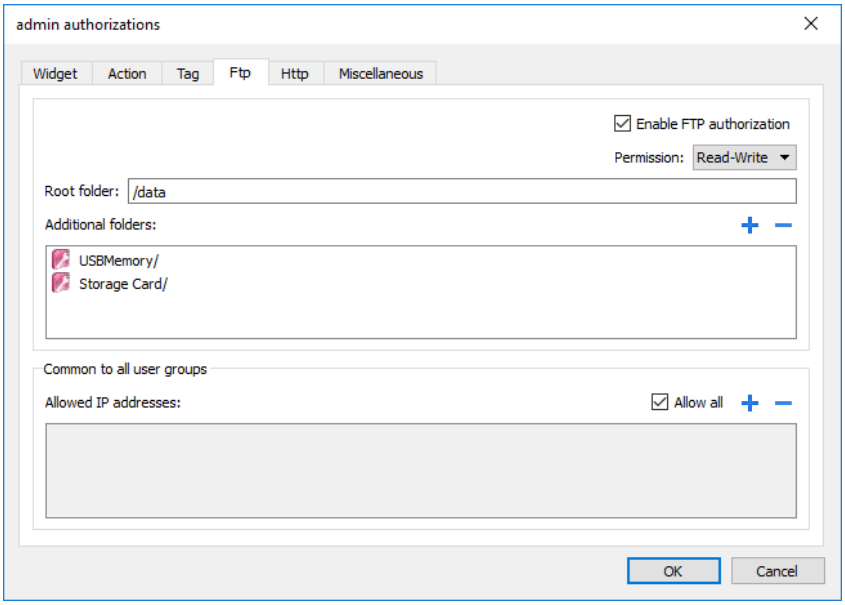
Tag permissions

For each group of tags, you can define the Read/Write access rights



FTP authorizations

In the **Ftp** tab you can set specific authorizations for the FTP server.



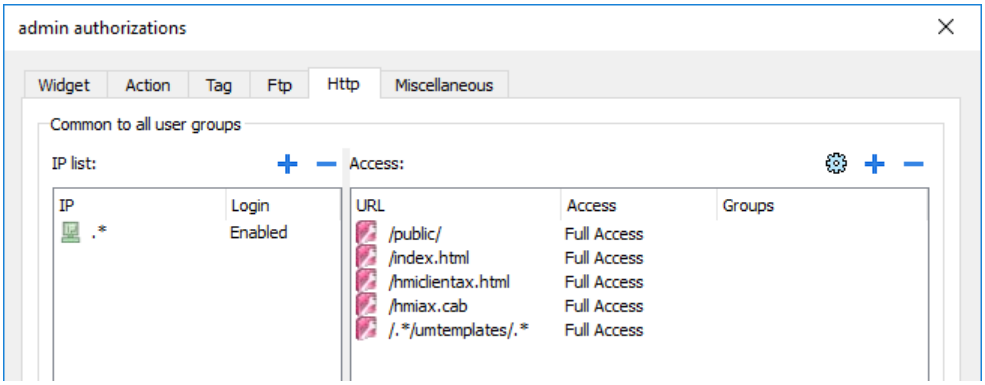
Element	Description
Enable FTP authorization	Enables the FTP function for the specific group
Permission	Type of permission: • **Read-Only** • **Read-Write**
Root Folder	Folder to be used as root for FTP access. This is a relative path.
Additional folder	Extra folders to be used as root for FTP access (for example, on USB drive or SD card)
Allowed IP Addresses	List of IP addresses from which FTP connection can be accepted. **This setting is common to all users groups.**

HTTP authorizations

In the **HTTP** tab you set restrictions to HTTP access to the web server integrated in LRH SW HMI Runtime.

Wildcards can be used to identify a range of IP addresses.

For example, the two following rules set the HMI device unit can only be accessed by all the IP addresses 192.168.*.* on your local network in which only the IP address of 192.168.1.20 can access the device without entering a login name.



Element	Description
IP list	IP addresses authorized to access the HTTP server. By default the login is required from any IP address (IP=.*, Login=Enabled).
Login	When disabled, the username and password are not required.
Access limits	List of resources for which access is limited

Effect of these settings depends on whether the option **Force Remote Login** has been selected. See "Force remote login" on page 336 for details.

Force Remote Login	Default Access to workspace	Access limits
-	Full	-
Disable	Full	Can be used to block access to some files/folders or to require authorization
Enable	No Access	Can be used to open access to files/folders

Important: This setting is common to all users groups.

Adding an HTTP configuration

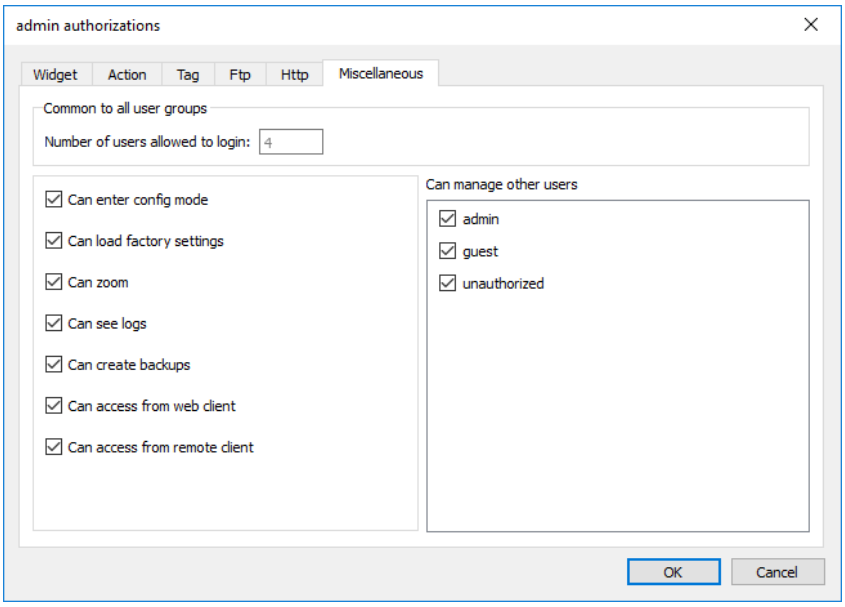
To add and configure a new access click **+**: the **Access limits** dialog is displayed.

To restore the default configuration click the **Set default access limits** icon. Default configuration allows access to the following:

- PUBLIC folder and Index.html

Miscellaneous settings

In the **Miscellaneous** tab you can define various authorization settings.



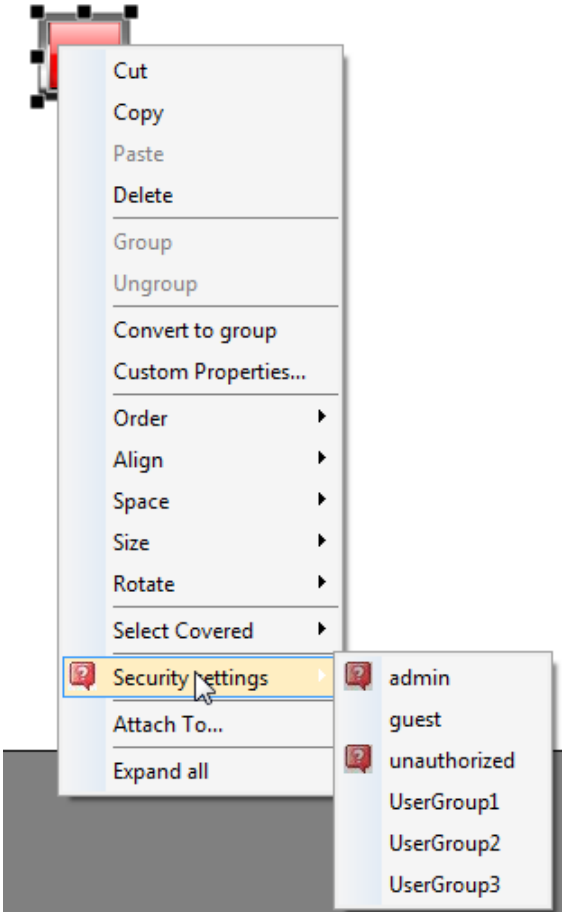
Option	Description
Can enter config mode	Enables switching from runtime to configuration mode. Normally used for maintenance.
Can load factory settings	Restores factory settings.
Can zoom	Enables zoom in/out in context menu at runtime

Option	Description
Can see log	Allows user to see logs at runtime
Can create backup	Allows user to backup project.
Can access from web client	Enables connecting from a web client
Can access from remote client	Enables connecting from LRH SW Client
Can manage other users	Gives super user privileges at runtime to manage the select groups. Allows adding, deleting and modifying users' permissions.
Number of users allowed to login	Maximum number of users that can be connected to the HMI Runtime at the same time. **This setting is common to all users groups.**

Assigning widget permissions from page view

You can assign different levels of security, to different user groups, on a single widget, directly from the project pages.

1. Right-click on the widget and select **Security settings**.
2. Choose the group: the authorization dialog for the group is displayed.
3. Set the security properties to access the widget.

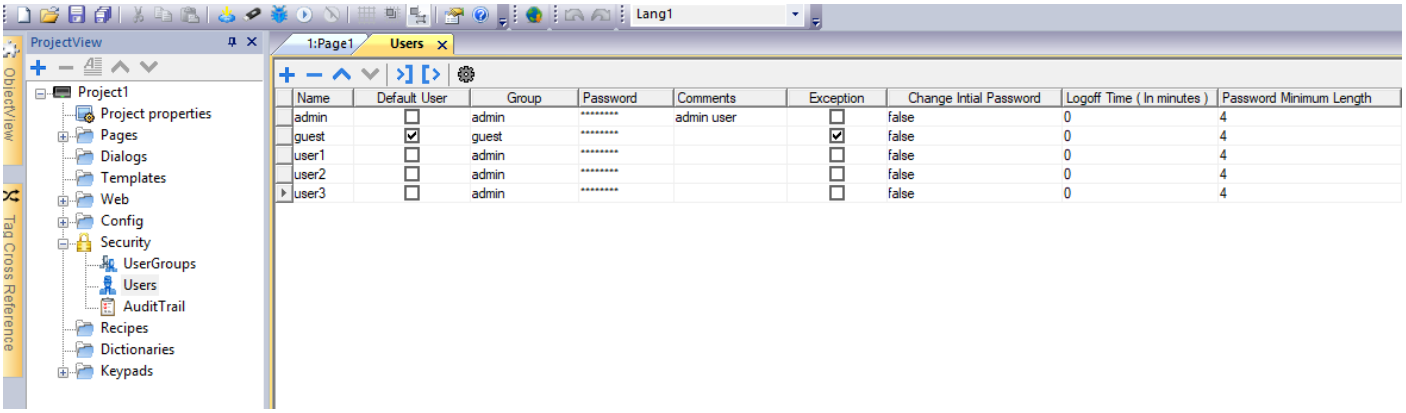


See "Modifying access permissions" on page 327 for details.

Configuring users

Path: ***ProjectView*** *>* ***Security*** *> double-click* ***Users***

In the Users editor, click **+** to add a user: one row is added to the table.

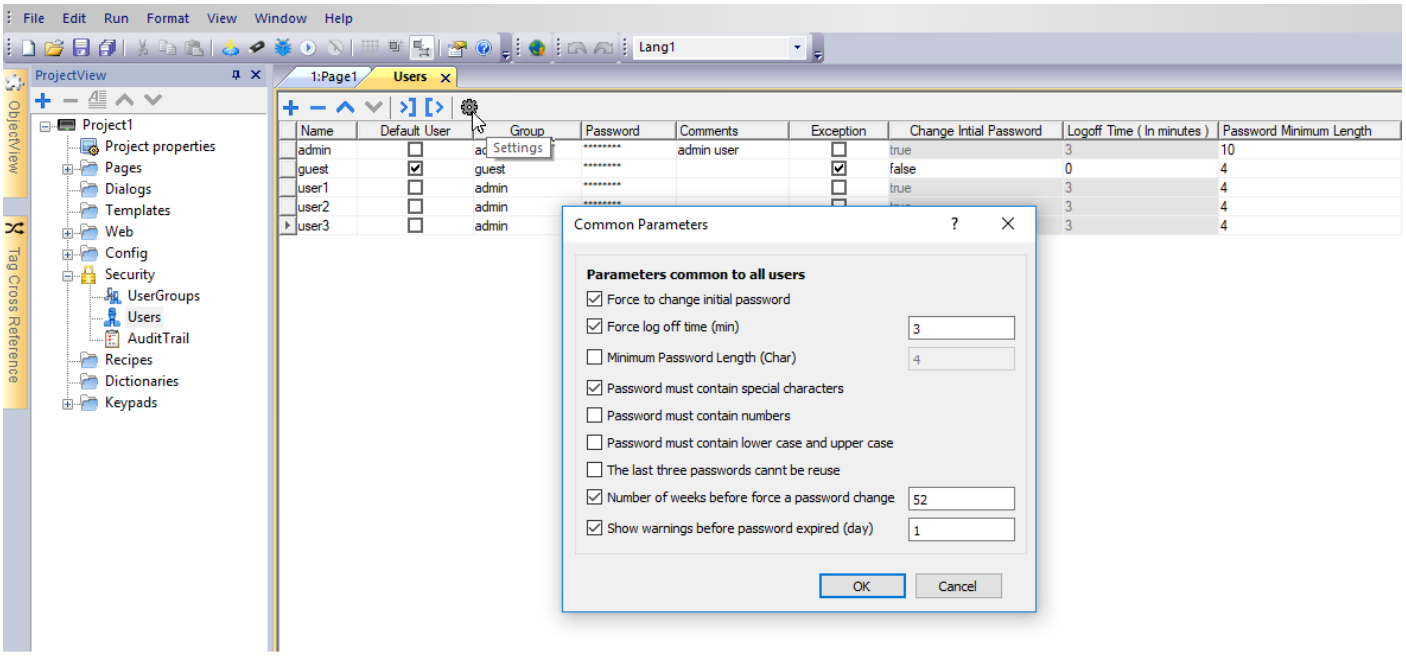


Parameter	Description
Name	User name
Default User	This user is automatically logged in when the system is started or after another user has logged off. Only one Default user can be set
Group	User group
Password	User password. Note that for security reasons the password will never be displayed  **Passwords are encrypted and cannot be retrieved not even for specialized technicians**  New project are create with "admin" default user. Password for this user is "admin". It is recommended to change this password when setting up User Management.
Comments	Further user description
Exception	Allows to change the values forced from the User Settings parameters
Change Initial Password	This user is forced to change his password at first log in
Logoff time (minutes)	Minutes of inactivity after which the user is logged off. Set to 0 to disable
Password minimum length	Minimum length of password
Must contain special characters	Password must contain at least one special character
Must contain numbers	Password must contain at least one numeric digit
Must contain lower case and upper case	Password must contain lower case and upper case
Password cannot be reused	The new password must be different from the last 3 used passwords
Password aging (weeks)	Number of weeks before forcing a password change (1/52 weeks)
Warning (days)	Show a warning message before password expires (1/30 days)

Users Settings

From the Settings command, there is the possibility to define parameters values that will be common to all users.

Users with the Exception flag checked are not force to use the common parameters.



Default user

You can define only one default user in a project. This is the user automatically logged in at system start up and when the currently logged user logs out or is logged out after time-out.

To log into LRH SW HMI Runtime with a different user, use one of the actions:

- **SwitchUser**
- **LogOut**

See "User management actions" on page 210for details.

Managing users at runtime

The default user, if any, is automatically logged in when the HMI Runtime is started. If no default user is configured, the system requires a user name and password. See "User management actions" on page 210for details on the actions that can be executed on users.

Removing user data

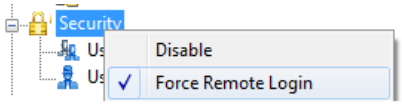
All the user information modified at runtime is stored in dedicated files. To remove these dynamic files and all the changes applied to user configuration at runtime you can:

- on HMI Runtime: execute the action DeleteUMDynamicFile
- with LRH SW: select the **Delete Dynamic Files** in the download dialog.

Note: When any modification is performed on user management in Studio, it is needed to delete User Management dynamic files to apply new User Management settings.

Force remote login

Path: ***ProjectView*** *>* *right-click* ***Security*** *>* ***Force Remote Login***



Select this option to force user to log in when using remote access viaLRH SW Client. If not selected, remote access will use the same level of protection of local access.

Important: This function only works when user management is enabled.

WARNING: Use this option when you have a default user but at the same time you want to protect remote access.

See "Enable/disable security management" on page 326for details.

The only files/folders still accessible when this flag is enabled are:

- PUBLIC folder and Index.html.

See "Modifying access permissions" on page 327 for details on HTTP access limits.

28 Audit trails

The Audit trail is a chronological sequence of audit records. Each record contains information on the actions executed and the user that performed them.

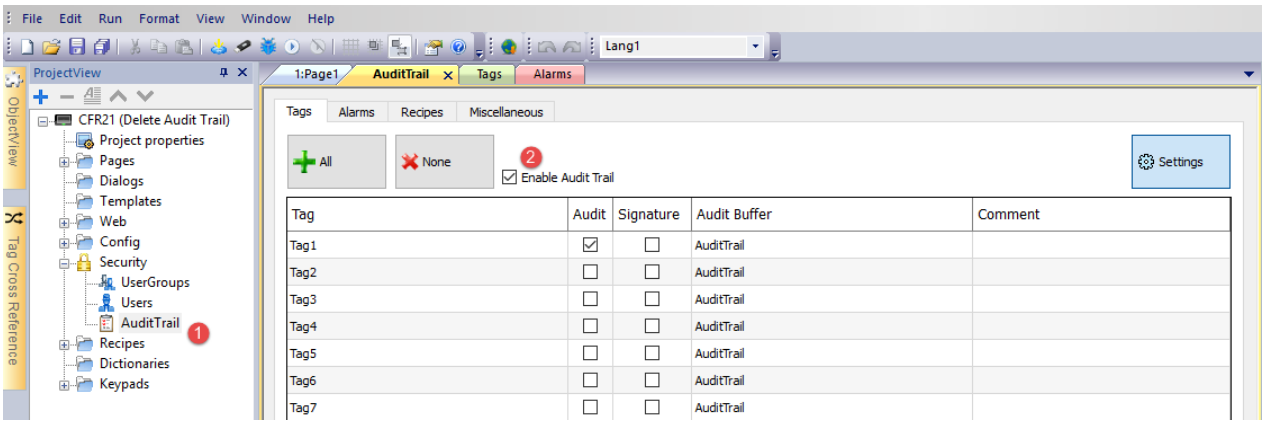
This function provides process tracking and user identification with time stamp for events.

Enable/disable audit trail

Path: ***ProjectView*** *>* ***Security*** *> double-click* ***AuditTrail***

Audit trail logging can be enabled from the “Enable Audit Trail” check box

When enabled, all changes to the selected resources will be logged to the audit buffer with the time stamp, user name that performed the operation and some additional information concerning the modified resource (e.g. new value and previous value for tags)



From the main tabs (Tags, Alarms, Recipes and Miscellaneous) of the Audit trail Editor you can switch between the list views of the available resources.

Parameter	Description
Audit	Enable tracking of the selected resource
Signature	The user password is required before allowing the resource to be modified from the user (see "Electronic Signature" on the facing page to additional information)
Audit Buffer	Internal buffer where store the related audit events (see "Events Buffer" on page 249 to additional information)
Comment	Comment space available for the developers

Tags

- Keep track of when tag value changes.

Alarms

- Keep track of when user acknowledges or resets an alarm event

Recipes

- Keep track of when user downloads or uploads recipes

Miscellaneous Resources

- User login details
  Keep track of when user login, logout or change password
- User management actions
  Keep track of when a user is added, removed or when the user properties are modified
- System actions
  Keep track of system actions (HMI Device Restart, Power On, Backup, Update, Download, enter in System Setting, open Project Manager)
- FTP actions
  Keep track of ftpGET, ftpPUT, OpenTextEditor, SaveTextEditor
- Buffer actions
  Keep track of dump and delete actions on alarms, audit or trends buffers

LogMessage Macro

In addition of that, the LogMessage macro gives the possibility to define additional events to log to the audit trail buffer.

See "LogMessage" on page 203 for additional details.

Cache Memory

Data is temporarily saved in cache memory and flushed to file system when at least one of the following conditions is true:

- temporary cache buffer is full
- an explicit dump procedure has been called
- 5 minutes cycle time has expired

Warning: data in cache memory will be lost if there is a power failure before data has been flushed to the file system.

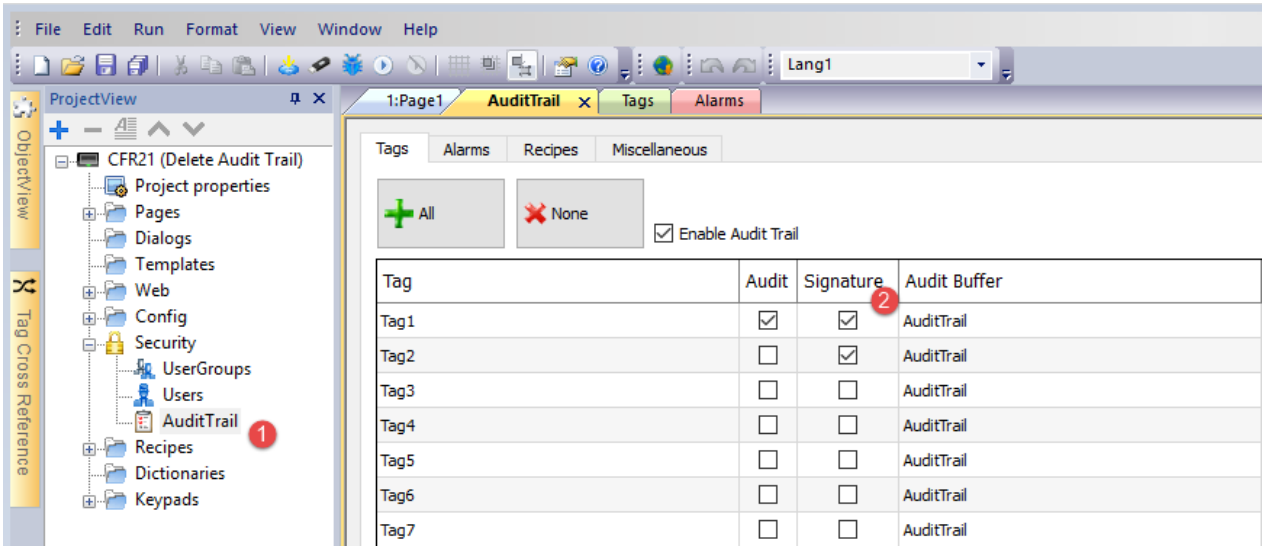
Backup audit events

From the "Events Buffer" on page 249 you can configure the size of the audit buffer and activate the backup of the audit events when the buffer is full.

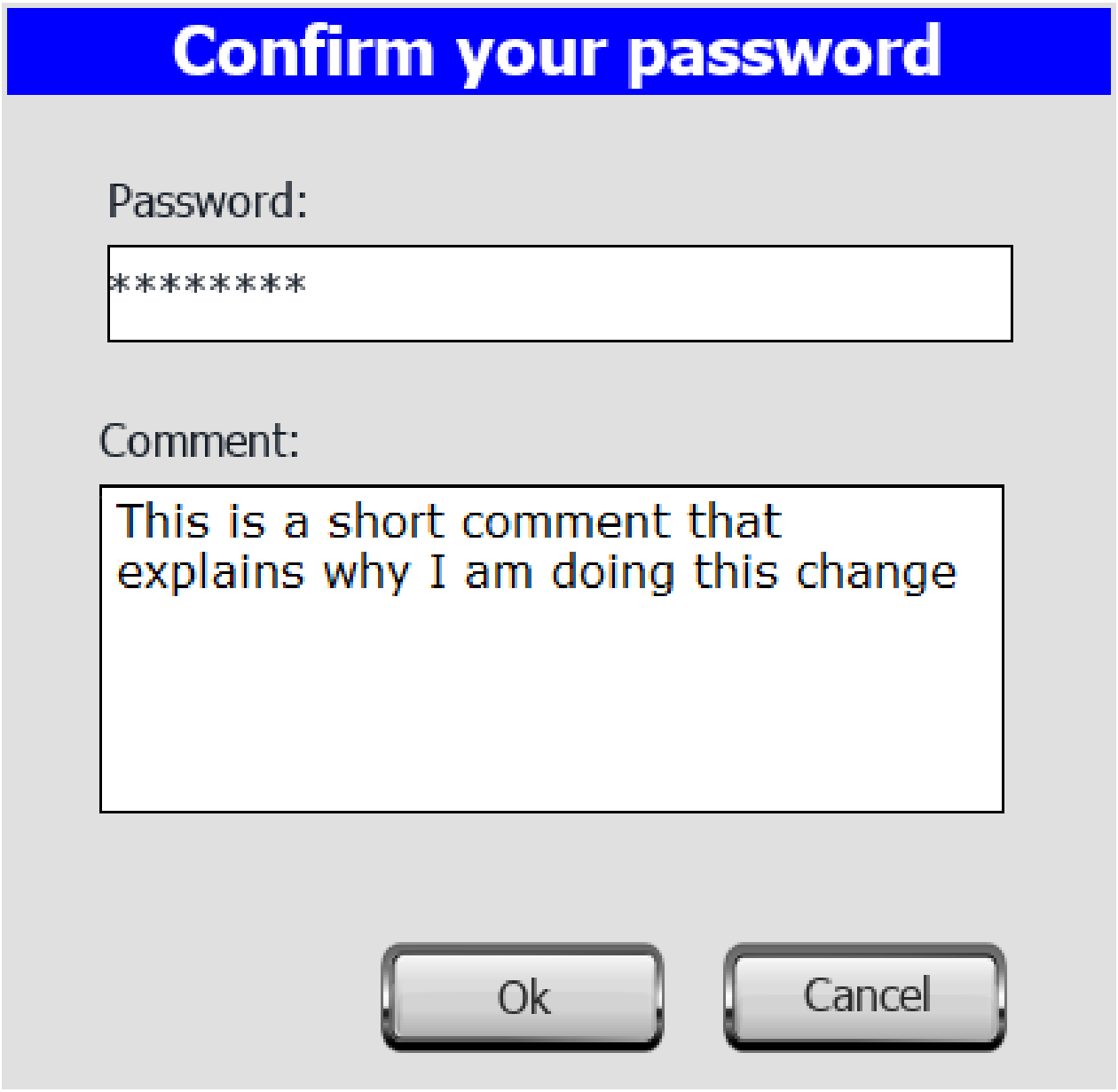
Electronic Signature

For each resources listed within the Audit Trail editor, it is possible configure the LRH SW HMI Runtime to require the password confirmation before changing it. If the audit trail log is enabled, the user has the option of adding a comment that will be recorded within the Track Log.

Path: ***ProjectView*** *>* ***Security*** *> double-click* ***AuditTrail***



The user password is required before allowing the resource to be modified by the user



The introduced password will be not required again for the commands released in the next 10 Sec. The validity time can be modified from the Settings dialog.

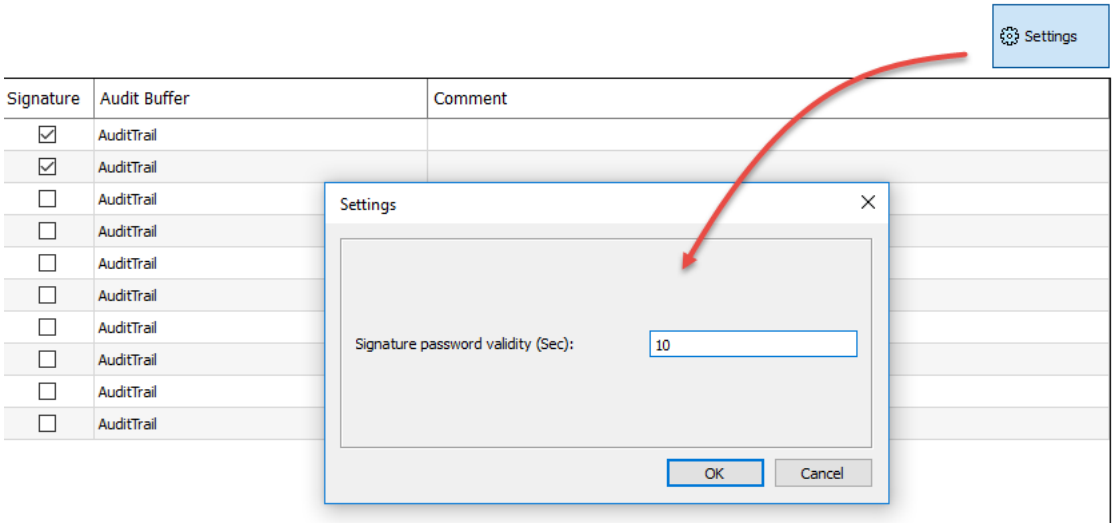
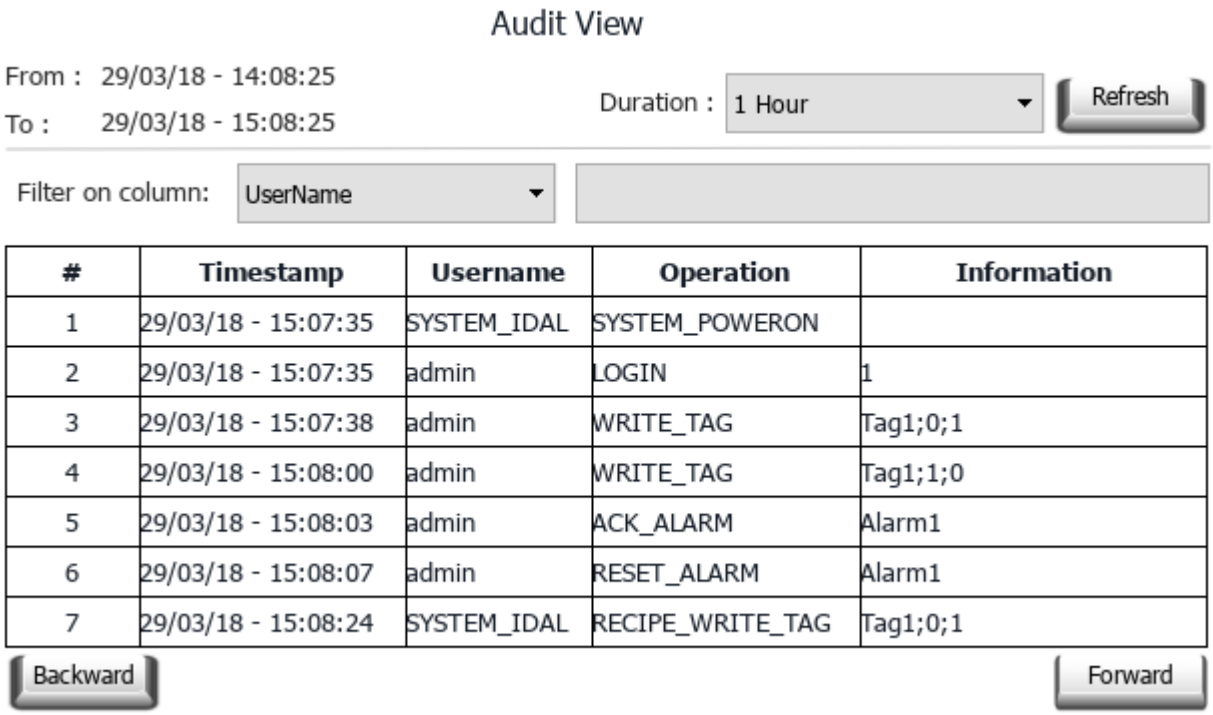


Table audit widget

Path: **Widget Gallery**> **Basic**> **Audit Tables**

Display contents of the audit trail inside a widget



#	Timestamp	Username	Operation	Information
1	29/03/18 - 15:07:35	SYSTEM_IDAL	SYSTEM_POWERON	
2	29/03/18 - 15:07:35	admin	LOGIN	1
3	29/03/18 - 15:07:38	admin	WRITE_TAG	Tag1;0;1
4	29/03/18 - 15:08:00	admin	WRITE_TAG	Tag1;1;0
5	29/03/18 - 15:08:03	admin	ACK_ALARM	Alarm1
6	29/03/18 - 15:08:07	admin	RESET_ALARM	Alarm1
7	29/03/18 - 15:08:24	SYSTEM_IDAL	RECIPE_WRITE_TAG	Tag1;0;1

Buttons:

- REFRESH
  Retrieve trend data from internal buffer and refresh table view
- BACKWARD/FORWARD
  Move the display window forward or backward as specified in the duration parameter

Filter:

Use the combo box to select the column where search for and the text filed on the right to enter the string to search to.

Parameter	Description
AuditBuffer	Event Buffer from which the event list is retrieved (see "Events Buffer" on page 249)
Heading	Heading label
Default Duration	Initial value of time window to show
Time Spec	Time format: • **Local** = show the time values of the HMI device. • **Global** = show the time values using UTC format.
Date Format	Select the Date and Time format

Parameter	Description
Filter List	Labels to show in filter column selection
Table Layout	Defines the characteristics of the scroll bar and allows to remove the header of the table

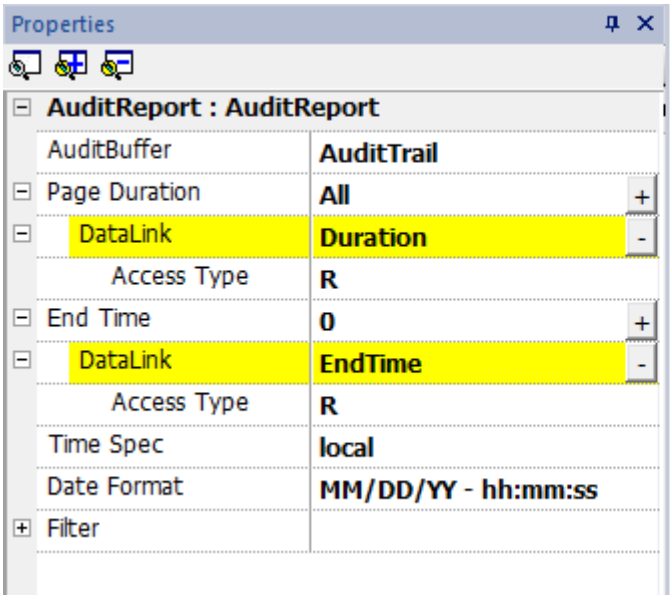
Printing audit table

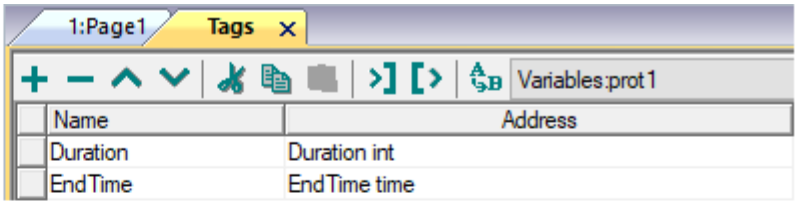
An audit table widget without buttons can be found and used from the print report gallery. The table can be drawn and enlarged to fill the entire page. If the number of lines to printed is greater of one page, the audit table will be printed using additional pages.

Using the “attach to tag” feature is possible to use tags to define some properties of the historical trend to print at runtime:

- Page Duration
- End Time

"Page Duration" with "End Time" define the piece of the audit buffer to print.





Exporting audit trail as .csv files

Data recorded inside the audit trail can be exported inside a csv file using the **SaveEventArchive** action. See "SaveEventArchive" on page 201 for details.

File structure

	A	B	C	D	E	F	G	H	I	J	K	L
1												
2	**Record ID**	**Date**	**Time**	**User ID**	**Interface**	**Action**	**Status**	**Data**				
3	1	27/03/2018	14:22:06	SYSTEM_IDAL	SYSTEM_IDAL	SYSTEM_POWERON	S_OK					
4	2	27/03/2018	14:22:06	admin	LOCAL	LOGIN	S_OK	1				
5	3	27/03/2018	14:22:08	admin	LOCAL	WRITE_TAG	S_OK	Tag1	0	1		
6	4	27/03/2018	14:22:09	admin	LOCAL	WRITE_TAG	S_OK	Tag2	0	1		
7	5	27/03/2018	14:22:26	admin	LOCAL	WRITE_TAG	S_OK	Tag2	1	5	This is a test	
8	6	27/03/2018	14:22:50	SYSTEM_IDAL	SYSTEM_IDAL	RECIPE_WRITE_TAG	S_OK	Tag1	1	1		
9	7	27/03/2018	14:22:50	SYSTEM_IDAL	SYSTEM_IDAL	RECIPE_WRITE_TAG	S_OK	Tag2	5	3		
10	8	27/03/2018	14:22:50	SYSTEM_IDAL	SYSTEM_IDAL	RECIPE_WRITE_TAG	S_OK	Tag3	0	5		
11	9	27/03/2018	14:22:50	admin	LOCAL	DOWNLOAD_RECIPE	S_OK	Recipe0	set-00			
12	10	27/03/2018	14:22:54	admin	LOCAL	ACK_ALARM	S_OK	Alarm2				
13	11	27/03/2018	14:22:58	admin	LOCAL	RESET_ALARM	E_FAIL	Alarm2				
14	12	27/03/2018	14:23:02	admin	LOCAL	DUMP_AUDIT_BUFFER	S_NEEDNOT_NOTIFY	AuditTrail				
15												
16												
17	**Record ID**	**Date**	**Time**	**User ID**	**Interface**	**Action**	**Status**	**Data**				
18	13	27/03/2018	14:23:24	admin	LOCAL	DELETE_AUDIT_BUFFER	S_OK	AuditTrail				
19	14	27/03/2018	14:23:26	SYSTEM_IDAL	SYSTEM_IDAL	RECIPE_WRITE_TAG	S_OK	Tag1	1	2		
20	15	27/03/2018	14:23:26	SYSTEM_IDAL	SYSTEM_IDAL	RECIPE_WRITE_TAG	S_OK	Tag2	3	4		
21	16	27/03/2018	14:23:26	SYSTEM_IDAL	SYSTEM_IDAL	RECIPE_WRITE_TAG	S_OK	Tag3	5	6		
22	17	27/03/2018	14:23:26	admin	LOCAL	DOWNLOAD_RECIPE	S_OK	Recipe0	set-01			
23	18	27/03/2018	14:23:27	user1	CGI	LOGIN	S_OK	192.168.49.242				
24	19	27/03/2018	14:23:37	user1	CGI	WRITE_TAG	S_OK	Tag1	6	55		
25	20	27/03/2018	14:24:28	admin	LOCAL	DUMP_AUDIT_BUFFER	S_NEEDNOT_NOTIFY	AuditTrail				
26												

Exported data file has the following content	
RecordID	Each record is stored with a progressive number which will give the possibility to easily identify missing records or confirm that they are not lost. Note that the progressive number is not reset to zero when the buffer is deleted.
Date, Time	Event time stamp. Time can be configured as local or global from the dump action.
User ID	User that perform the operation
Interface	LOCAL: when the action is performed in the HMI device CGI: when the action is performed by a remote client. SYSTEM_IDAL: when the action is performed from the LRH SW HMI Runtime application
Action	Action executed.
Status	Result of the executed action • S_OK Action executed correctly • E_FAIL Action non executed • S_NEEDNOT_NOTIFY Action triggered (will be executed asynchronously)
Information	Additional info related with the executed action.

29 Reports

A report is a collection of information that will be printed when triggered by an event. When the programmed event is triggered, the printing starts in background.

You can configure reports, their contents, trigger conditions and output printer in the Reports editor.

Not all widgets can be used in reports. When configuring reports, LRH SW provides access to a dedicated widget gallery featuring only widgets available for reports.

Reports format can be customized using predefined templates for page layout.

Note: Report printing is not supported by LRH SW Client.

Adding a report

Path: ***ProjectView*** *>* ***Config*** *> double-click* ***Reports***

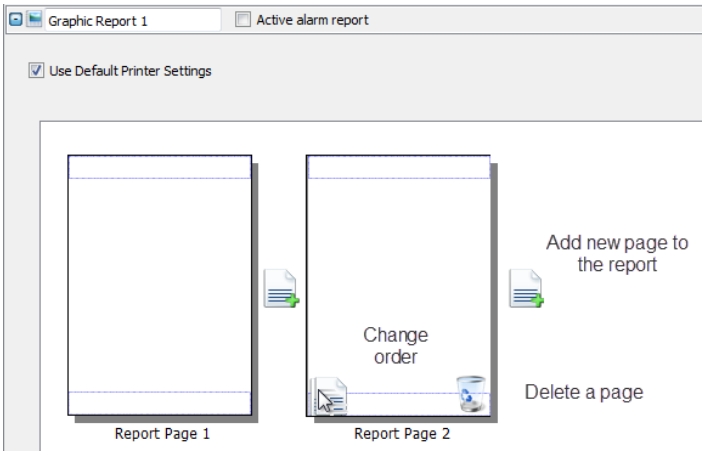
In **Reports** editor, click **Graphic Report**: one new row is added to the table.

Report types

Report type	Description
Graphic Reports	Contain graphical elements and may include complex widgets such as screenshots or alarms. **Important: Each printer requires a specific printer driver. See "Configuring graphic reports" below for a list of supported printer drivers.**

Configuring graphic reports

Use the **Report** editor to configure graphic reports.



Adding a report page

Click **+** to add a new page to the report layout.

When the mouse goes over a page, two icons are displayed and allow you to reorder or delete the pages.

Modifying report page content

1. Double click on a page to edit its content: the **Graphic Report** editor appears.

   Each page is divided in: header, footer and page body.

2. Double click on the area you want to edit: the edit area is shown in white, others are grayed out.

The Widget Gallery is context-sensitive and displays only the widgets available for the area you are editing.

Widgets available for reports

Widgets that can be used for a graphic report:

Widget	Function
Page Number	Automatic page numbering
Screenshot	Screen capture of the page currently displayed by the HMI device. The report page is automatically resized to fit the HMI device page. Note: The full screen is printed, including all open dialogs.
Alarm	Entire contents of the event buffer (default buffer is Alarm Buffer1).
Text	Widgets such as labels and numeric fields

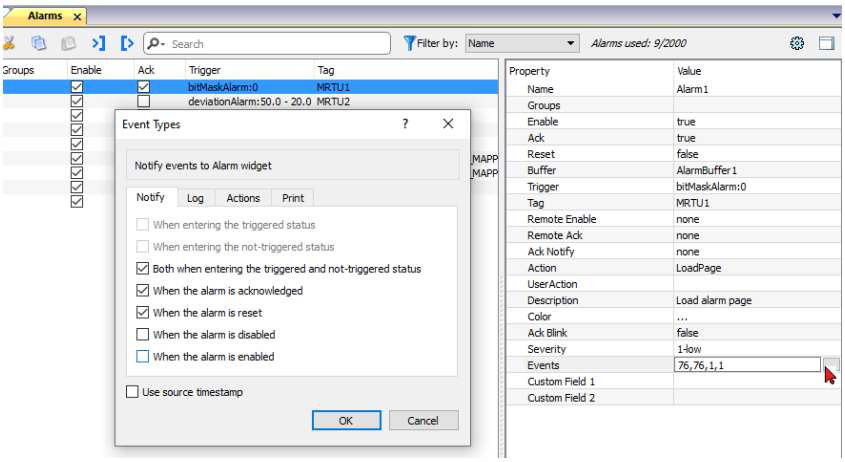
Print triggering events

Report printing can be triggered by events.

Configuring alarm printing

*Path: **ProjectView**> **Config** > double-click **Alarms***

1. In the Alarms editor, open the **Event Types** dialog from the **Events** property.
2. In **Print** tab select all the conditions for which you want to trigger printing.

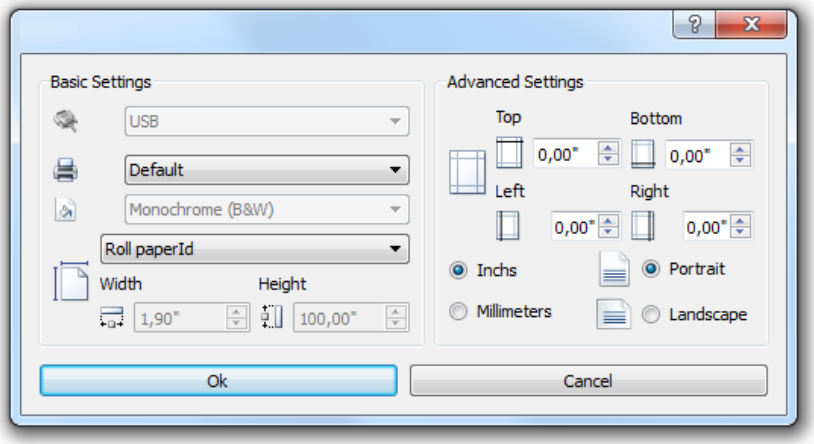


Important: Only one report can be set as Active alarm report in a project and it can be either a text report or a graphic report.

Adjusting printer settings at runtime

A graphic report printing can be started also using the action **PrintGraphicReport**.

Set the action property **silent** to **false** to have a pop-up dialog.



Default printer

Printer setting

You can set a default printer for all graphic reports. Each report can then be configured to use the default printer or any other printer available. Click **Printer Setting** button to set printer parameters.

For PDF printers you also define the folder where files are saved by using **Printed Files Location**.

30 Screen saver

Screen saver can be used to execute actions and/or display a slide show when the HMI device is not in use. The screen saver starts after a timeout if none of the following events occur:

- touch of display
- mouse movement
- external keyboard key pressed
- active dialogs

When the display is touched or a mouse movement is detected or a key from an external keyboard is pressed or a dialog is launched, if the screen saver is active it is deactivated.

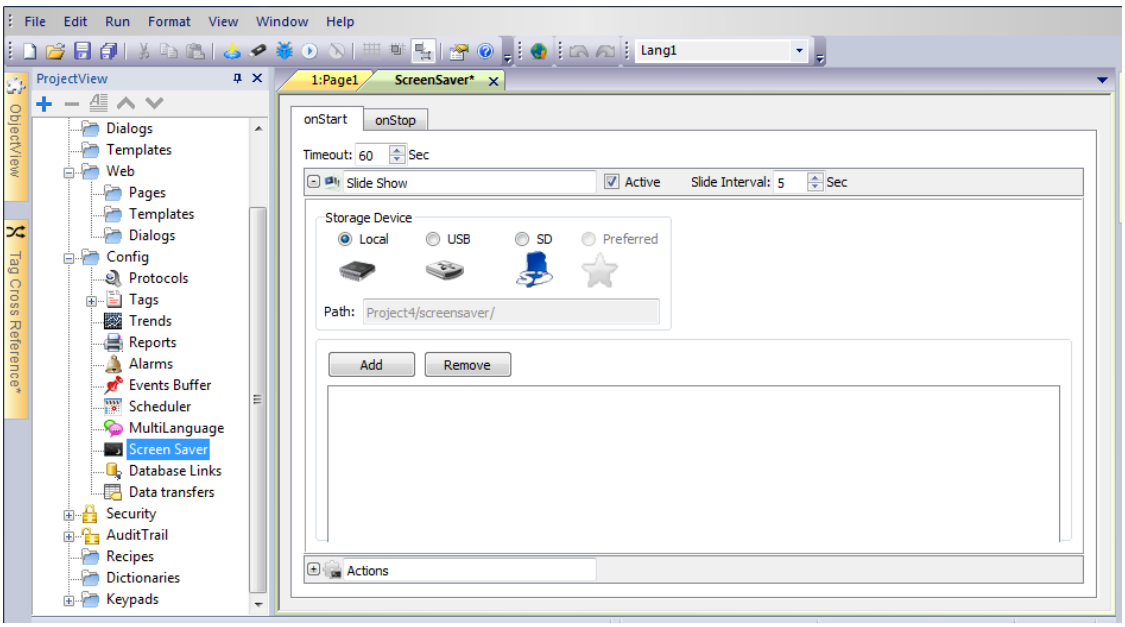
Enabling the screen saver function

Path: ***ProjectView*** *>* ***Config*** *> right-click* ***Screen Saver*** *>* ***Enable***

Important: You must enable the screen saver before you can configure it.

Configuring a screen saver

Path: ***ProjectView*** *>* ***Config*** *> double-click* ***Screen Saver***



Slide show parameters

Parameter	Description
Timeout	Time after which the slide show starts
Slide Interval	Interval between slides
Storage Device	Location of the images used in the slide show. Images stored locally are saved in *workspace\projectname\screensaver* and can be downloaded to the HMI device when the project is downloaded. Images stored on USB or SD devices are saved in a screensaver folder on the device itself. **Important: Only JPEG and PNG images are supported.**

Associating actions to the screen saver

Actions can be triggered by the screen saver start and/or stop.

- Click **+** next to **Actions** in the **onStart** tab to configure actions to be executed when the screen saver starts.
- Click **+** next to **Actions** in the **onStop** tab to configure actions to be executed when the screen saver stops.

31 Backup/restore of Runtime and project

You can backup all the content of the HMI device, including

- LRH SW HMI Runtime
- HMI Application Project

to an external memory. This backup copy can be used to restore the content of the HMI device at a later time or copy it to a new HMI device.

The backup function is available only if enabled for the logged user. See "Modifying access permissions" on page 327 for details.

Note: Backup is not supported in LRH SW Client.

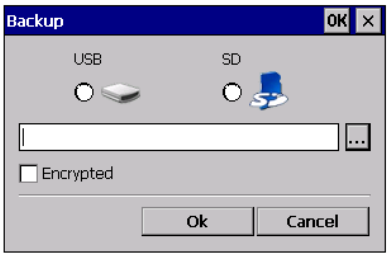
Backup function

The backup function automatically performs the following procedure:

1. Unloads the current project to unlock files in use.
2. Archives the content of the \QTHMI folder (containing LRH SW HMI Runtime, projects, dynamic files such as recipes, alarms, trends and so on) to a .zip file (standard or encrypted).
3. Reset the HMI device (reloads the project).

To start the backup procedure:

1. In LRH SW HMI Runtime right click to open the context menu.
2. Select **Backup:** the **Backup** dialog is displayed.



3. Select the path for storing the backup file.

Note: The backup process does not include files stored in USB and SD cards. Dynamic data such as recipes, trends, events stored in these devices will not be included in the backup.

Restore function

Restore the backup package can be perform on HMI device

- from the Context Menu (see "Update package" on page 95 for details)
- or from the System Settings (see "System Settings" on page 543 for details)

32 Keypads

Several keypads are provided by default in the LRH SW so that they can be used for data entry.

The alphabet keypad can be use associate with a string data type

The numeric keypad can be use associate with a numeric data type

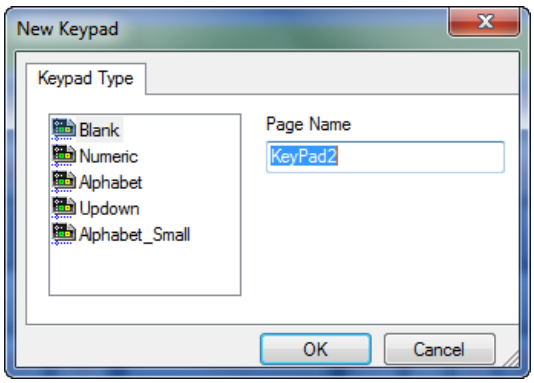
The calendar keypad can be use associate with a date data type

Creating and using custom keypads

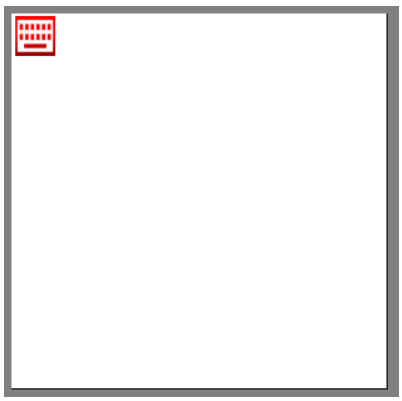
You can either create a new keypad or customize an existing one.

Creating a keypad

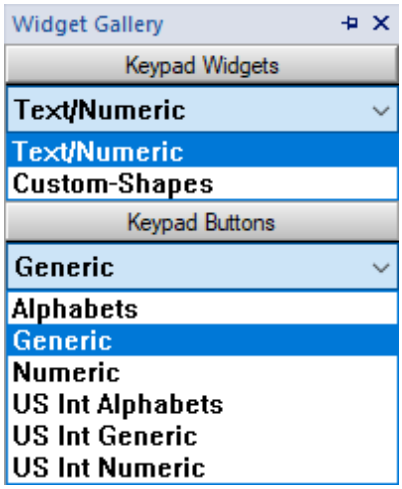
1. In **ProjectView**, right-click **Keypads** and select **Insert Keypad**: the **New Keypad** dialog is displayed.



2. Select one of the available keypads, or **Blank** to create a keypad from scratch. In this case a blank keypad is displayed.



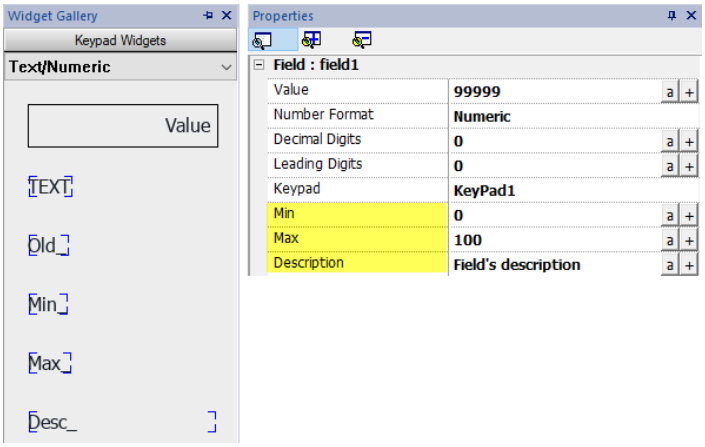
3. Use the **Keypad Widgets** and **Keypad Buttons** from the Widget Gallery to create your custom keypad.



The keypad you create, as in this example, will be saved in the project folder.

Text/Numeric Controls

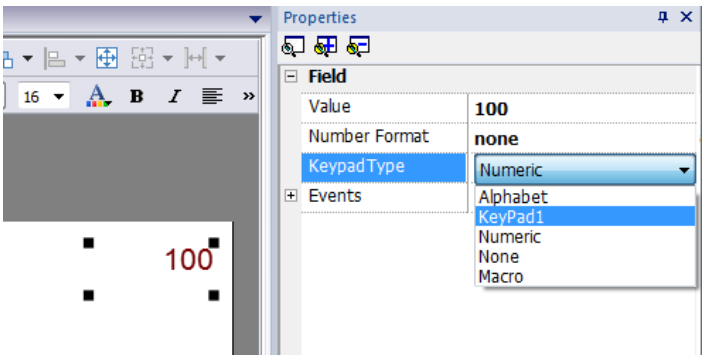
The Text/Numeric folder contains some specific controls to keypad development.



Data source	Description
TEXT	Simple text label
Old_	Current value
Value	New value
Min_	Min value defined inside the field's property currently editing.
Max_	Max value defined inside the field's property currently editing.
Desc_	Description defined inside the field's property currently editing.

Attaching custom keypads to fields

Custom keypads can then be reused for any field where the **Keypad** property points to it as in this example.



Tips and tricks with custom keypads

By default, any numeric widget (read/write numeric field) are assigned the numeric keypad.

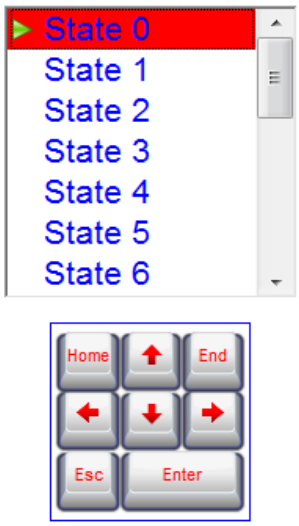
If you want to apply a customized version of the numeric keypad to all the numeric widgets you add to your project proceed as follows:

1. Create a new keypad and select **Numeric** as **Keypad** type. This will be a backup of the original settings for the numeric keypad.
2. Customize the default numeric keypad and save it. This customized version of the numeric keypad will now be assigned as default in the project.

See "Deleting or renaming custom keypads" below for details on how to rename a custom keypad.

Up-down arrows keypad

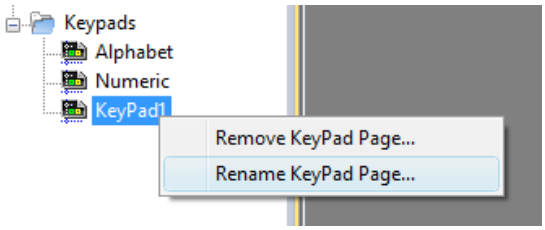
This type of keypad is particularly useful to move the cursor up and down within widget requiring this functionality. Here an example using a **Control List** widget. See "Control list widgets" on page 408 for details.



Deleting or renaming custom keypads

In **ProjectView**, right-click on a custom keypad and select one of the options:

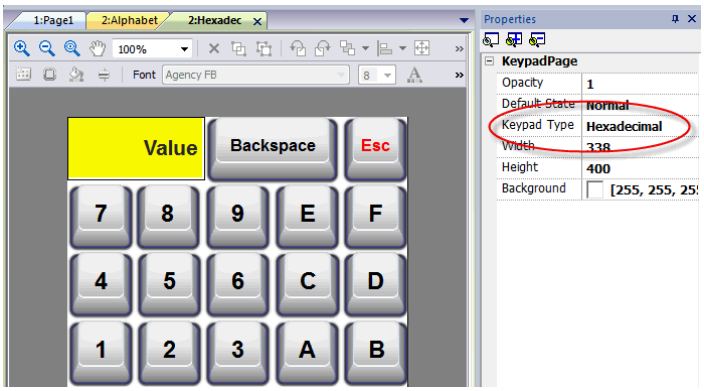
- **Remove KeyPad Page** to remove the keypad from the project
- **Rename Keypad Page** to rename the keypad.



Keypad type

Path: ***ProjectView*** *>* ***Keypads*** *> double-click a keypad >* ***Properties***

Set **Keypad Type** parameter for a keypad to define the type of data entry.



Keypad Type	Description
Auto	Default setting
Decimal	Only numeric keys are accepted. Entering 10, the keypad returns 10 that will be displayed as "10" if the attached field is numeric or ASCII, as 'A' if the attached filed is hexadecimal.
Hexadecimal	Only hexadecimal keys are accepted. Entering 10, the keypad returns 16 that will be displayed as "16" if the attached field is numeric or ASCII, as "10" if the attached field is hexadecimal.
Ascii	All keys are enabled. Entering 1A, the keypad returns 1A that will be displayed as '1' if the attached field is numeric, as "1A" if the attached field is ASCII or if the attached field is hexadecimal.

Keypad position

Runtime Positioning property of keypads can be used to define where keypads will appear in the screen.

Option	Description
Automatic	The best position is selected according to here data entry is required.
Absolute	X,Y coordinates are entered to identify the exact position
Left-top **Left-center** **Left-bottom** **Center-top** **Center-center**	Predefined screen positions

Option	Description
Center-bottom	
Right-top	
Right-cente	
Right-bottom	

Select the **Lock Keypad position** option if you do not want the keypad to be moved by dragging.

33 External keyboards

LRH SW HMI Runtime has been designed to work with external keyboards connected via USB.

Keyboards can be used for:

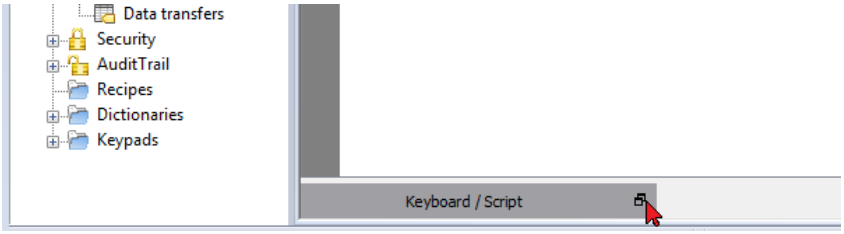
- data entry (default)
- execute actions mapped on specific keys

For example, the right arrow key **OnClick** event can be mapped to the **LoadPage** action.

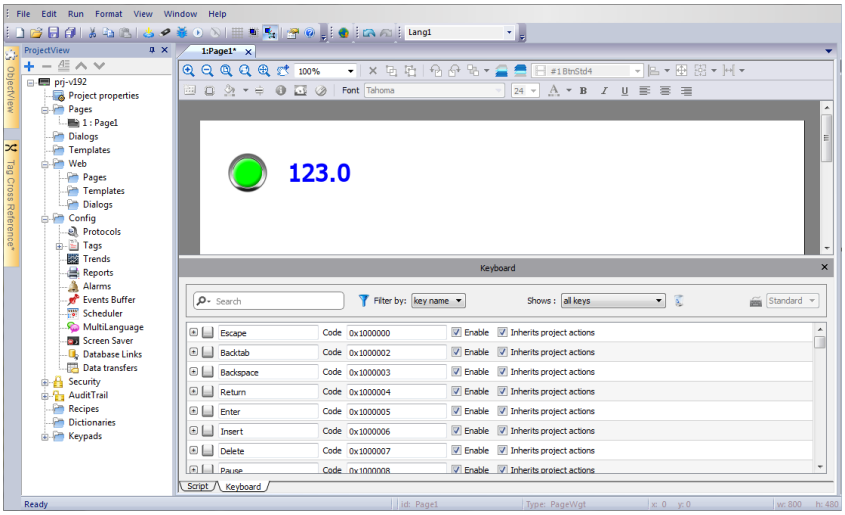
Keyboard can be programmed at project level so that settings will be inherited by all the pages. In each page you can then choose which key setting will be inherited from the project and which one you will customize for the specific page.

Opening external keyboards

1. In the Page Editor, click on the icon on the right of **Keyboard/Script** at the bottom of the workspace: the Keyboard/Script Editor is displayed.
2. Select the **Keyboard** tab.



Each row in the Keyboard Editor corresponds to a key.



For each key, the following information is displayed:

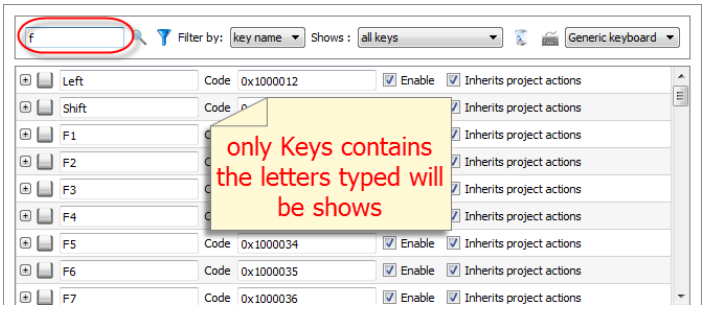
Element	Description
Label	Key name
Code	Key code
Enable	Key enable status
Inherits project actions	Defines whether the key is inheriting the action programmed at the project level

Here the possible configurations:

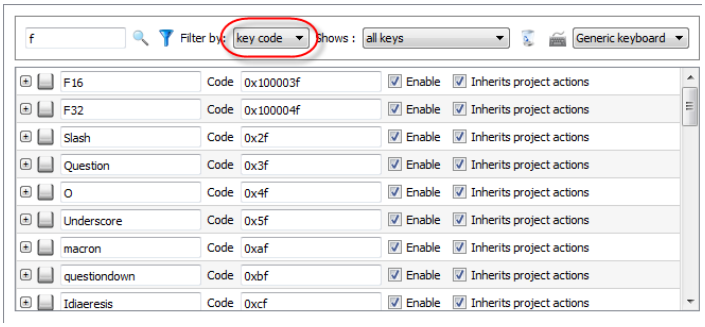
Enable	Inherits project actions	Editor appearance	LRH SW HMI Runtime behavior
Checked	Unchecked	Action lists show the page actions (or nothing if the list is empty)	Only the page actions (if any) will be executed.
Checked	Checked	Action lists show the project actions only and cannot be edited	Only the configured project actions (if any) will be executed.
Unchecked	Checked	Inherits project actions check box and all action lists are disabled. Action lists show the project actions only.	No page or project action will be executed.
Unchecked	Unchecked	Inherits project actions check box and all action lists are disabled. Action lists show the project actions only.	No page or project action will be executed.

Search and filter

To display a filtered set of keys, in **Filter by** select **key name** and type a letter in the search field: only the keys containing that letter in their name will be displayed in the Keyboard editor.



Alternatively, in **Filter by** select **key code** and type a letter in the search field: only the key containing that letter in their code will be displayed in the Keyboard editor.



Displayed keys

You can easily select what keys will be listed in the Keyboard editor window. To display a limited set of keys, select an option in **Shows**.

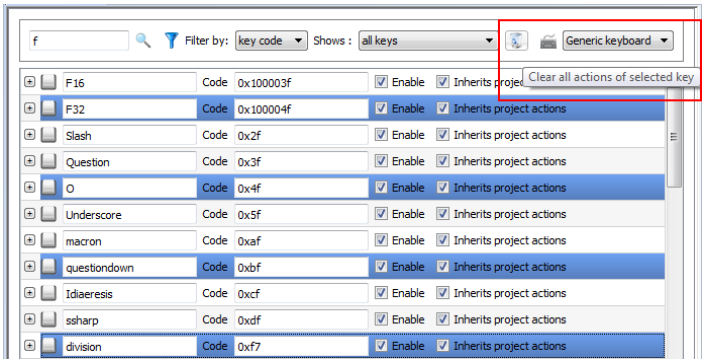
Option	Description
all keys	All keys available in the keyboard layout are listed
modified keys	Only the keys associated with actions at the page level are listed
modified keys in project	Only the keys associated with actions at project level are listed

Removing action associations

To remove all the associations you created between keys and actions:

1. Select the keys for which you want to remove the association.
2. Click the **Clear all actions of selected keys** button.

If you are working at page level, page actions will be removed, if you are working a project level, project actions will be removed.

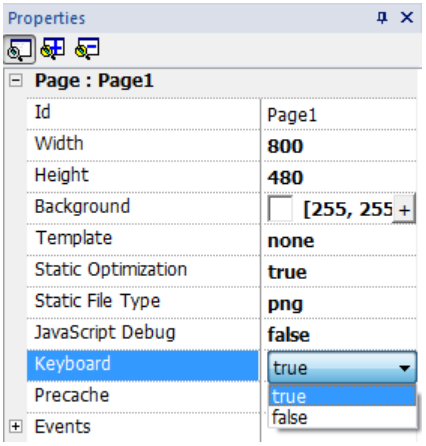


Keyboard layout

Select the layout of the keyboard from the **Keyboard Layout** combo box. **Generic Keyboard** refers to a generic international keyboard layout.

Enable/disable keyboard

You can enable/disable keyboard actions both at project and at page level. To enable keyboard actions, in the **Properties** pane set **Keboard macro** to **true**.

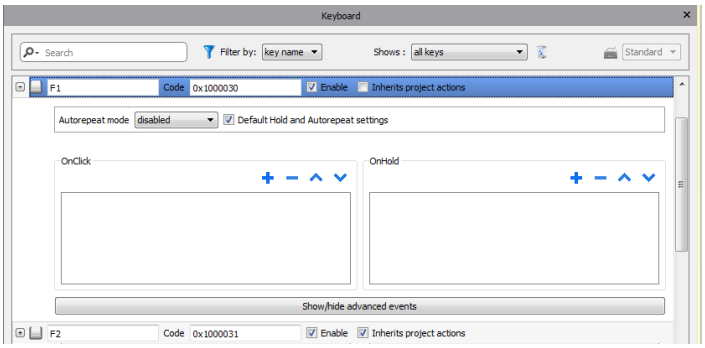


You can enable/disable keyboard actions also at runtime using the KeyboardMacros action. See "Keyboard actions" on page 175 for details.

Associating actions to keys

You associate actions to a keys from the Keyboard editor.

1. Click **+** next to the key you want to program: the fields for key configuration are displayed.

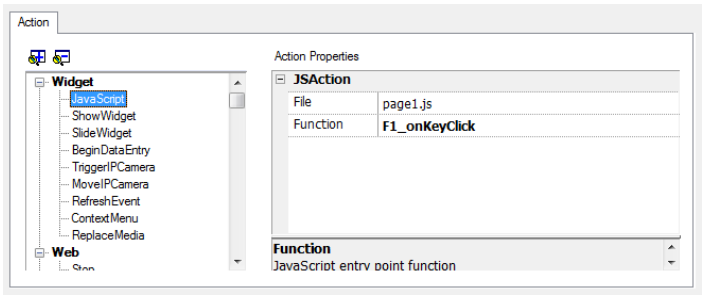


1. Click **+** to add actions.

You can associate actions both to the **OnClick** event and toe the **OnHold** event.

See "Events" on page 49 for details.

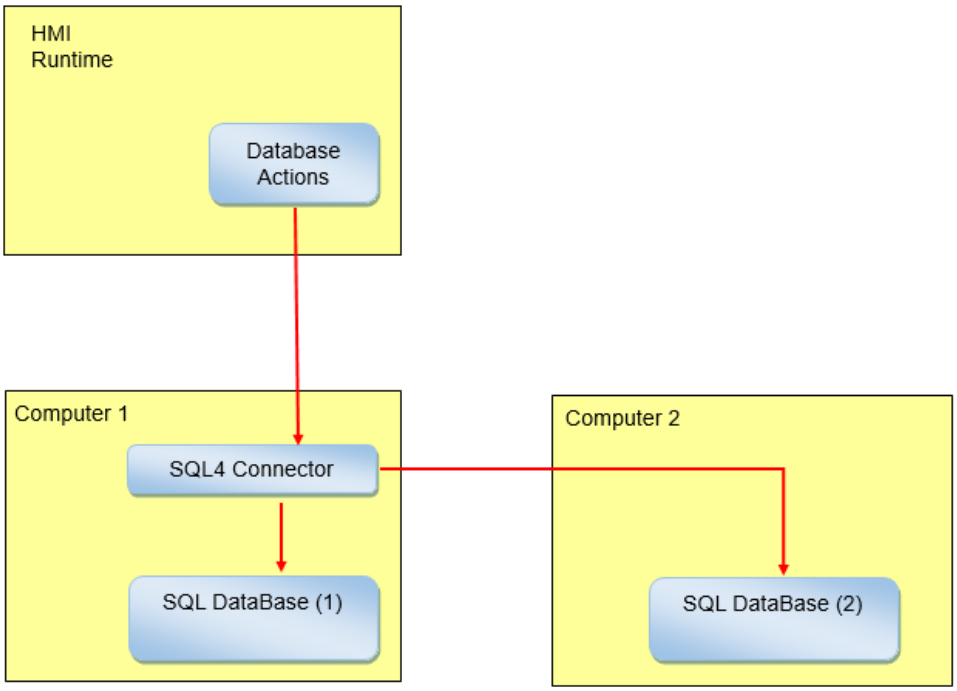
Note: Also JavaScript code can be associated to a key event.



34 Storing data to external databases

LRH SW allow to connect to the SQL4Automation Connector, a software solution for the industrial usage. It connects HMI, PLC and robotic controls directly with SQL databases. HMI directly access SQL databases via the connector and can query data from tables, insert, change and delete data in tables by using SQL commands [structured query language].

The database site communicates by ODBC. Therefore all SQL databases can be integrated, which support an ODBC interface. The SQL syntax needs to be adapted to the given database, e.g. MS SQL Server, mySQL, MS Office Access, SQLite, Oracle, PostgreSQL...



To store data into an external database:

1. Install the SQL4Automation tool on the computer hosting the database or in a computer between the HMI device and the database.
2. Configure the SQL4Automation tool.
3. Create a project that use the dedicated DB actions to access at the external database.

Installing SQL4Automation

Download the latest version of SQL4automation and install it on the computer. Refer to www.sql4automation.com for details and download.

Procedures described in this document refer to SQL4Automation Connector Version 3.3.2.0

Configuring SQL4Automation

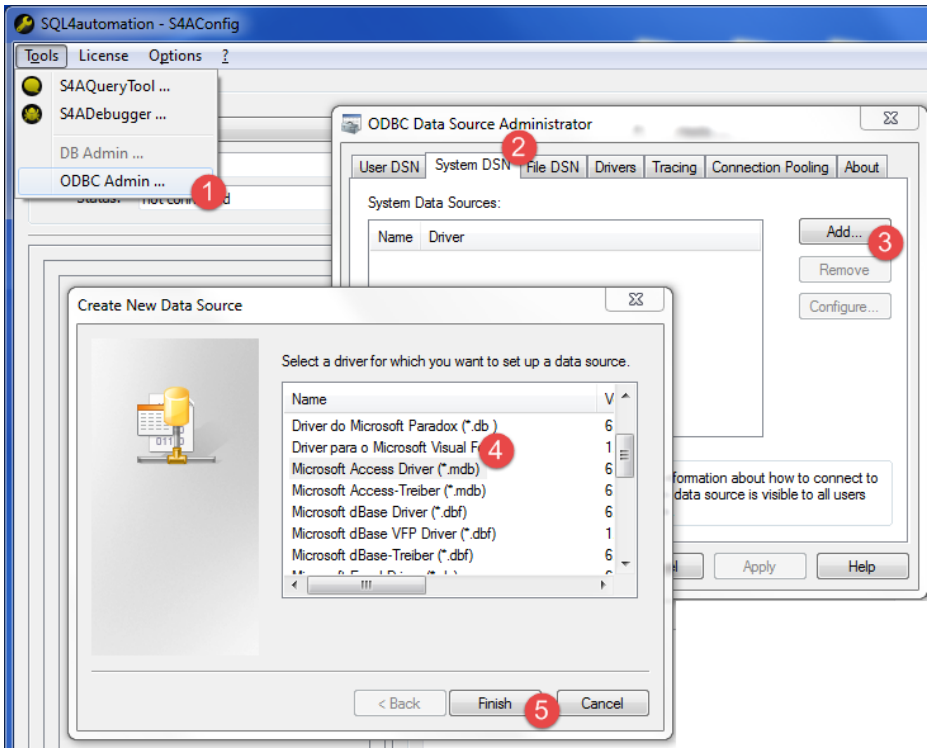
Important: Refer to the SQL4Automation user manual for detailed configuration instruction.

Here is a quick description of how to access to a MS Office Database (MS Access).

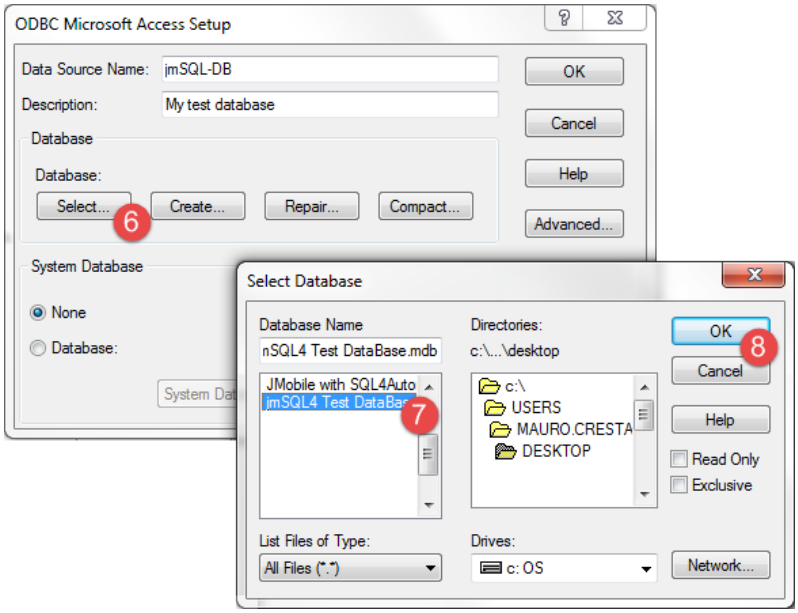
You must have the MS Office Suite installed on a computer and create an empty database using Microsoft Access.

Start SQL4Automation and follow the procedure to configure your SQL4Automation Connector:

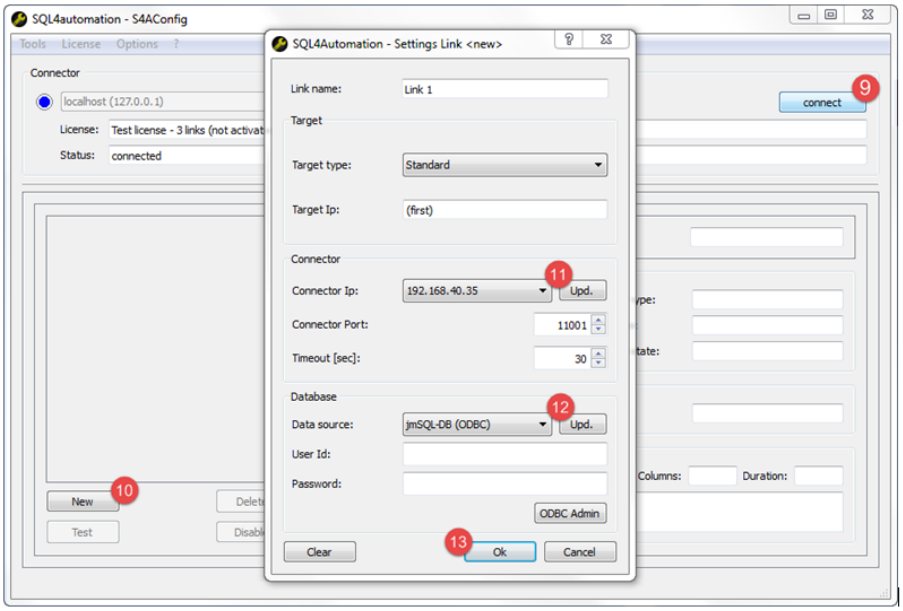
1. Select **ODBC Admin**: the **ODBC Data Source Administrator** dialog is displayed.
2. Select the **System DSN** tab.
3. Click **Add**: the **Create New Data Source** dialog is displayed.
4. Select the Microsoft Access Drive
5. Click **Finish** to confirm.



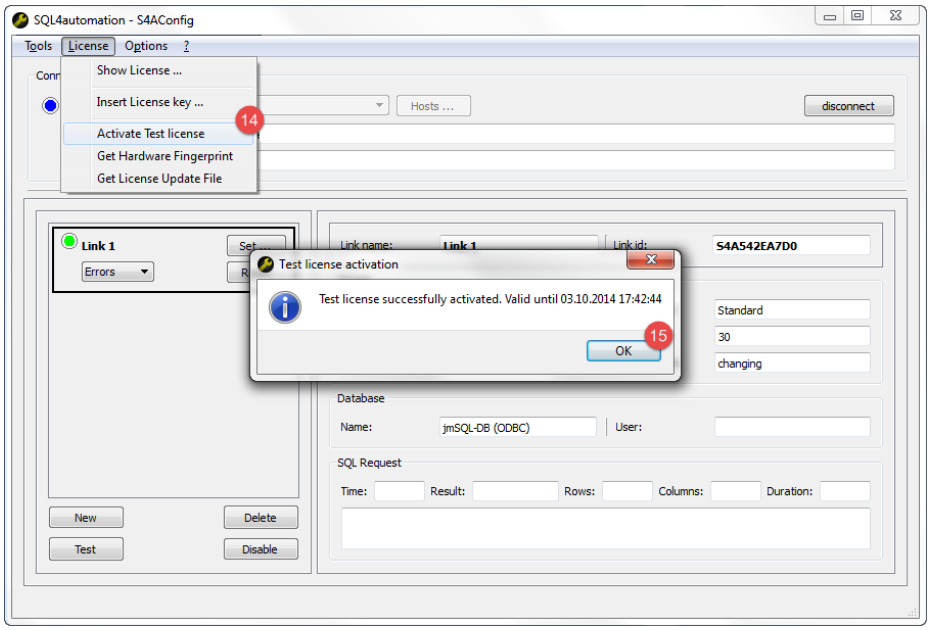
6. Enter **Data Source Name** and **Description** then click **Select**: the **Select Database** dialog is displayed.
7. Select your Access database.
8. Click **OK** to confirm.



9. At the fist connection, click **New** to select your Data Source
10. Select the IP address of your computer. This will be the connection IP Address used from your HMI device.
11. Select the Data Source.
12. Click **OK** to confirm.
13. Click **Connect**.



14. Select **License> Activate Test License**: when the **Link 1** led turns green the procedure has been completed correctly.
15. Click **OK** to confirm.

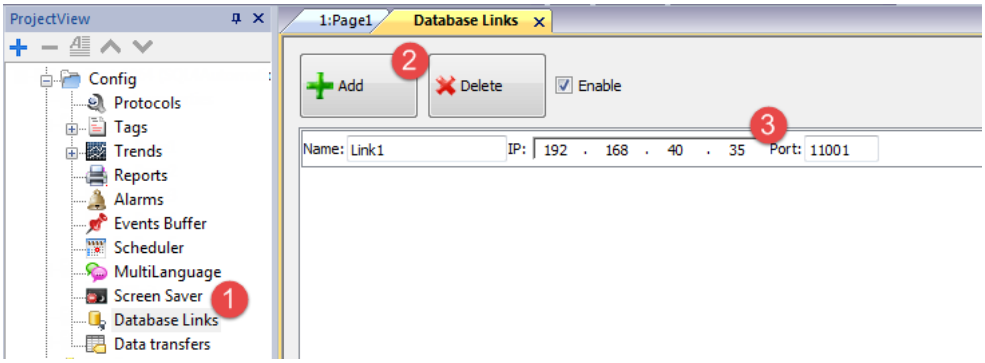


Configuring the HMI project

Path: ***ProjectView*** *>* ***Config*** *> double-click* ***Database Links***

To save a project data to an external database you need to create a link with the specific database

1. In the **Database Links** editor select **Enable** to enable the function.
2. click **Add** to create a new link.
3. Enter the IP Address the computer hosting the SQL4Automation Connector.



Important: The link name here is not necessarily the same defined inside the SQL4Automation Connector. But this is the name to be used in all actions using the remote database.

Transfer data with JavaScript

Some actions used to transfer data from a HMI device to a remote database can be used as macros inside a JavaScript code as in the example below.

Status of database connection is available through system variable tags. See "Database variables" on page 1.

Error status can be reset with actions. See "Database actions" on page 171

```
function myButton1_onMouseClick(me, eventInfo) {
     var CustomSQL = '' ;
     var DatabaseLink ='Link1';
     project.dbInit(DatabaseLink, CustomSQL);
};

function myButton2_onMouseClick(me, eventInfo) {
     var CustomSQL = '' ;
     var DatabaseLink ='Link1';
     var Tags ='Alarm1;SystemTime;Tag01;Tag02;';
     project.dbReadTags(DatabaseLink, CustomSQL, Tags);
};

function myButton3_onMouseClick(me, eventInfo) {
     var CustomSQL = '' ;
     var DatabaseLink ='Link1';
     var Tags ='Alarm1;SystemTime;Tag01;Tag02;';
     project.dbWriteTags(DatabaseLink, CustomSQL, Tags);
};
```

dbQuery

```
project.dbQuery(databaseLink, customSQL, dbCallback);
```

Using this query you can execute SQL Queries.

Parameter	Description
databaseLink	Link to the database to use
customSQL	String with the SQL query
dbCallback()	Function that will be call when query data are ready

dbCallBack

```
project.dbCallBack(dbStatus, dbResponse);
```

Parameter	Description
dbStatus	0: no error found
dbResponse	Query response. Table column names followed by its rows: In the example: TagnName - Tagvalue Tag09 - 103 Tag10 - 302

Script

```
function JS1_onMouseClick(me, eventInfo) {

    var customSQL = "SELECT Tagname, Tagvalue FROM Tags WHERE Tagname='Tag09' OR Tagname='Tag10' ORDER BY Tagname"
    var databaseLink ="Link1";
    project.dbQuery(databaseLink, customSQL, dbCallback)
};


function dbCallback(dbStatus, dbResponse){

    alert("SQL Answer = " + dbResponse + "\ndbStatus =" + dbStatus);
};
```

Database tables

Here the structure of the database tables used by the database actions.

Note: These tables can be generated on an empty database from the **DBInit** action.

Table: Tags

FieldName	Text(255)	PRIMARY KEY
TagValue	Text(255)	

Table: Trends

Id	Long Integer	PRIMARY KEY
TrendName	Text(255)	
SampleTime	Text(255)	
TrendValue	Text(255)	
Quality	Text(255)	
RefreshTime	Text(255)	

Table: Recipes

Recipe	Text(255)	PRIMARY KEY
SetName	Text(255)	PRIMARY KEY
ElementName	Text(255)	PRIMARY KEY
SetValue	Text(255)	

Table: Event

Id	Long Integer	PRIMARY KEY
EventName	Text(255)	
SampledTime	Text(255)	
EventType	Text(255)	
EventSubTime	Text(255)	
EventValue	Text(255)	

Custom tables

SQL queries released from the DB actions are listed inside the project file config\dbconnector.xml.

Modify the commands defined inside this file to customize the SQL strings released from the DB actions and then get access to a different structured database.

Example

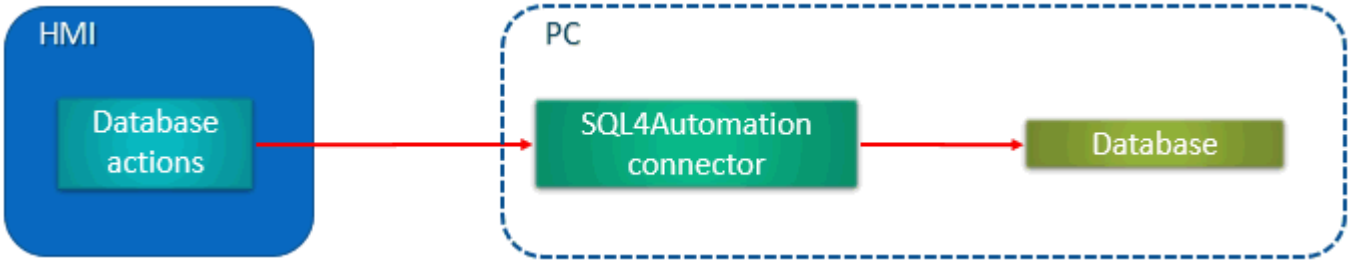
```
CREATE TABLE myTagsTable (tagname VARCHAR(255) PRIMARY KEY, tagvalue VARCHAR(255))
UPDATE myTagsTable SET Tagvalue= '%_JMV' WHERE Tagname= '%_JMT'
INSERT INTO myTagsTable (Tagname, Tagvalue) Values ('%_JMT', '%_JMV')
```

Where "%_JMV" will be replaced with the tag value and "%_JMT" with the tag name.

Connection Limits

SQL4Automation is delivered as a USB dongle with a license for a predefined number of connections. Connections are called "Links" inside SQL4Automation Connector. The number of connections depends on the license you have purchased.

SQL4Automation Connector can be installed on the same Computer/Server running the databse:



Or onto a separated Computer/Server:

35 OPC UA Server

Path: ***ProjectView*** *>* ***Config*** *>* ***Interfaces*** *> double-click* ***OPC UA***

Use OPC UA Server to publish data according to the OPC UA standard.

Parameter	Description
Enable OPC UA Server	Main flag to activate OPC UA Server. Data values defined in the HMI device are published by the OPC UA Server.

Features

Parameter	Description
Enable alarms	Activates publication of real-time alarms data (active alarms).
Enable historical alarms	Activates publication of historical alarms data.
Enable trends	Activates publication of trends data.
Tag groups	Only tags belonging to selected groups will be available to the OPC UA Server.
Alarm groups	Only alarms belonging to selected groups will be available to the OPC UA Server.

Network

Parameter	Description
Node Name	Enter node name or leave empty to use host name.
Port	Port number of OPC UA Server. Port number proposed as default may be different from port used by OPC UA Client.

Authentication

Select authentication options for OPC UA Server.

OPC UA Clients will be responsible for choosing, from available options, the most appropriate option to use according to their capabilities.

User authentication

Parameter	Description
Anonymous	Anonymous clients accepted.
User/Password	Authentication with user name is accepted. Any valid user has unrestricted access to OPC UA Server (see "Configuring users" on page 333).

Using x.509 Certificates

OPC UA provides a secure communication channel using digital certificates. Configurable levels of end-to-end security ensuring encryption, confidentiality and integrity of each message are available. When enabled, the server validates the client certificate and vice versa.

OPC UA Clients will be responsible for choosing, from available options, the most appropriate option to use according to their capabilities.

Security Mode	Description
None	Connection without certificate is allowed. Not recommended in public networks.
Sign	OPC UA Client must provide its own certificate: communication through signed messages is allowed.
SignAndEncrypt	OPC UA Client must provide its own certificate: communication through signed and encrypted messages is allowed.

Security Policy	Description
Basic128Rsa15	Accepted encryption level (used only when Security Mode is active).
Basic256	Accepted encryption level (used only when Security Mode is active).
Basic256Sha256	Accepted encryption level (used only when Security Mode is active).

Parameter	Description
Automatically trust any new clients	All certificates provided from any OPC UA Clients are accepted.
Trusted Certificates	Only OPC UA Clients that provide one of the listed certificates are accepted. To add a new certificate to the list of trusted certificates, you must have the certificate file supplied by the owner of the OPC UA Client device. Both binary and ASCII certificate file formats are accepted.

Global Discovery Server

OPC UA Server is compatible with the GDS Push Model. This means that you can use a remote GDS tool for central certificate management.

To be able to successfully connect to OPC UA Server, you must retrieve the certificate of the GDS tool and add it to the trusted certificate list of OPC UA Server.

Example

When an OPC UA Client attempts a connection with the OPC UA Server, the server checks if the client certificate is available inside its own trusted certificate list. If it is not found, the communication will be rejected and the certificate will be stored in a list of unreliable certificates.

Using a GDS tool, you can connect to the OPC UA Server, inspect available certificates and define trust or not trust state of each certificate.

Certificate Files

HMI device will store certificates inside the subfolders of folder *"/workspace/<ApplicationName>/config/pkiserver"*

- own
  Own certificate and private key
- trusted
  Trusted self signed certificates and CA certificates
- rejected
  Rejected certificates
- issuers
  Trusted intermediate (not directly trusted) CA required to validate the trust chain

Server Identity

Parameter	Description
Manufacturer name	Human readable name of the manufacturer of the product. OPC UA Client can retrieve this information from tag: *ServerName|Objects|Server|ServerStatus|BuildInfo|ManufacturerName*
Product name	A human readable name for the product running in the server. The OPC UA Client can retrieve this information from tag: *ServerName|Objects|Server|ServerStatus|BuildInfo|ProductName*

Certificate Parameters

Server certificate can be either generated automatically or by adding an existing certificate file.

Automatically generate self-signed certificate

If auto generated certificate is enabled the certificate is regenerated after every change made by user to certificate parameters. The certificate is also replaced by any explicitly set certificate.

Certificate parameters

Each certificate must contain information that should identify the certificate and its restrictions. If you have chosen to use a self-generated certificate, enter the information you want to be inside the self-generated certificate. Otherwise parameters are read from the certificate you have supplied.

Parameter	Description
Server Name (Common Name)	Name of the certificate (e.g. the device name).
Organization	Organization name
Unit	Organization unit This field could be useful to differentiate different divisions within an organization.
Location	Locality field denotes the city where organization resides in
State	State or Province field specifies where the organization is physically located. Content of State or Province field should not be abbreviated. For example, "CA" is not a valid state name. "California" is the proper state name.
Country	The X.509 naming scheme standard requires a 2-character country code. Country code for the United States is US; country code for Italy is IT.

Parameter	Description
Produc URI	A globally unique identifier for the server. Example: "*urn:NodeName:CompanyName:ServerName*"
DNS Names **IP Addresses**	DNS name or IP Address of the device where this OPC UA Server is installed. Multiple DNS Names and/or IP Addresses can be in a single certificate. **The certificate will be valid only if the IP address where the OPC UA Server is running is included in this list.**
Validity	Period of validity of the certificate starting from creation date
Key Length	Length of the key used by RSA encrypting algorithm

Script to generate a Certificate

If you want provide your own certificate, note that the certificate must include the “Subject Alternative Name (SAN)” parameters as required by the OPC UA standard.

Here is an example of how to generate a certificate using a public OpenSSL-Win32 library (Reference: https://www.openssl.org/)

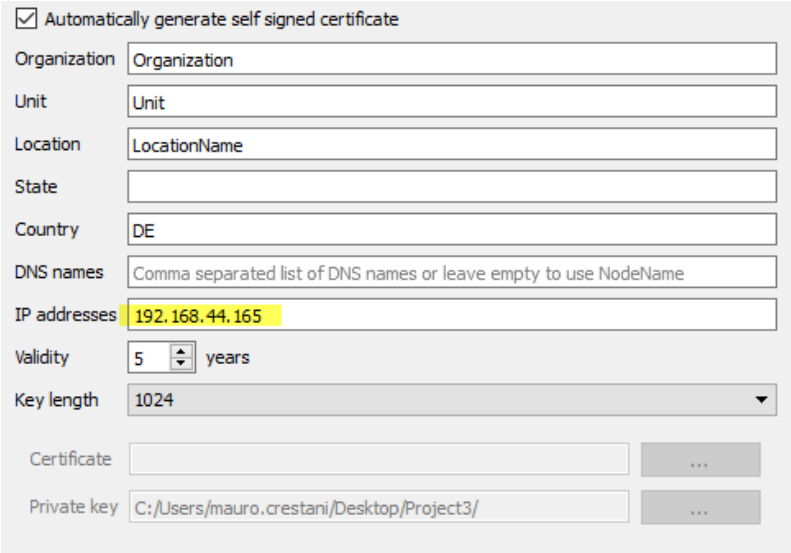
```
@echo off
set OpenSSL="C:\Program Files (x86)\OpenSSL-Win32\bin\openssl.exe"
set NodeName=HMI-Server
set IPAddress=192.168.44.165

rem Generate an RSA key
    %OpenSSL% genrsa -out server-key.pem 2048

rem Creating Certificate Signing Requests
    %OpenSSL% req -new -key server-key.pem -out server.csr -subj "/ST=NY/C=US/L=New
York/O=CompanyName/OU=R&amp;D Team/CN=OPCUAServer@%NodeName%"

rem Creating Certificate (.pem)
    echo subjectAltName=URI:urn:%NodeName%:CompanyName:OPCUAServer,IP:%IPAddress% >
san.txt
    echo
keyUsage=digitalSignature,nonRepudiation,keyEncipherment,dataEncipherment,keyCertSign
>> san.txt
    echo extendedKeyUsage=critical,serverAuth,clientAuth >> san.txt
    echo authorityKeyIdentifier=keyid,issuer >> san.txt
    echo basicConstraints=CA:TRUE >> san.txt
    %OpenSSL% x509 -req -days 3650 -in server.csr -signkey server-key.pem -out
server.crt -extfile san.txt

rem Convert Certificate (.der)
    %OpenSSL% x509 -in server.crt -outform der -out server.der

rem Not necessary files
    del san.txt

pause
```

Using self-signed certificates

This chapter is a step by step example that explains how to configure two HMI devices to communicate using self-signed certificates

OPC UA Server

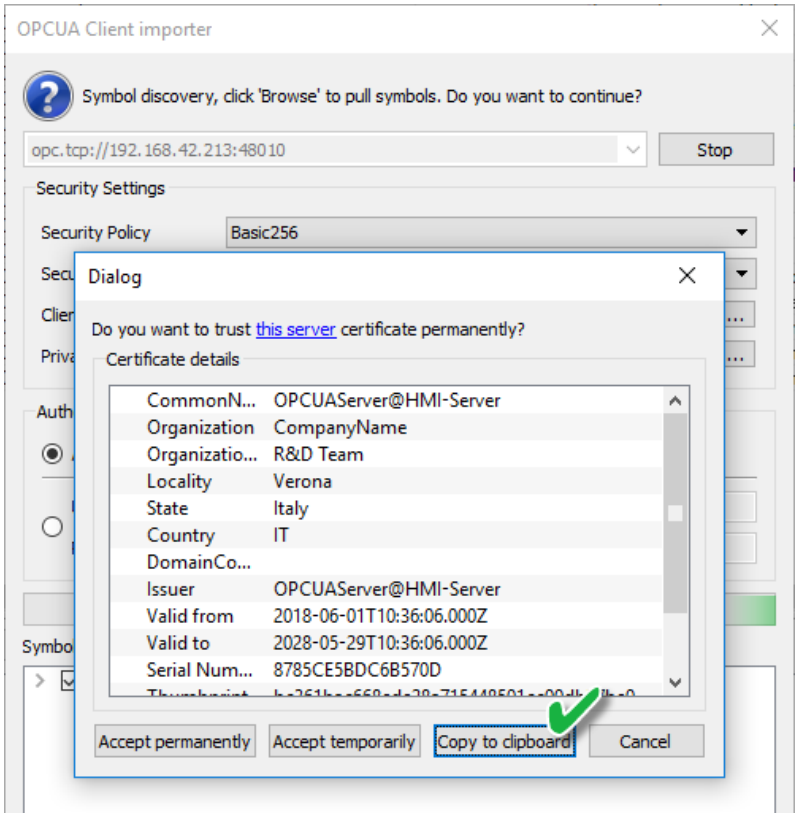
1. Create a simple project including a few tags
2. Open the OPC UA dialog and enable OPC UA Server. Be sure to enable tag groups (e.g. select “All”)
3. Enter in “IP addresses field” the IP address of the HMI device where OPC UA Server will run



4. Download the project to the HMI device

OPC UA Client

5. Create a simple project
6. Add the OPC UA Client protocol. Enter the IP address of the remote OPC UA server and its port number (48010). Leave certificate parameters empty.
7. Open tag editor and import tags. Select “OPC UA Discovery” mode
8. Choose to copy the certificate to the clipboard as shown in the figure. Then, close this dialog and return to protocol configuration dialog to paste the certificate inside the “Server Certificate” field.



9. Repeat step 7, accept the Server OPC UA certificate and import some tags. Note that you can accept the certificate permanently or temporarily. If you accept the certificate permanently, a copy of the certificate will be saved inside your computer for later use without popup again the dialog to asking for confirmation.

The certificate file will be copied inside the folder:
%AppData%\Roaming\Lovato\studio\OPCUA\pki\trusted\certs

10. Open again the protocol dialog box. Select the Security Policy = Basic256 and Security Mode = SignAndEncrypt
11. Download the project to the HMI device

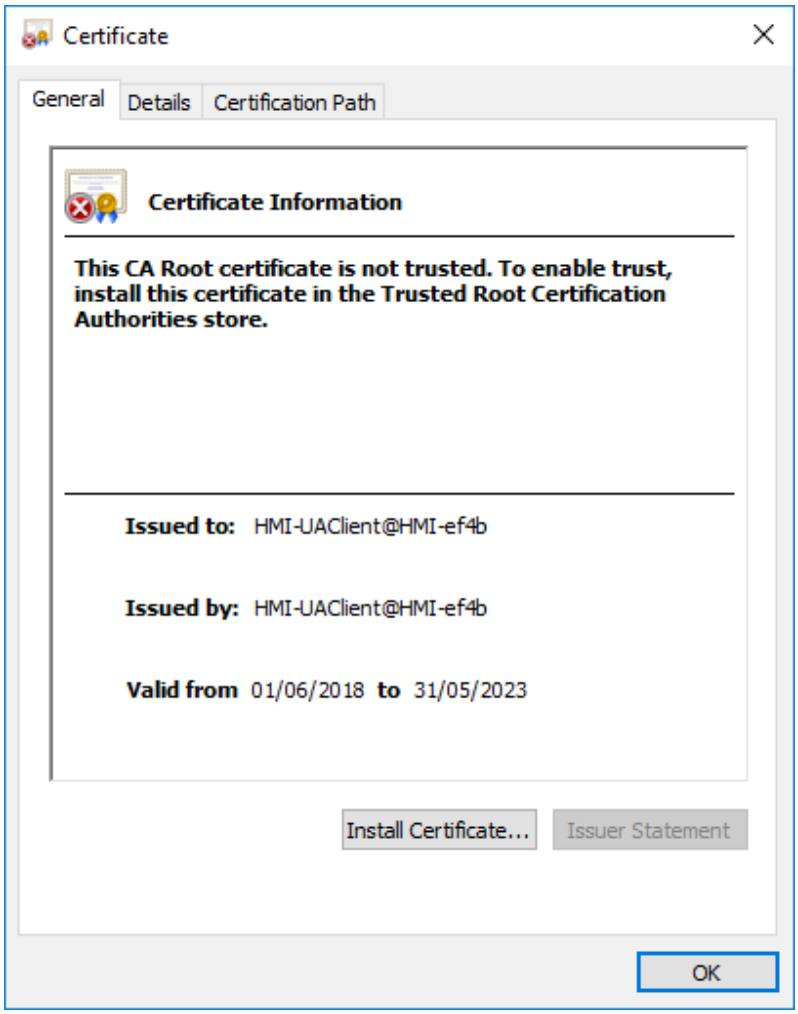
Since in the OPC UA Client protocol parameters we left empty the “Client certificate” field, the OPC UA Client protocol has generated its own certificate and sent it to the OPC UA Server but since the server does not know this certificate it rejects the connection request. Now we have to tell the server to trust these certificates. There are different ways to do it.

Make rejected certificate trusted using FTP client:

1. Connect to OPC UA device using an FTP client
2. Look inside the certificate folders and move the rejected certificate from the rejected folder to the trusted folder.

   /workspace/<YourProjectName>/config/pkiserver/rejected
   /workspace/<YourProjectName>/config/pkiserver/trusted/certs

   You can double click the certificate file to open it and look to certificate parameters to be sure about the certificate you are validating



Now the communication will start

Make rejected certificate trusted using GDS tool:

1. Open the GDS tool and export its certificate
2. Open the project and add the certificate of the GDS tool to the Trusted Certificate list
3. Download the updates project to the HMI device

Now you can manage certificates using the tools in the HMI device.

Using external certificates

This chapter is a step by step example explaining how to configure two HMI devices to communicate using external certificates.

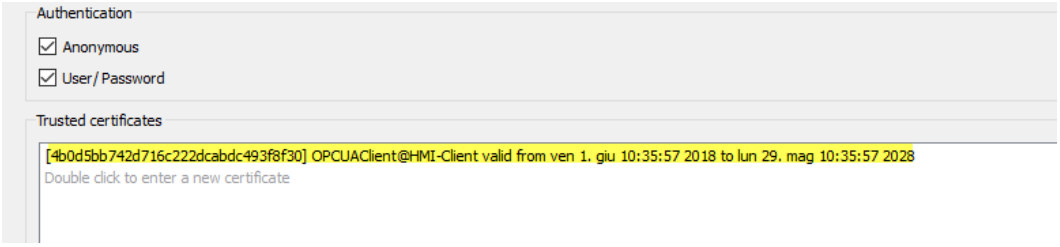
Generate certificates

You can use the script given in this manual to generate a copy of your own certificates, one for OPC UA Server and another one for OPC UA Client.

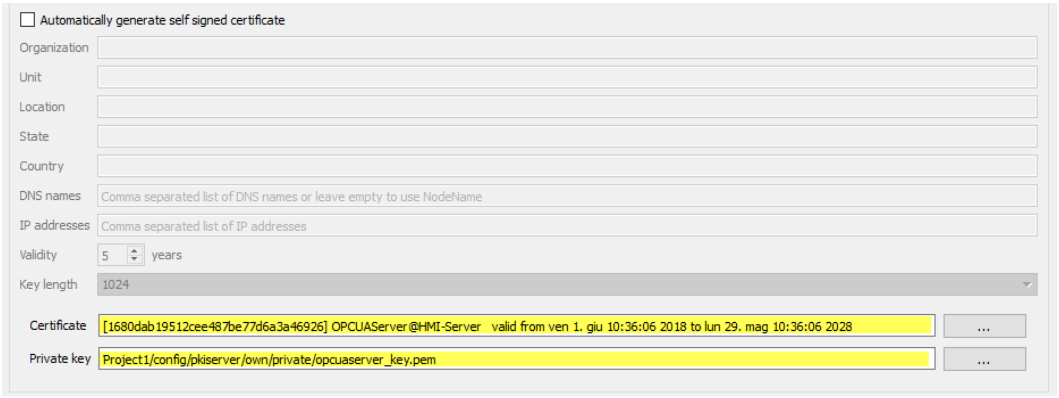
1. Install a OpenSSL-Win32 library (Reference.: https://www.openssl.org/)
2. Use the script ("Script to generate a Certificate" on page 382) to generate OPC UA Server certificate. Be sure to set the IPAddress variable with the IP Address of the HMI device where OPC UA Server will run before running the script.
3. Find in the OPC UA Client protocol manual a sample script to generate a certificate for the OPC UA Client protocol

OPC UA Server

4. Create a simple project that using a few tags
5. Open the OPC UA dialog and enable OPC UA Server. Be sure to enable tag groups (e.g. select “All”)
6. Add the client.der certificate to the Trusted Certificate area to enable the OPC UA Client to communicate with OPC UA Server



7. Remove the check on “*Automatically generate self-signed certificate*” and add the server certificate (server.der) and the server private key (server-key.pem)



8. Download the project to the HMI device

OPC UA Client

9. Create a simple project
10. Add OPC UA Client protocol.
11. Enter the IP address of the remote OPC UA server and its port number (48010).
12. Open the ASCII version of the server certificate (server.crt), remove all Newline characters and then copy and paste the ASCII characters of your certificate inside the Server Certificate field.
13. Repeat the same with Client Certificate (client.crt) and Client private key (client-key.pem)
14. Select the Security Policy Basic256 and the Security Mode = SignAndEncrypt
15. Open tag editor and import tags. Select “OPC UA Discovery” mode
16. Accept OPC UA Server certificate, import some. Note that you can accept the certificate permanently or temporarily. If you accept the certificate permanently, a copy of the certificate is saved inside your computer to later usage without asking you for confirmation
17. Download the project to the HMI device

You will note that OPC UA Client is retrieving data from OPC UA Server using the given certificates.

Alarm map

The alarm states are mapped to OPC UA states according to the following rules:

OPC UA Alarm state	LRH SW Alarm state
Opcua.Alarm.Active	True when alarm state is triggered • TRIGGERED • TRIGGERED_NOT_ACKED • TRIGGERED_ACKED
Opcua.Alarm.Acked	True when alarm acknowledgment is not required • TRIGGERED_ACKED • NOT_TRIGGERED_ACKED • NOT_TRIGGERED
Opcua.Alarm.Retain	True when alarm is pending • TRIGGERED | • TRIGGERED_NOT_ACKED | • TRIGGERED_ACKED | • NOT_TRIGGERED_ACKED but a RESET is required
Opcua.Alarm.Confirmed	True when alarm is returned (Not triggered, acknowledged and reset) • NOT_TRIGGERED This info is available only when alarm is configured to required a RESET

36 MQTT Interface

Path: ***ProjectView*** *>* ***Config*** *>* ***Interfaces*** *> double-click* ***MQTT***

Use MQTT Interface to publish data according to the MQTT standard.

Note that a tag or an alarm, to be transferred through the MQTT protocol, must be defined within a group.

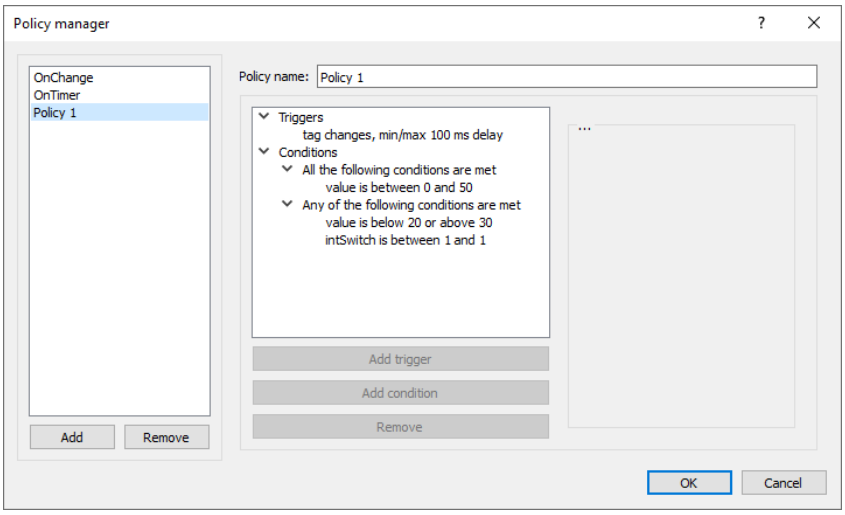
Parameter	Description
Enable MQTT Interface	Main flag to activate MQTT service. The selected groups of tags will be published to the MQTT broker.
Enable Alarms	The selected groups of alarms will be published to the MQTT broker. Alarms are published whenever there is a change in the alarm status.

Tags configuration

Parameter	Description
Enable	Enable the transferring of the tags listed inside the group
Tag Group	List of tags that will be transferred when the assigned police condition will be satisfied.
QoS	QoS to use 0 = Delivered at most once (Fire and forget) which means no confirmation 1 = Delivered at least once, which means confirmation required 2 = Delivered exactly once, which means a 4 step handshake is done
Retain	This flag defines whether the message is saved by the broker as the last known good value for a specified topic. When a new client subscribes to a topic, they receive the last message that is retained on that topic.
Persistence	When true, the messages with QoS greater than 0 are queued into the file system file to be available even after a panel reset or when a broken communication with the MQTT server is reestablished.
Policy	Defines the criteria for deciding when to publish the value of a tag. When it is empty, the "Default push policy", defined on top of the table, is used.

Manage push policy

A policy consists of a trigger criterion and several (optional) conditions that must be verified in order for the tag value to be transmitted.



Trigger

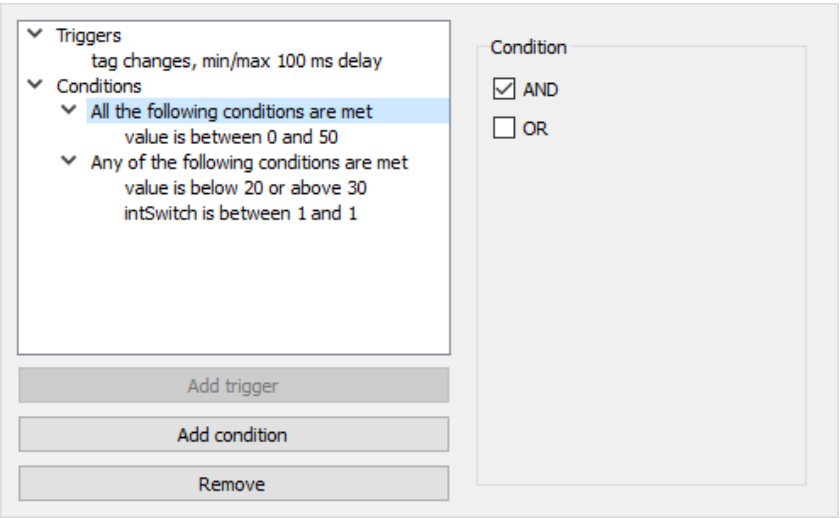
Parameter	Description
Timer	Publish is performed continuously even value is not changing. • Interval (ms) Cyclical publication time
On change	Publish is performed when a tag value changes. • Min interval (ms) Value check interval • Deadband The difference, from previous publish, that must be found to trigger the new publish. • Use percentual Dead band value express in percentage • Tag Name Tags to be checked to activate the publication. If empty, the tag to be published is used.

Conditions

Conditions contain folders of conditions. Each folder can be of two types:

- All the following conditions are met (AND)
- Any of the following conditions are met (OR)

All folders must be validated to have the transmission requested by the trigger. A folder of type "All the following condition are met" is validated when all the contained conditions are true while a folder of type "Any of the following conditions are met" is validated when at least one contained condition is true.



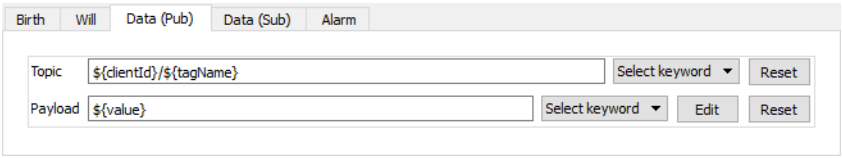
Settings

Topic	Description
Max pending messages	The number of messages that can be queued in RAM when there are communication errors. Queue messages will be released as soon as the MQTT Server returns reachable.
Defaults	Values of "QoS", "Retain" and "Persistence" parameter to use for the topics that are not defined inside the "Tags configuration" table.

Topic and Payload

There are five types of supported topics:

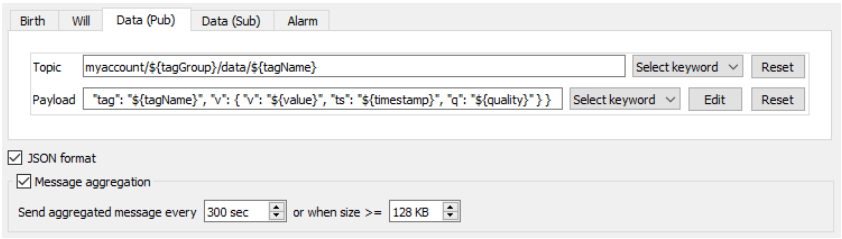
Topic	Description
Birth	This is a special topic that is publish only one time when HMI device start.
Will	This is a special topic that is publish when HMI device start but it is keep hidden and stored from the MQTT Broker. It will be published from the MQTT Broker if it detects that the client has disconnected ungracefully.
Data (Pub)	Topic is used to publish the tags' values following the transmission policies associated with tag groups.
Data (Sub)	Topic is used to subscribe to tags. The payload is the template used to recognize the values of the received tags.
Alarm	Topic used to publish alarms



For each topic, the payload defines the structure of the associated value. Note that in topic and payload definitions can be used placeholders.

Placeholder	Description
${clientId}	MQTT Client ID
${currentTimestamp}	Current time
${protocolName}	Name of the protocol associated to a tag
${tagGroup}	Name of the group the tag belongs to
${tagName}	Name of the tag
${alarmGroup}	Name of the group the alarm belongs to
${alarmName}	Name of alarm
${value}	Last known value of the tag
${timestamp}	Timestamp of when tag value was read
${activeValue}	Value of the tag when the alarm became active
${quality}	Quality (i.e. reliability) of the tag
${activeTimestamp}	Timestamp of the last event that raised the alarm
${inactiveTimestamp}	Timestamp of the last event that ceased the alarm condition
${ackTimestamp}	Timestamp when the operator acknowledge the alarm
${description}	Alarm description
${customField1}	Alarm Custom Field 1
${customField2}	Alarm Custom Field 2
${state}	Alarm State
${severity}	Alarm Severity
${lowLimit}	Alarm “Low limit”
${highLimit}	Alarm “High limit”
${[0]}	If available in the alarm description, the value of the first live tag, [1] the second, etc.
${[Tag1]}	If available in the alarm description, the value of "Tag1" live tag

JSON Payload



When the **JSON format** is selected, the quotation marks are added around string values to conform to the JSON syntax.

With the use of the JSON format is possible to optimize the communication to include multiple messages inside a single message. When the "**Message aggregation**" is selected, the messages are sent to MQTT Server after the selected timeout expired or when the message to send reaches the select size.

MQTT Broker Settings

Current supported MQTT Broker are:

- Generic MQTT broker
- Azure
- Amazon AWS
- IBM BlueMix
- Murano

Note that some parameters depend on the broker has chosen.

Generic MQTT Broker

Parameter	Description
Broker address	Name or IP address of the MQTT server
Port	MQTT server port. Generally, the default TCP/IP port is 1883, or the port 8883 when MQTT over SSL is used.
Client ID	The client identifier is an identifier of each MQTT client connecting to an MQTT broker. You can write what you prefer, but it has to be unique per broker. The broker uses it for identifying the client and the current state of the client.
Username **Password**	If the MQTT broker is configured to require client authentication using a valid user name and password
Keep-alive time (s)	Time interval before sending a PING request to the server when there are no data flows (useful to know if both client and server are still alive and reachable).
Use clean session	When the clean session flag is set to false, the broker creates a persistent session for the client. All information and messages are preserved until the next time that the client requests a clean session. If the clean session flag is set to false and the broker already has a session available for the client, it uses the existing session and delivers previously queued messages to the client.
Use legacy	The “Use legacy” flag makes client comply with MQTT spec 3.1

Enable TLS

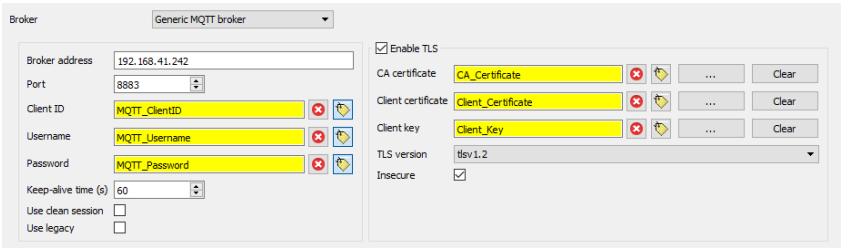
If the MQTT server is configured to works over TLS connection, the HMI device must provide its own certificate to the server. Even it's not mandatory, each client should have its own certificate (however it is possible you can deploy the same certificate to all clients).

It is required that both server and client certificates are signed by the same authority.

Parameter	Description
Enable TLS	Enable the TLS encryption
CA Certificate	Public certificate of the CA that has signed the server certificate on the Mosquitto Broker
Client Certificate	Public certificate of the HMI Device. Must be signed from the CA Certificate
Client Key	Private key associated with the client certificate
TLS Version	TLS Version to use (must be aligned with the encryption level used from the MQTT Broker) • tlsv 1 • tlsv 1.1 • tlsv 1.2
Insecure	This option disables verification of the server host name in the server certificate. This can be useful when testing initial server configurations but makes it possible for a malicious third party to impersonate your server through DNS spoofing, for example. Use this option in testing only.

Note that you can use the "attach to tag" to entered the MQTT parameters at runtime using, e.g., macros, JavaScript or a configuration page. This could be useful to have different values (e.g. for the ClientID) even downloading the same project to different HMI devices.

If you use tags to define MQTT settings (e.g. Client ID), be sure to not include these tags into the tags list exchanged with the MQTT server to avoid to receive back wrong settings.



The string Tags used for the certificate must be great enough (e.g. 2.048 bytes) to contain the entire certificate. The format of the certificates must be ASCII with the first and the last text line included as for the below example.

MQTT Broker Example

Here is a little example that explains how to configure an application to communicate with an MQTT server. In this example, we configure the HMI device to communicate with an open-source MQTT broker (https://mosquitto.org) using certificates. The certificates will be created using a public OpenSSL-Win32 library (https://www.openssl.org).

Generate the certificates

The following script file will create a couple of server and client certificates and a public Certificate Authority that will be used to sign server and client certificates and to verify the authenticity of these certificates.

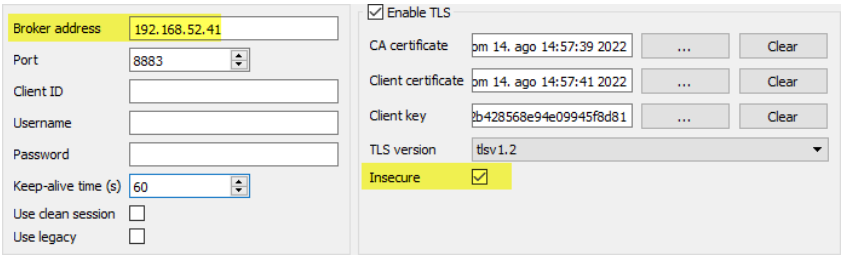
- ca.crt
- server.crt, server.key
- client.crt, client.ket

File: *CreateCertificates.cmd*

```
@echo off
set OpenSSL="C:\Program Files (x86)\OpenSSL-Win32\bin\openssl.exe"

rem Generate self signed CA certificate (Certificate Autority)
    %OpenSSL% req -nodes -batch -new -x509 -days 1000 -keyout ca.key -subj
"/CN=Broker/O=company.com" -out ca.crt


rem Generate MQTT Server private key
    %OpenSSL% genrsa -out server.key 2048

rem Generate MQTT Server certificate signed request
    %OpenSSL% req -batch -new -key server.key -subj "/CN=localhost/O=company.com" -out
server.csr
```

```
rem Sign the MQTT Server certificate
    %OpenSSL% x509 -req -days 1000 -in server.csr -CA ca.crt -CAkey ca.key -
CAcreateserial -out server.crt


rem Generate HMI Client private key
    %OpenSSL% genrsa -out client.key 2048

rem Generate HMI Client Server certificate signed request
    %OpenSSL% req -batch -new -key client.key -subj "/CN=client/O=company.com" -out
client.csr

rem Sign the HMI Client certificate
    %OpenSSL% x509 -req -days 1000 -in client.csr -CA ca.crt -CAkey ca.key -
CAcreateserial -out client.crt


rem Remove unnecessary files
    del *.rnd *.srl *.csr

pause
```

Note the server hostname is localhost (/CN=localhost), this means that you cannot use the secure connection if in the Broker address parameter you cannot write the "localhost" domain. You can use the "localhost" domain only if both the MQTT Server and the HMI device are running on the same device otherwise, to be able to reach the MQTT server, you must use the IP Address and the "Insecure" flag.



MQTT Broker configuration

The server certificate (server.crt, server.key) and the authority certificate (ca.crt) must be place inside a subfolder of the MQTT folder, e.g. inside the "certs" subfolder.

The "*mosquitto.conf*" file has to be configured to use the TLS support

```
# =================================================================
# Default listener
# =================================================================
... (omiss) ...
# Port to use for the default listener.
#port 1883
port 8883

# -----------------------------------------------------------------
# Certificate based SSL/TLS support
# -----------------------------------------------------------------
```

```
... (omiss) ...
#cafile
#capath
cafile certs/ca.crt
certfile certs/server.crt
keyfile certs/server.key
tls_version tlsv1.2
```

MQTT Broker can be started using the below command from a dos command window:

```
mosquitto -v -c mosquitto.conf
```

MQTT Client

For testing purposes, it could be useful to start an MQTT client with the subscription of all the topics so that you can see the messages that will be exchanged with HMI Device. Since we are using TLS communication, we must provide the client certificate. We can copy client.crt, client.key and the authority certificate ca.crt inside the certs-client subfolder.

So the command to activate an MQTT client is:

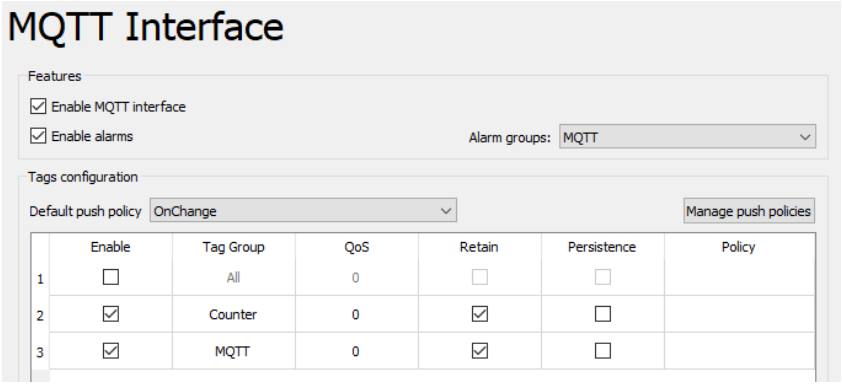
```
mosquitto_sub --cafile certs-client\ca.crt --cert certs-client\client.crt --key certs-client\client.key -p 8883 -t /#
```

HMI Device

To configure the HMI device we must provide:

- set the broker address parameter with the IP address where the MQTT server is running
- set the port address to 8883
- load the authority certificates, the client certificate and the client key files
- set the TLS Version to version 1.2 to be aligned with the MQTT server settings
- since it is probably that you are referencing the MQTT server using the IP address, which is different from the domain declared by the server certificate you must set the "Insecure" flag

To perform the first tests, you can leave the default values on topics and payloads and configure the alarms groups and tags groups that you want to transfer to the MQTT broker.



37 Special widgets

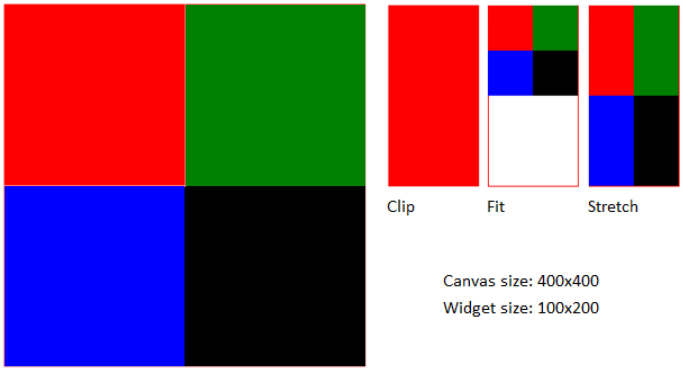
Widgets designed for special purposes are called special widgets and include control lists, date and time widgets, variable widgets and so on.

Canvas Widget

Path: **Widget Gallery> Basic> Generic Canvas**

Canvas widget can be used to draw graphic via JavaScript scripting.

Note: the JavaScript methods are the same that are available for the HTML5 <canvas> tag

Parameter	Description
Canvas Width **Canvas Height**	Canvas size. Note this is not the widget size. For example, the canvas size could be 500x500 pixels where the widget size could be 100x100 pixels. Draw Hint parameter will define how to stretch the canvas size to fit the widget size.
Draw Hint	Define how fit the canvas inside the widget size • Clip No Transformation is applied, coordinate system is not scaled and drawing is clipped inside the widget bounding rectangle. • Fit to size Fit to the widget size preserving the canvas model aspect ratio. • Stretch Fit to the widget size ignoring the canvas model aspect ratio. Example using a Canvas size larger than the widget size: 
Design Time Preview	Canvas preview inside LRH SW Note the JavaScript code could use data not available inside LRH SW but only inside the HMI device
Auto Clear Background	Automatic clear the background before draw canvas. When disabled, the painted items are persisted and is not necessary redraw everything from scratch.

Parameter	Description
OnDraw Action	The OnDraw event is executed when the page is painted. This event has to be linked with the JavaScript code that draws the canvas graphic.
OnMousePress Action **OnMouseRelease Actions** **OnMouseDrag Actions**	Mouse events

Available Canvas Methods

// Painter Save/Restore

- void save(); // calls painter save
- void restore(); // calls painter restore

// Scale/Transform

- void scale(qreal x, qreal y);
- void rotate(qreal angle);
- void translate(qreal x, qreal y);
- void transform(qreal m11, qreal m12, qreal m21, qreal m22, qreal dx, qreal dy);
- void setTransform(qreal m11, qreal m12, qreal m21, qreal m22, qreal dx, qreal dy);

// Gradient

- CanvasGradient createLinearGradient(qreal x0, qreal y0, qreal x1, qreal y1);
- CanvasGradient createRadialGradient(qreal x0, qreal y0, qreal r0, qreal x1, qreal y1, qreal r1);

// Rectangle Functions

- void clearRect(qreal x, qreal y, qreal w, qreal h);
- void fillRect(qreal x, qreal y, qreal w, qreal h);
- void strokeRect(qreal x, qreal y, qreal w, qreal h);
- void rect(qreal x, qreal y, qreal w, qreal h);

// Path

- void beginPath();
- void closePath();
- void moveTo(qreal x, qreal y);
- void lineTo(qreal x, qreal y);
- void quadraticCurveTo(qreal cpx, qreal cpy, qreal x, qreal y);
- void bezierCurveTo(qreal cp1x, qreal cp1y, qreal cp2x, qreal cp2y, qreal x, qreal y);

// Drawing Text

- void fillText(const QString &text,qreal x, qreal y);

// Arc

- void arcTo(qreal x1, qreal y1, qreal x2, qreal y2, qreal radius);
- void arc(qreal x, qreal y, qreal radius, qreal startAngle, qreal endAngle, bool anticlockwise);

// Fill/Stroke

- void fill();
- void stroke();
- void clip();
- bool isPointInPath(qreal x, qreal y) const;

// Image manipulation (Draw CImageWgt using target and source rect)

- void drawImage(QObject *pObjImage, qreal sx, qreal sy,qreal sw, qreal sh, qreal dx, qreal dy,qreal dw, qreal dh);
- void drawImage(QObject *pObjImage, qreal dx, qreal dy);
- void drawImage(QObject *pObjImage, qreal dx, qreal dy, qreal dw, qreal dh);
- void drawImage(const QVariant& image, int width, int height, const QString& format, qreal sx, qreal sy,qreal sw, qreal sh, qreal dx, qreal dy,qreal dw, qreal dh);

// Pixel manipulation

- ImageData createImageData(double sw, double sh);//Empty Image
- ImageData createImageData(ImageData fromImage);//from another Image
- ImageData createImageData(ArrayBuffer value); //From arraybuffer
- void putImageData(ImageData imgData,double dx, double dy);
- void putImageData(ImageData imagedata, double dx, double dy, double dirtyX, double dirtyY, double dirtyWidth, double dirtyHeight);
- ImageData getImageData(qreal sx, qreal sy, qreal sw, qreal sh);

Canvas JavaScript Example

The canvas is initially blank. To display something, a script first needs to access the rendering context and draw on it:

```
var ctx = me.context2d;
```

then you can use the canvas methods, as in the below example

```
function GenericCanvasWgt1_onDraw(me, eventInfo)
{
    var ctx = me.context2d;
    ctx.fillStyle = 'red';
    ctx.fillRect(0,0,250,250);
    ctx.fillStyle = 'green';
    ctx.fillRect(250,0,250,250);
    ctx.fillStyle = 'blue';
    ctx.fillRect(0,250,250,250);
    ctx.fillStyle = 'black';
    ctx.fillRect(250,250,250,250);
}

function GenericCanvasWgt1_onMouseDown(me, eventInfo)
{
    alert("X = " + eventInfo.posX + "\nY = " + eventInfo.posY );
```

```
}
```

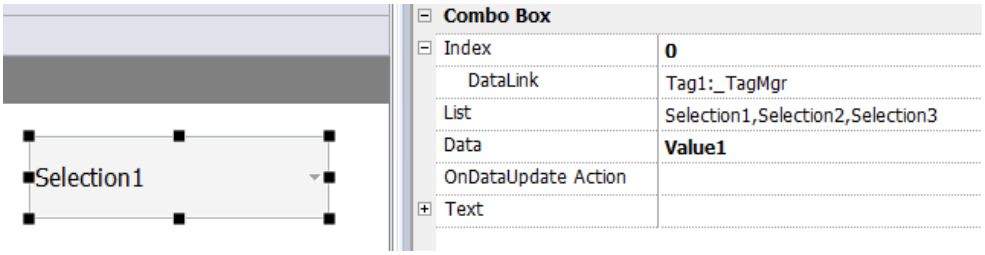
The update method can be used to dynamically redraw a canvas widget

```
function BtnStd1_btn_onMouseClick(me, eventInfo)
{
    var myCanvasWidget = page.getWidget("GenericCanvasWgt1");
    myCanvasWidget.update()
}
```

Combo Box widget

Path: ***Widget Gallery> Basic> Controls***

Use this widget as a selector widget or to filter rows in a table to display only the values selected in the combo box.



Parameter	Description
Index	Index of the selected item.
List / String List	Item strings in the combo box. Note: This field is multi-language.
Data / Data List	Returns the value in the Data List column (as string) in the Data field of the widget. Tip: Use this parameter to return a custom value based on an item selected in the combo box.
Text	Format of displayed text.

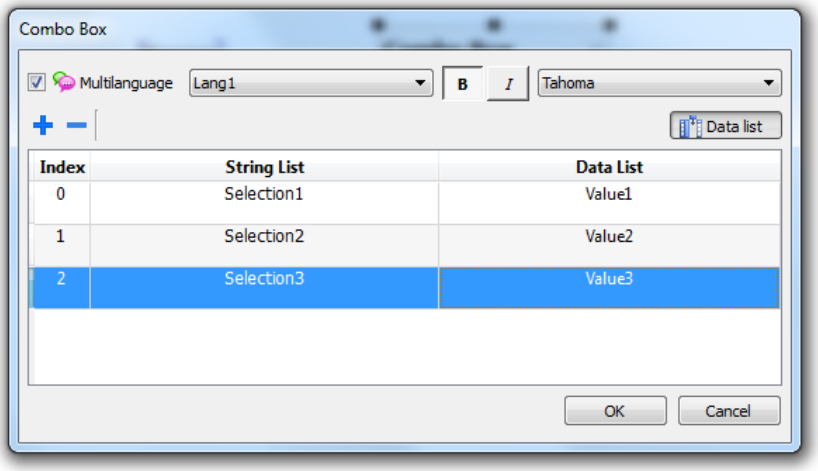
Data List

The Data List is associated with the "listData" property and can be modified dynamically using the JavaScript code.

```
// To read the Data List
   var comboWgt = page.getWidget("Combo1");
   var listData = comboWgt.getProperty("listData")

// To write the Data List
```

```
var comboWgt = page.getWidget("Combo1");
comboWgt.setProperty("listData", "NewData1,NewData2,NewData3");
```

Attaching data vs. attaching indexes



In many projects you may need to attach fields such as **Index** or **Data** to tags to know the values of the selected item in the combo box. Use:

- **Index**: to display the index (integer) of the selected item (0...n).
- **Data**: to display the data value (string) specified in the Data List column.

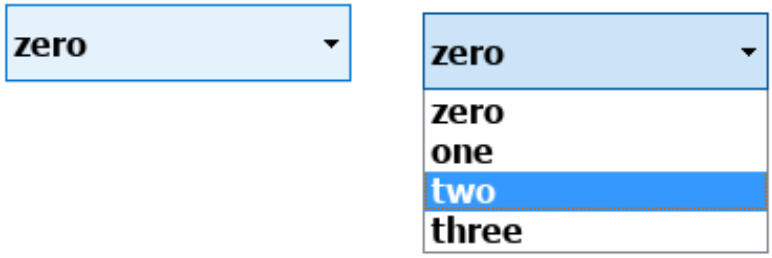
Combo Box widget “full screen” mode with images

From the "Project properties" on page 73 the look and behavior of Combo Boxes can switches from Context mode to Full Screen mode

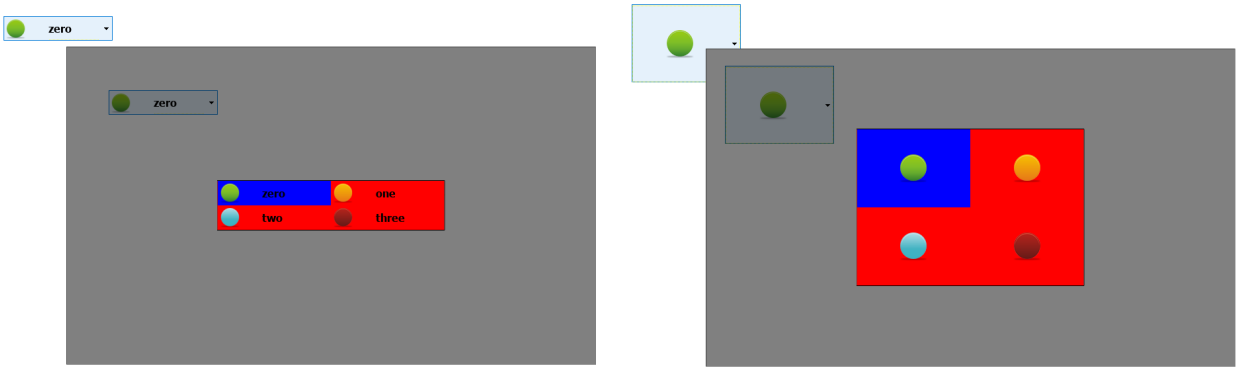
Path: **ProjectView**> double-click **Project properties**> **Properties pane**> **Project**> **ComboBox View Mode**

Parameter	Description
ComboBox View Mode	Select the visualization mode of all the Combo Box widgets of the project **Context** Classic view with drop-down menus **Full screen** Enhanced view with configurable texts and images that will pop up in the middle of the screen for easy scroll and selection.

Context view example

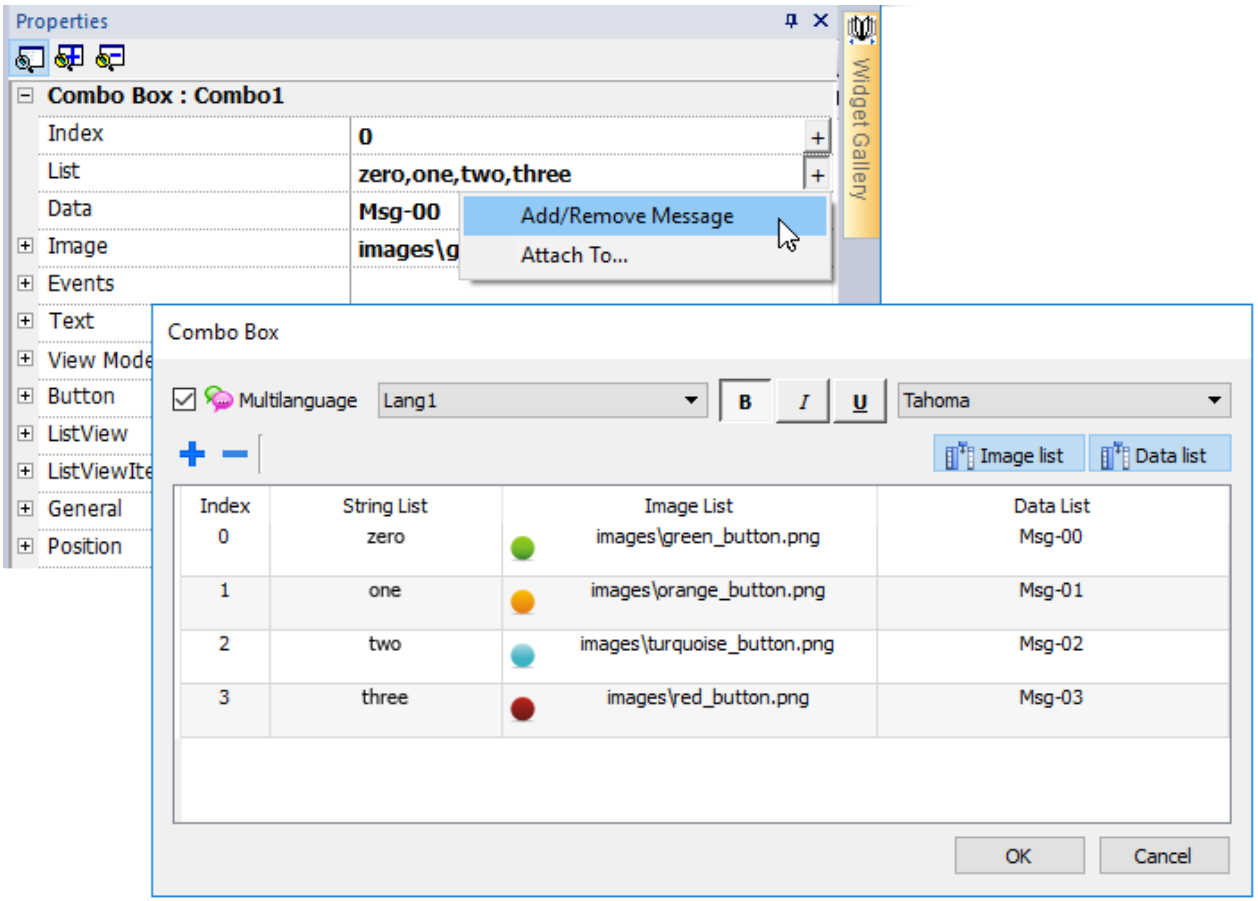


Full screen view example



Additional parameters available in full screen mode

The additional "*Image List*" column will be available inside **Combo Box**> **List** parameter:



Note: Some properties are displayed only in advanced mode.

Parameter	Description
Image	Return, inside the attached tag, the file name of the selected image
Button	Define the look of the Combo Box • Show background = true Combo Box button is showed • Show background = false Only image or text is showed
ListView	Layout parameters of the Combo Box in edit mode
ListViewItems	Define the items type that will be inside the Combo Box Image Mode: • Only Text • Only Images • Text and Images

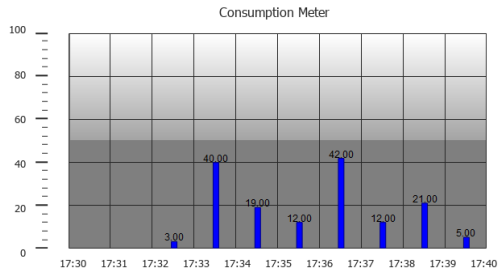
Consumption Meter widget

Path: ***Widget Gallery> Basic> Trends/Graphs***

Use this widget to monitor a resource which is continuously increasing. The system reads the value of the resource and calculates the increment in a set range of time, the increment is then displayed in a bar-graph in a trend-like window.

Different colors can be used to used in the graph based on the time frame.

Tip: Use this widget to calculate the power consumption of a system.

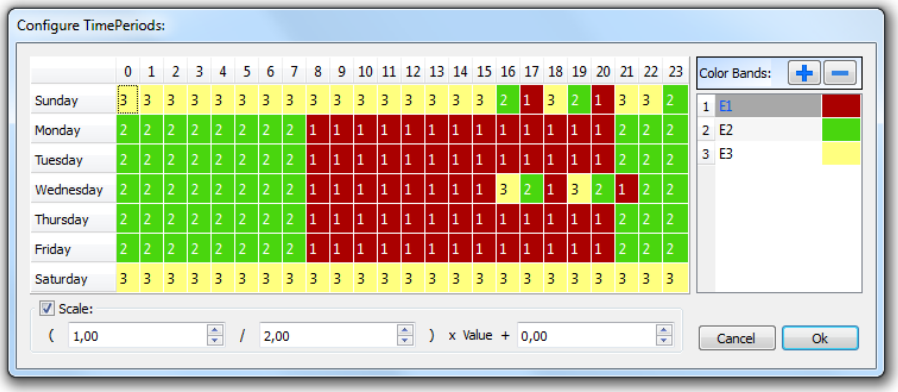


Parameter	Description
Value	Resource monitored
Graph Duration **Graph Duration Units**	Time period displayed in the window
Bar Duration **Bar Duration Units**	Time period represented by each bar in the graph
Time Periods	Assigns a specific color to highlight the increment of the monitored resource in a specified time period (minimum resolution = 1 hour).
Color **Bar Width**	Bar color and width
Bar Value	Show/Hide the value of each bar
Consumption Meter	Number of labels to be displayed on graph.

Example: how to monitor energy consumption

In the following example a widget is design tho monitor energy consumption with a weekly scale and a daily unit.

1. Attach a tag to the physical variable to monitor. In this example, to the total energy consumed (Tag KWh). This tag contains an incremental number that indicates how many KW/h have been consumed from when energy consumption started.
2. Add a Trend and link it to the tag to be monitored, Tag KWh.
3. Add a **Consumption Meter** widget to a page.
4. Attach the **Value** property of the Consumption Meter to the Trend you created in step 2.
5. Set **Graph Duration/Units** to 1 week: this will give you a weekly graph of consumed energy.
6. Set **Bar Duration/Units** to 1 day, this is the time range when energy consumption is calculated.
7. In **Consumption Meter** set the number of labels to show in the bar graph, in this case 7 to display a weekly graph.
8. From the **Time Periods** property open the **Configure Time Periods** dialog: set the different colors for different values of Tag KWh in each bar.



Tip: To assign the color to the cells of the table, select the cells and click on the desired color, or enter the index value of the band (1, 2, 3) into the cell.

9. Add as many color bands as you need, in this example 3 color bands.
10. Assign a band to each hour in the weekly table, in this example a red band (E1) is used to indicate the range of time in the day/week where the cost of energy is the highest.

Note: You can apply a scale factor to each color band, if needed.

The result is a bar graph consumption meter showing daily consumption of energy in KW/h, with colors indicating the different energy costs. The height of each bar represents the amount of energy in the time range considered, 1 day in this example.

Use the action ConsumptionMeterPageScroll to scroll the bar graph back and forth and the action RefreshTrend to refresh the bar graph since data is not refreshed automatically.

Important: No other Trend action is currently supported by the Consumption Meter widget.

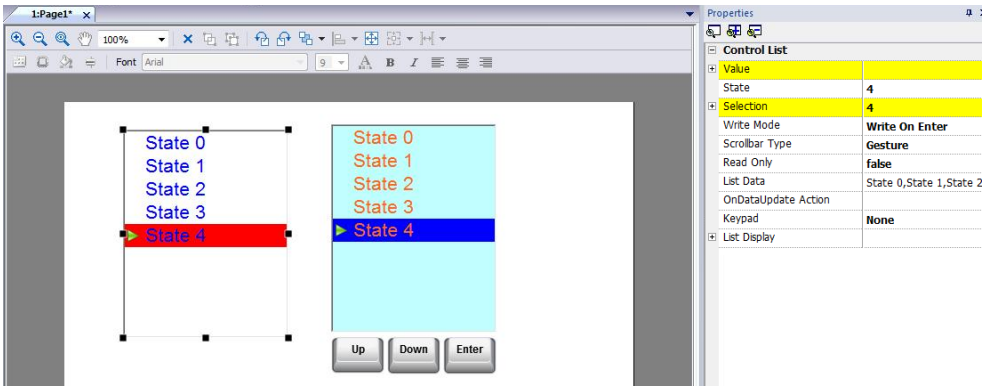
Control list widgets

Path: ***Widget Gallery> Advanced> Control List***

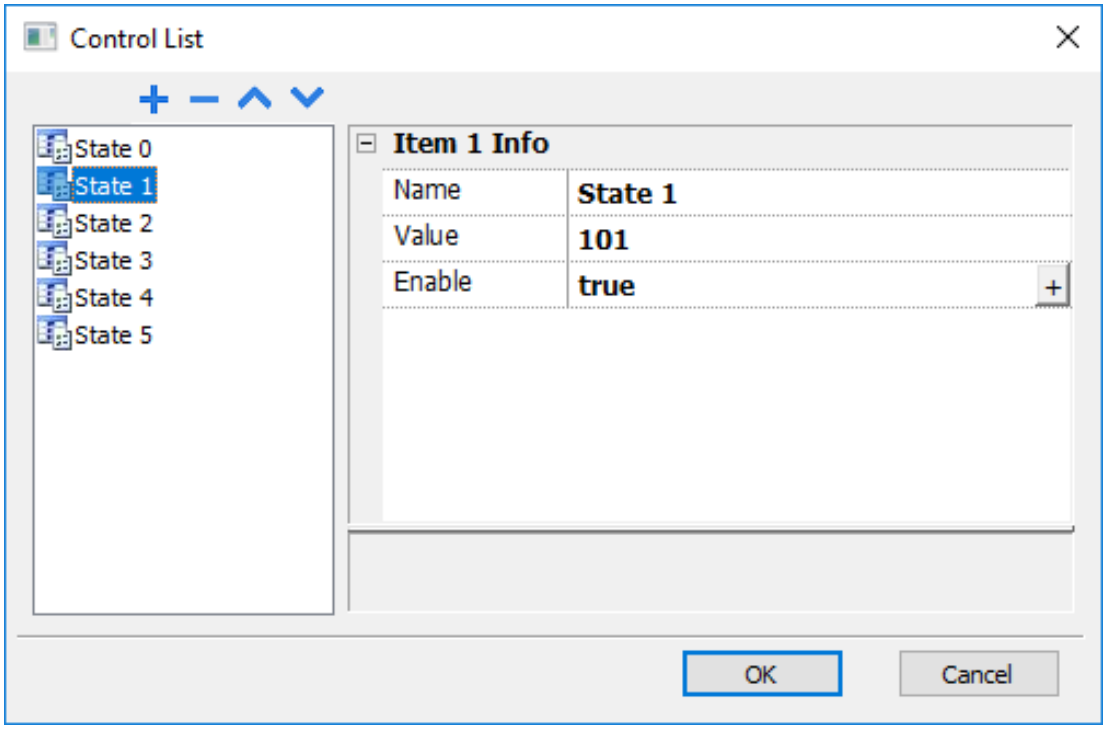
Use these widgets to represent the status associated with a particular process and to control that process from the same widget.

Two types of control lists are available:

- a group control list, with a limited set of navigation button already included, and
- a basic control list with no pre-configured button to be navigated using the touch screen feature.



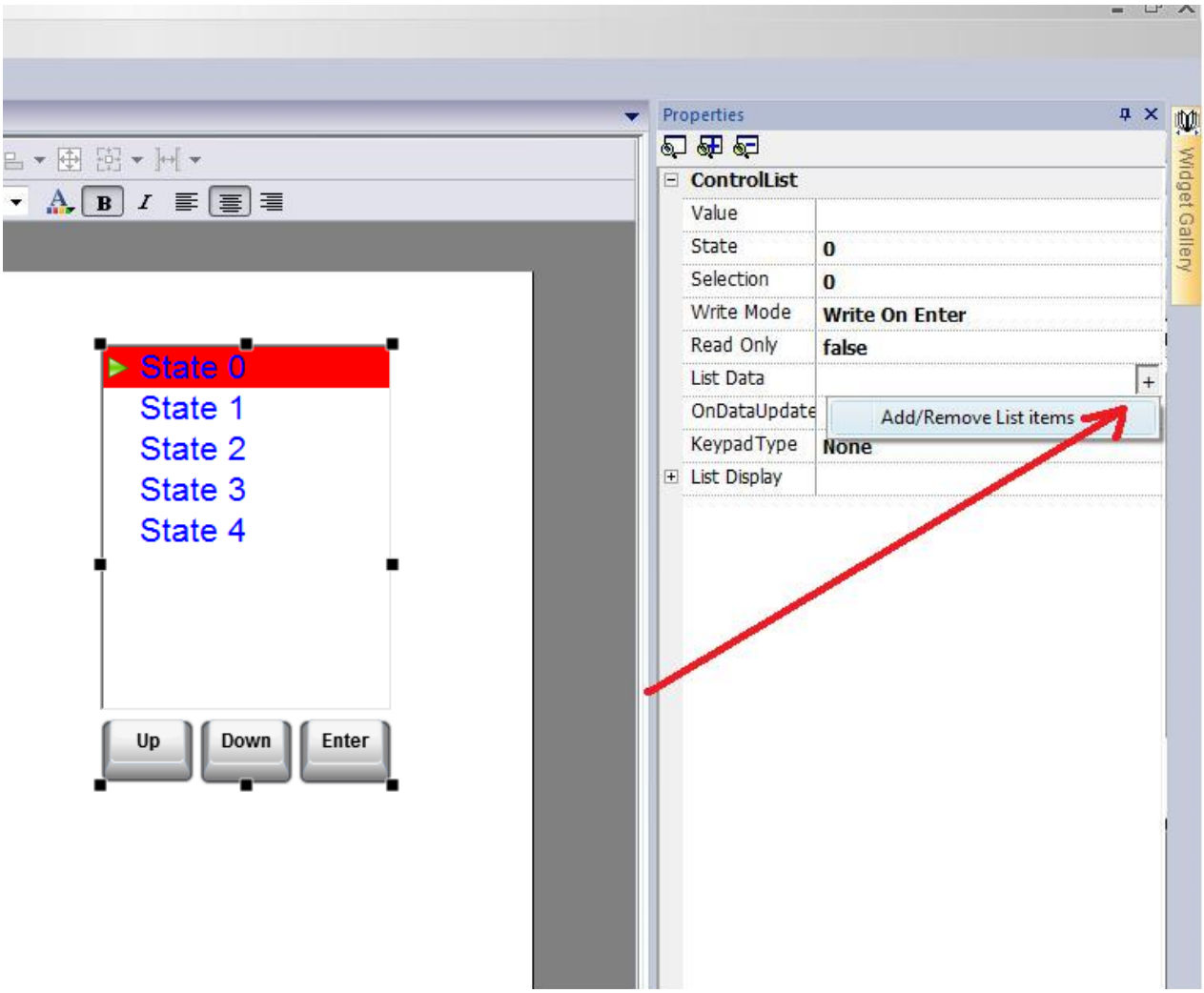
Parameter	Description
Value	The value corresponding to the status of the widget. If there is a tag attached to the value property, when loading the widget, the State will be aligned with the tag value.
State	State of widget. The widget highlights the item related to its State with a different background color (see "state color" in the properties of the widget).
Selection	State selection. The selected item will be displayed with a small triangle on the left side of the list.
Write Mode	Select the State update mode • **Write On Select**: The state is updated automatically to be aligned with the cursor position. • **Write On Enter**: The status is updated with the cursor position only when the user presses enter
Scrollbars Type	Select the scroll mode of the table • **Gesture**: Pan gesture can be used to smoothly scroll the table. • **Scrollbar**: Use the scrollbar to scroll the table

Parameter	Description
Read Only	Defines whether the list is only an indicator.
List Data	List of status items. Each item has a status name, a corresponding value and a flag that enables to display the item inside the widget. 

Defining states

Add/remove states, that is items in the list, from the **List Data** property.

Any value can be assigned to a state. When you activate the state, by selecting the related item if in **WriteOnSelect** mode or selecting it and confirming with enter if **Write On Enter**, this will write the value assigned to state to the tag linked to the Control List widget **Value** .



Manage list data items from JavaScrip code

The list of data items can be modified, at runtime, from JavaScript code using the ***setProperty("listData", <NewControlList>)***. The below example shows how to modify the list of items

```
function SetItemsList_btn_onMouseClick(me, eventInfo)
{
    var NewControlList = [["OFF",100,true],["ON",101,true],["MAN",102,true],
["AUTO",103,true]];
    var ControListWgt = page.getWidget("controlListBtn.controlList");
    ControListWgt.setProperty("listData", NewControlList);
}
```

Where

- *NewControlList* is an array with the items description
- *controlListBtn.controlList* is the ID of the Control List Widget to modify

The ***getProperty("listData")***, instead, will just return a comma separated string of just the names.

```
function Read_btn_onMouseClick(me, eventInfo)
{
    var ControListWgt = page.getWidget("controlListBtn.controlList");
    var ListData = ControListWgt.getProperty("listData");
}
```

Where the result of ListData will be: "*OFF,ON,MAN,AUTO*"

State

The ***getProperty("state")*** can be used to retrieve the State value. Here is an example of the JavaScript code

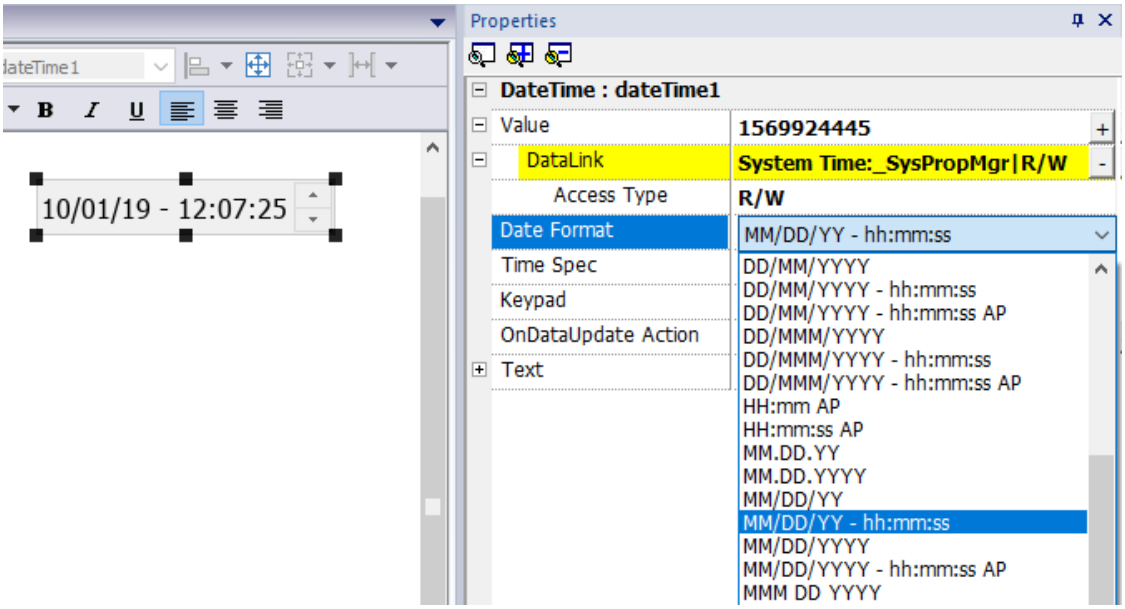
```
function controlListBtn_onDataUpdate(me, eventInfo)
{
    var ControListWgt = page.getWidget("controlListBtn.controlList");
    var State = ControListWgt.getProperty("state");
    project.setTag("State", State);
    return false;
}
```

DateTime widget

Path: ***Widget Gallery> Basic> Controls***

Use this widget to display and edit current date and time .

In the **Properties** pane different formats are available for representing date and time.



Time options

For the **Time Spec** property select which time the widget will show at runtime.

Option	Description
local	shows local time, the time of the HMI device where the project is running
global	shows Global Time (GMT)
server	shows time information as handled by the server side of the HMI device

Time and Date placeholders

You can use placeholders to freely define the Time and Date format

Date	Description
d	the day as number without a leading zero (1 to 31)
dd	the day as number with a leading zero (01 to 31)
ddd	the abbreviated localized day name (e.g. 'Mon' to 'Sun')
dddd	the long localized day name (e.g. 'Monday' to 'Sunday')
M	the month as number without a leading zero (1-12)
MM	the month as number with a leading zero (01-12)
MMM	the abbreviated localized month name (e.g. 'Jan' to 'Dec')
MMMM	the long localized month name (e.g. 'January' to 'December')
yy	the year as two digit number (00-99)
yyyy	the year as four digit number

Time	Description
h	the hour without a leading zero (0 to 23 or 1 to 12 if AM/PM display)
hh	the hour with a leading zero (00 to 23 or 01 to 12 if AM/PM display)
m	the minute without a leading zero (0 to 59)
mm	the minute with a leading zero (00 to 59)
s	the whole second without a leading zero (0 to 59)
ss	the whole second with a leading zero where applicable (00 to 59)
AP or A	use AM/PM display. A/AP will be replaced by either "AM" or "PM"
ap or a	use am/pm display. a/ap will be replaced by either "am" or "pm"

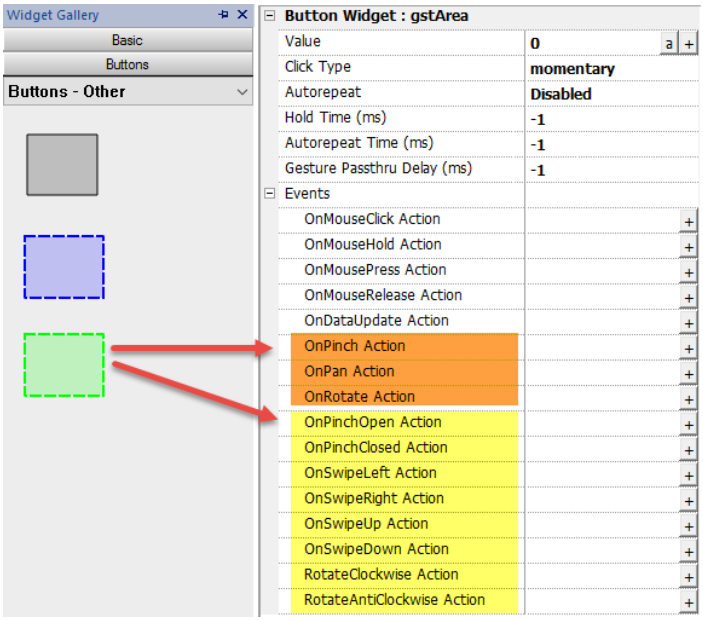
Regional Settings

You can use even the SHORT-DATE or the LONG-DATE placeholders to use the format defined inside the Regional Setting (see "Regional Settings" on page 86)

Gesture area widget

Path: ***Widget Gallery> Buttons> Others***

Gesture Area Widget is a hotspot button that generates gesture events.



Gesture Events	Description
OnSwipeLeft **OnSwipeRight**	An event is release when swipe gesture is detected
OnPinchOpen **OnPinchClose**	An event is release when pinch gesture is detected
RotateClockwise **RotateAntiClockwise**	An event is release when rotate gesture is detected
OnPan **OnPinch** **OnRotate**	A series of events released during the gesture. Only JavaScript can be used to service these events, through the JavaScript code the developer can manage the gestures events as he prefer. **WARNING: Only multi touch HMI devices can generate OnPinch and OnRotate events**

OnPan

```
boolean onGesturePan(me, eventInfo)
```

This event occurs when one point inside the area has pressed and a linear movement has been detected.

Parameter	Description
me	Object triggering the event.
eventInfo	**id** = Gesture id; it is used to identify different gestures. **running** = True except for last event delivered to notify gesture completion. **dx** = Total X axis movement in screen pixel units from initial touch position . **dy** = Total Y axis movement in screen pixel units from initial touch position.

OnPinch

```
boolean onGesturePinch(me, eventInfo)
```

This event occurs when two points inside the area have been pressed and a linear movement has been detected.

Parameter	Description
me	Object triggering the event
eventInfo	**id** = Gesture id; it is used to identify different gestures. **running** = True except for last event delivered to notify gesture completion. **dx** = Total X axis movement in screen pixel units from initial touch position. It represents the distance change between fingers. Positive value means that the distance is increasing; negative value means that the distance is decreasing. This amount may be used to control a zoom value. **dy** = Total Y axis movement in screen pixel units (see dx).

OnRotate

```
boolean onGestureRotate(me, eventInfo)
```

This event occurs when two points inside the area have been pressed and a rotate movement has been detected.

Parameter	Description
me	Object triggering the event
eventInfo	**id** = Gesture id; it is used to identify different gestures. **running** = True except for last event delivered to notify gesture completion. **drot** = How many degrees (0/360) have been added since the previous event. **trot** = Total degrees (0/360) of the entire movement. Positive numbers meaning clockwise rotation, negative anticlockwise rotation.

Gesture events pass thru

To use a widget (e.g. a button or a slider) covered from a gesture object, you have to keep pressed the widget 200 mSec to move the control to the underlying object. The time that must be waited for to send the command to the underlying object

can be modified from the “Gesture Passthru Delay” parameter that is available in the advanced properties view.

Parameter	Description
Gesture Passthru Enabled	Enable the possibility to pass gesture events to underlying widgets after a configurable delay. User has to keep pressed the finger and then execute the gesture. **default** = Use the value defined in the project properties. See "Project" on page 80 **true** = Gesture passthru enabled **false** = Gesture passthru disabled
Gesture Passthru Delay (ms)	The time that must be waited for to send the command to the underlying object 0/500 mSec -1 Use the delay defined in the project properties. See "Project" on page 80

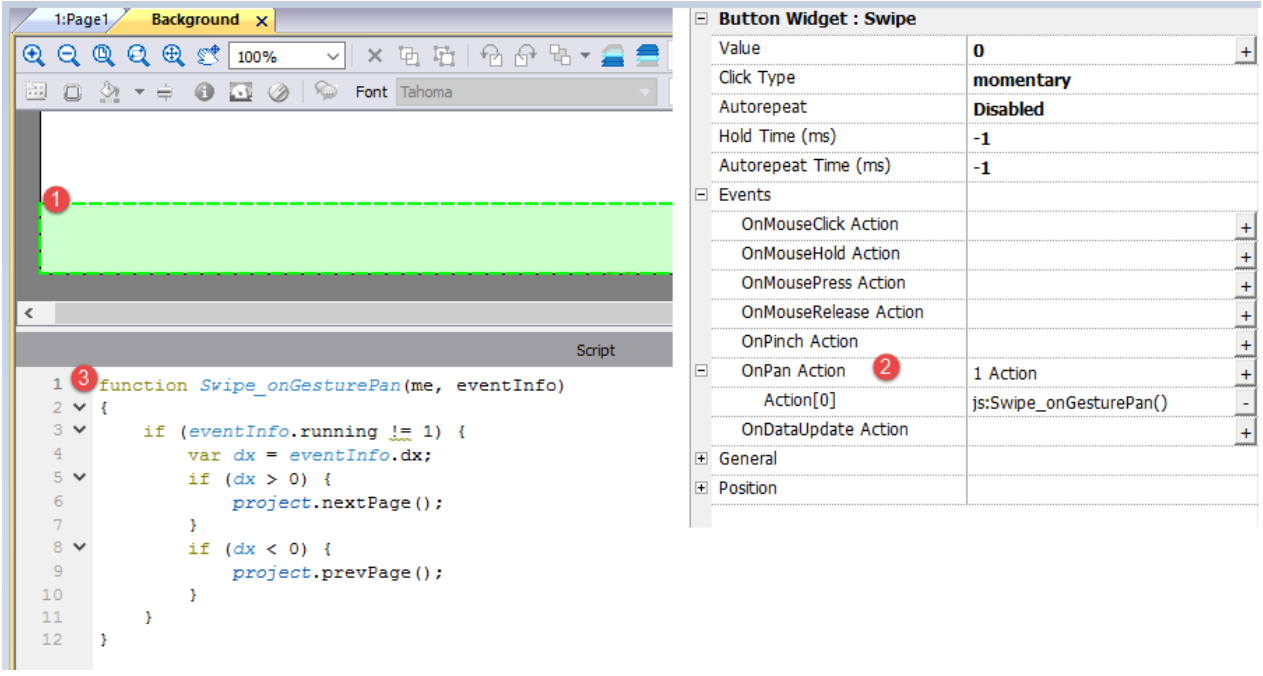
Examples of using gesture events in association with JavaScript

Here some example of using gesture events in association with JavaScript code to identify gestures and program the requested actions

Swipe Gesture

How to recognize a "swipe" gesture to change page in the application.

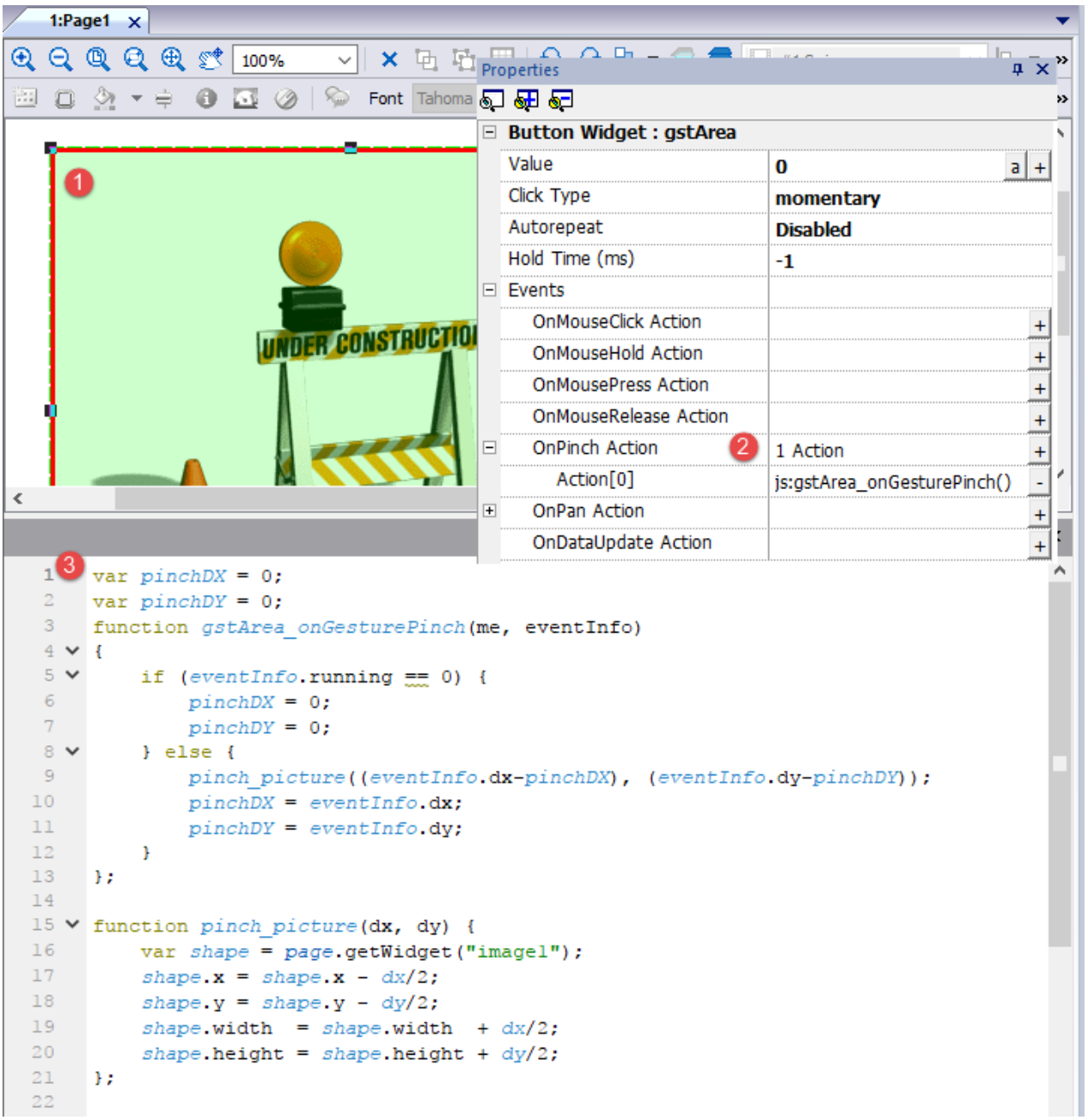
1. Put a Gesture area widget into the page
2. Configure the OnPan Action to trigger a JavaScript function
3. Write the JavaScript code that recognize and manage the swipe gesture



Pinch Gesture

How to recognize a "pinch" gesture to resize an image.

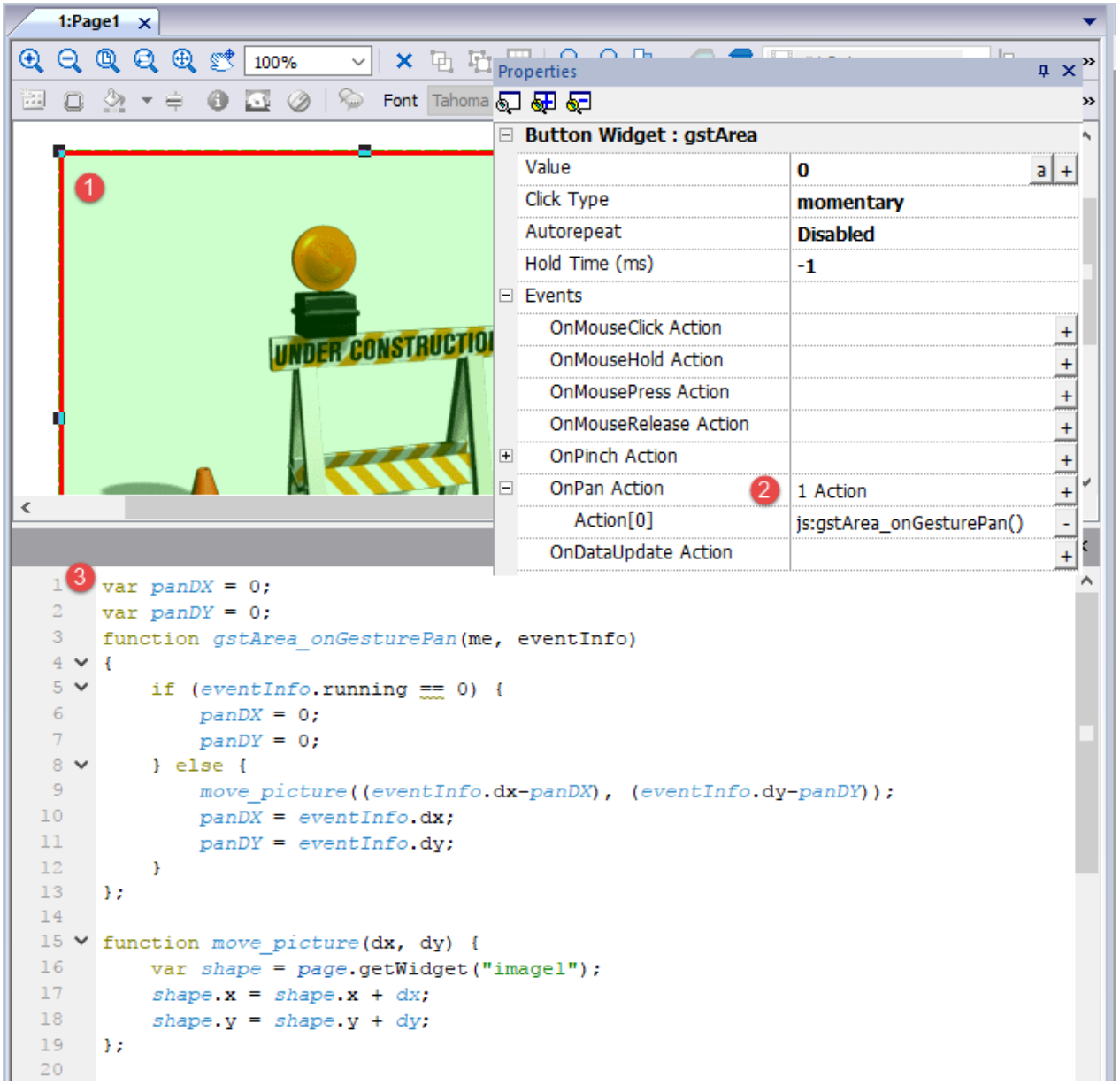
1. Put a Gesture area widget into the page over the image
2. Configure the OnPinch Action to trigger a JavaScript function
3. Write the JavaScript code that recognize and manage the pinch gesture



Pan Gesture

How to recognize a "pan" gesture to move an image.

1. Put a Gesture area widget into the page over the image
2. Configure the OnPan Action to trigger a JavaScript function
3. Write the JavaScript code that recognize and manage the pan gesture



JavaScript function block widget

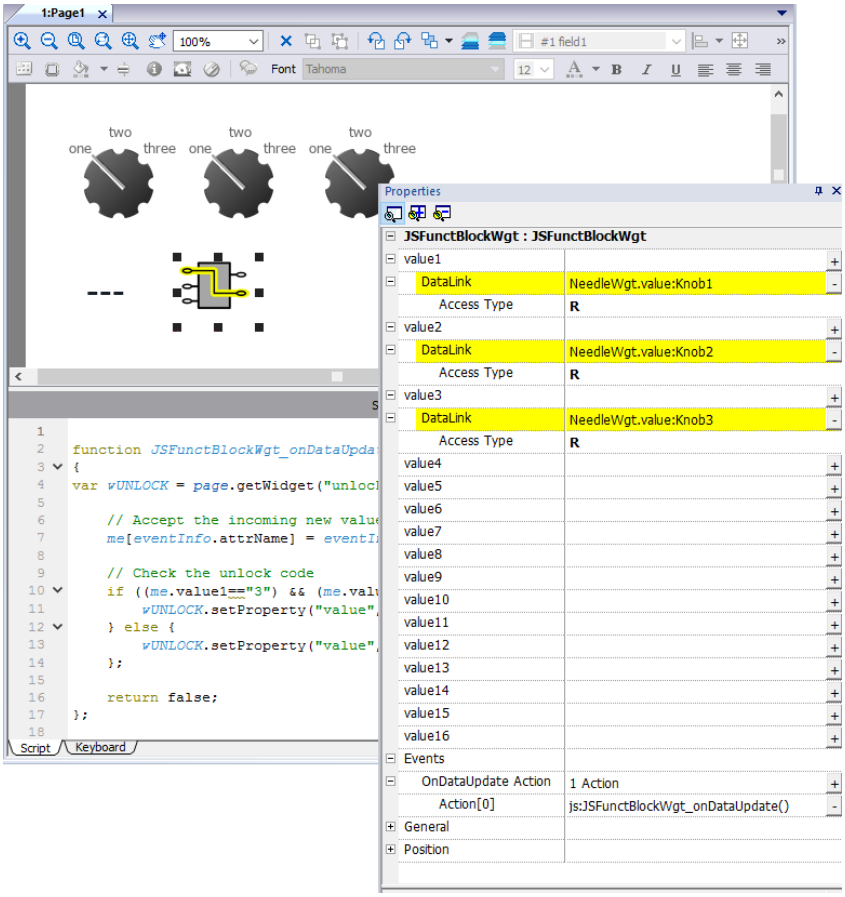
Path: ***Widget Gallery> Basic> JSFunctionBlock***

JavaScript Function Block is a widget that contains JavaScript logic that is executed when tags values change.

Parameter	Description
value1 **...** **value16**	Objects that will trigger the OnDataUpdate action.
OnDataUpdate	Action that will be executed when a change of an associated value is detected

Note: This widget is rendered only in LRH SW, and it is not rendered in the HMI device.

Example:
A JavaScript code that check the combination lock of three selectors



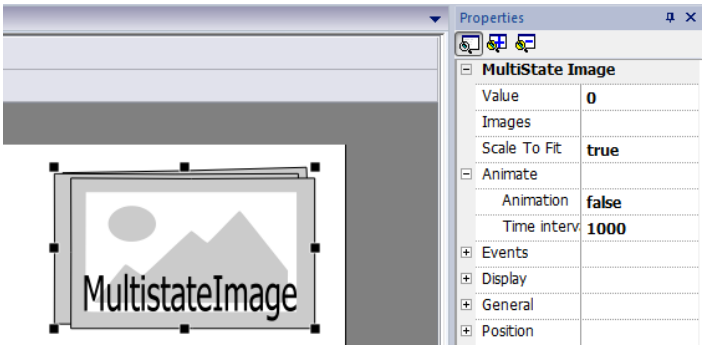
```
Script
1
2   function JSFunctBlockWgt_onDataUpdate(me, eventInfo)
3   {
4   var wUNLOCK = page.getWidget("unlock")
5
6       // Accept the incoming new value
7       me[eventInfo.attrName] = eventInfo.newValue;
8
9       // Check the unlock code
10      if ((me.value1=="3") && (me.value2=="3") && (me.value3=="3")) {
11          wUNLOCK.setProperty("value", "Unlock!");
12      } else {
13          wUNLOCK.setProperty("value", me.value1+"-"+me.value2+"-"+me.value3);
14      };
15
16      return false;
17  };
18
Script  Keyboard
```

See "Widget events" on page 474for the description of the onDataUpdate parameters

Multistate Image widget

Path: ***Widget Gallery*** *>* ***Basic*** *>* ***Images***

Use this widget to display an image from a collection based on the value of a tag used as Index. You can use this widget also for simple animations.

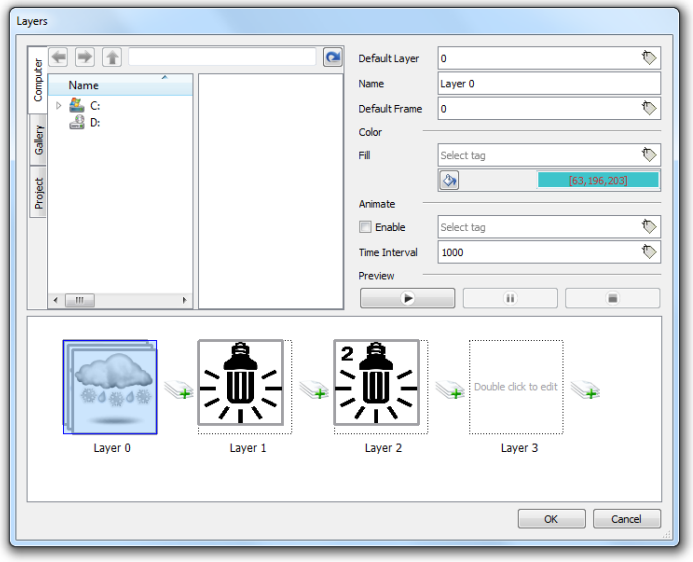


Parameter	Description
Value	Index of image to display. For example, set Value=0, to display the image with index 0 in the image collection.
Images	Images collection with associated index.
Animate	Set to true, to enable a slide show.
Time interval	Interval between images in the slide show.

Multistate Image Multilayer widget

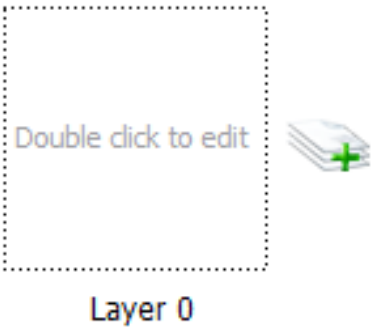
Path: ***Widget Gallery> Basic> Images***

Use this widget to create different animations and select the most suitable at runtime.

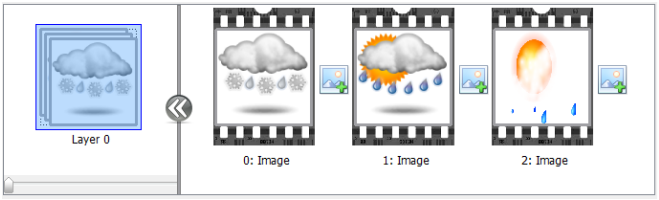


Setting up widget layers

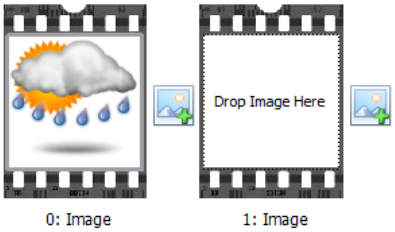
1. Open the **Layers** dialog from the **Properties** pane.
2. Click **+** to add as many layers as you need.



3. Double click on each layer to add as many images as you want to include in the layer.



4. Drag and drop images into the frame to add it to current layer.



5. Define widget properties.

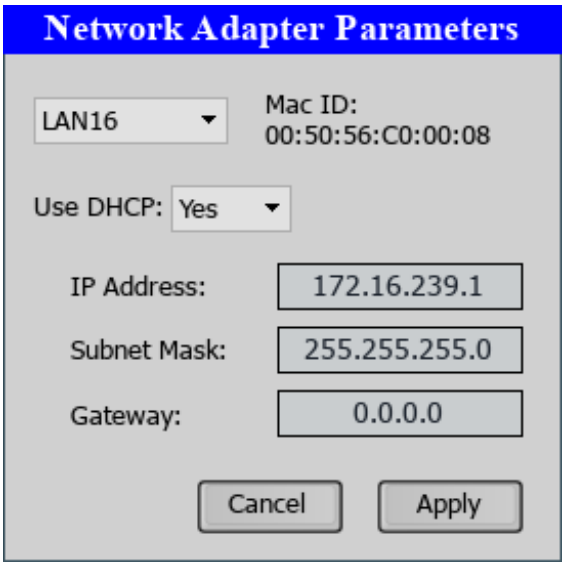
Parameter	Description
Default Layer	Layer shown at runtime.
Name	Name of selected layer.
Default Frame	Frame shown when current layer is displayed.
Color / Fill	Fill color for images of current layer.
Animate	Enables slide show for active layer. Animations can be started/stopped at runtime attaching it to a tag.
Time Interval	Time interval of slide show, if enabled.
Preview	Slide show simulation.

Note: **Default Layer**, **Default Frame**, **Color** and **Fill** can be changed at runtime, attaching the to a tag.

Network Adapters widget

Path: ***Widget Gallery*** *>* ***Basic*** *>* ***Control***

Use the IP Widget to set the network adapters parameters.



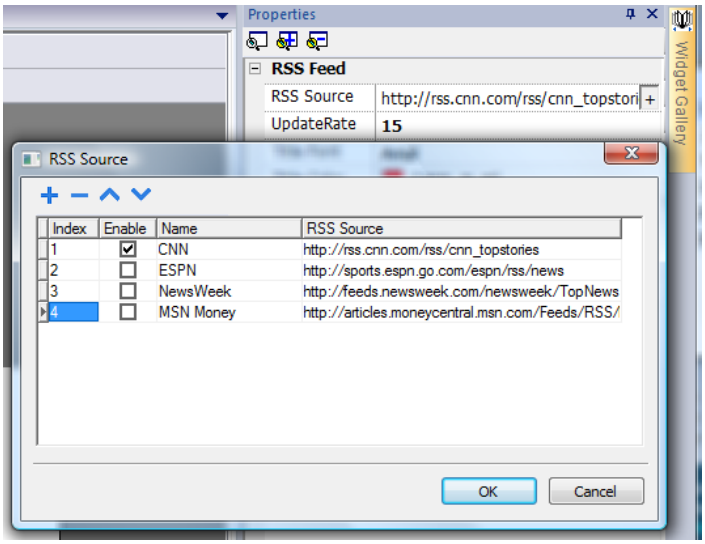
The system variable Network->Status contains the result of the last operation performed by the IP Widget (see "Network variables" on page 132for details)

RSS Feed widget

Path: ***Widget Gallery> Media> RSSFeed Source***

Use this widget to display on the HMI device your favorite RSS feeds directly from the Internet.

Parameter	Description
RSS Source	Feed URL Note: Feed sources cannot be modified at runtime.
UpdateRate	Refresh time

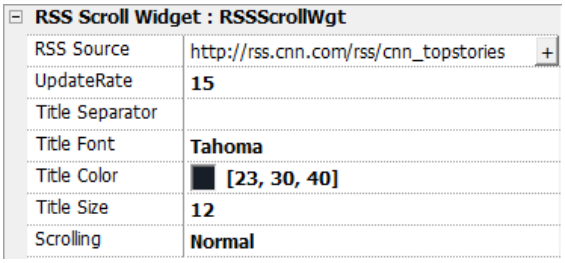


The RSS Feed widget has been specifically designed to work with Pocket Internet Explorer.

Scrolling RSS Feed widget

Path: ***Widget Gallery> Media> RSSFeed Scroll***

Use this version of the main RSS Feed widget to display highlights inside a text line using a smoothing scrolling text.



This widget has additional properties.

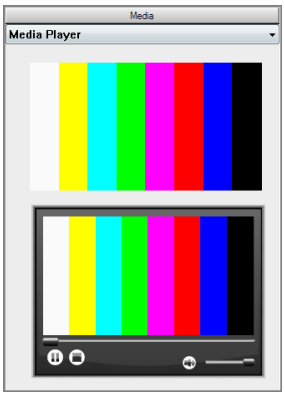
Parameter	Description
Scrolling	Scrolling speed
Title Separator	Separator character between highlights

Media Player widgets

Path: ***Widget Gallery> Media> Media Player***

Use these widgets to play videos from a playlist. The video files can be stored on a USB drive, on the Flash card or an SD Card.

Two widgets are available: one includes a multimedia frame with buttons to play and stop the video, the other is a plain frame where the video is played without user control.



Parameter	Description
Media Player List	Open Windows file browser for selecting video files to collect in the play list. Selected files will be downloaded to the HMI device together with the project. When a USB device or an SD Card has been selected, files must be placed in a subfolder “mediafiles” of the external memory media. Video files will be played according to filename alphabetical order. **Ensure you have the commercial rights of the multimedia files.**
Loop Style	Define how the video is played. • **NoLoop**: plays all the videos in the playlist, then stops. • **LoopOne**: repeats the first video in the playlist. • **LoopAll**: repeats the entire playlist. • **Random**: plays the videos in a random order.

Note: The Media Player widget only works with some HMI devices (see "HMI devices capabilities" on page 541). It doesn’t work the LRH SW Client.

Note: You can have only one Media Player widget in a page.

Supported video encoding

Two groups of codecs are supported:

- DSP based video codecs
- Software video codecs

List of HMI devices that support the DSP (video hardware acceleration) is available on "HMI devices capabilities" on page 541.

DSP video codecs

These include:

- H264 using AVI/MP4 container, CABAC off and Level 3 (suggested)
- MPEG4 using MP4 container

BSP v1.0.269 or greater is required

Software video codecs

This is only:

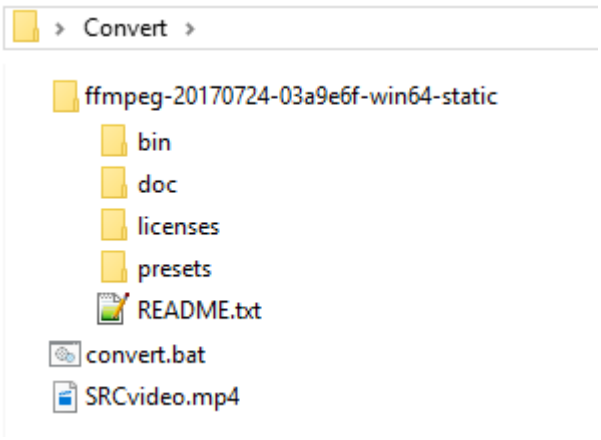
- Microsoft MPEG4 v3 using an AVI container.

Be aware that video performance are depending from the chosen resolution, bit rate and device capabilities. If video rendering is not smooth, try to reduce the resolution or the bit rate of your video.

The videos encoded with Microsoft MPEG4 v3 are not using the hardware acceleration and have more limitations. To prevent the videos from running jerky, a maximum resolution of 640x512 pixels and a bit rate of 1300 kb/s are suggested. In addition, the size of the Media Player widget used on the page should have the same size as the videos in the play list, in order to avoid up scaling and down scaling. Audio is not supported.

Converting a video

The FFMPEG (www.ffmpeg.org) can be used to convert a video into the correct codec supported from the HMI device. Using the folder structure of the below picture, the following batch file could be used to convert any video file.



convert.bat

```
@echo off
set FFMPEG=ffmpeg-20170724-03a9e6f-win64-static\bin\ffmpeg.exe

%FFMPEG% -i SRCvideo.mp4 ^
         -y ^
         -an ^
         -s 240x160 ^
         -b:v 4200k ^
         -maxrate 4200k ^
         -c:v libx264 ^
         -profile:v baseline ^
         -level:v 3 ^
         -bufsize 3000k ^
         -minrate 0 ^
         -f avi ^
         -preset slow ^
         HMIvideo.avi

pause
```

Now you can open the converted video with a standard video player, such as Windows Media Player and check the quality. You can add the resulting video to the play list of the Media Player widget.

Note : The FFMPEG tool is not distributed with the LRH SW.

Using Media Player in JavaScript

The Media Player widget can be also referenced in JavaScript programs with the following syntax:

```
//get the mediaplayer widget.
var mediaWgt = page.getWidget('MediaPlayerWgt2');
//load the play list
mediaWgt.setProperty('medialist', '/Storage Card/demo_3.avi,/Storage Card/video1_
3.avi');
// set the loopstyle 0 - noloop, 1 - loop one, 2- loop all, 3 - random
mediaWgt.setProperty('loopstyle', 2);
//start playing the first file.
mediaWgt.mediapath = '/Storage Card/demo_3.avi';
```

See "JavaScript " on page 469 for details on how to work with JavaScript.

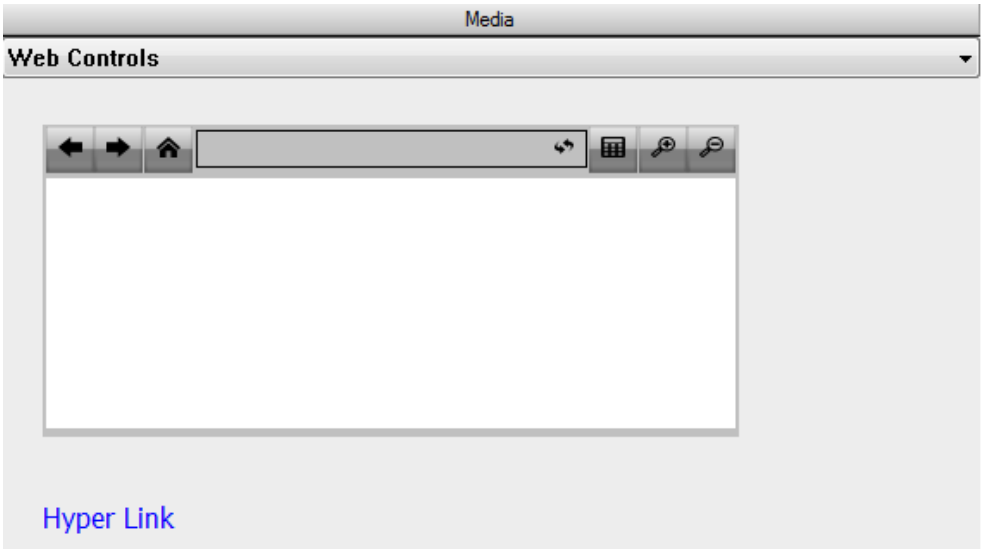
Browser widget

Path: ***Widget Gallery> Media> Web Controls***

Use this widget to embed web pages into your HMI device pages. This is an HTML5 compatible browser widget based on the WebKit engine.

Note: The WebKit library is available as a plugin (see "Plug-in" on page 79 for details) to download to the LRH SW HMI Runtime only when required.

Important: This widget is not supported by MIPS based devices.



Parameter	Description
Home Page	Default URL to open when widget is shown on the page.
Zoom to Fit	Automatically scales content to the size of view area.
Time out	Page load timeout in seconds.
Clear History	Automatic history clear on load
Scroll	Shows/hides scrollbars
Show Progress cursor	Shows/hides loading cursor

This allows you to save around 3 MB of space if the widget is not required in your project.

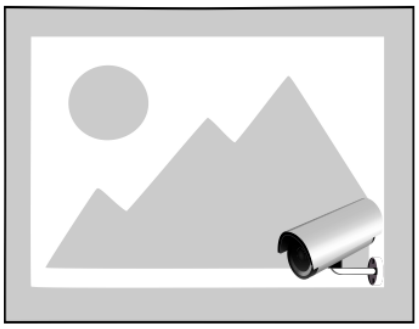
An **Hyper Link** widget is available to create pages hyperlinks. Once clicked these links notify to the browser widget that a particular web page is to be loaded.

Important: HTTPs protocol is not supported.

IP Camera widgets

Path: ***Widget Gallery> Media> IP Camera***

Use these these widgets to show images captured from an IP Camera or a video stream.



Parameter	Description
Camera URL	URL of the IP Camera when used in JPEG format.
Refresh Rate	Number of JPEG images for second allowed. Max rate = 1 fps.
User Name	Name of user allowed to access the camera. Set this parameter when access to the camera is password protected.
Password	Password to access the camera.
MJPEG Camera URL	URL of MJPEG streaming (for example, http://192.168.0.1/video.cgi)

When this widget is used to stream HTTP MJPEG, **Camera URL** and **Refresh Rate** are ignored.

Performance of streaming is not fixed and depends on many factors such as: frame size, frame compression level, CPU of HMI device, quality of IPCamera. Based on these factors the widget can reach up to 25 fps.

You can add multiple IP Camera widgets, but this will reduce the frame rate for each widget.

Supported IPCameras

The following IP Cameras have been tested so far:

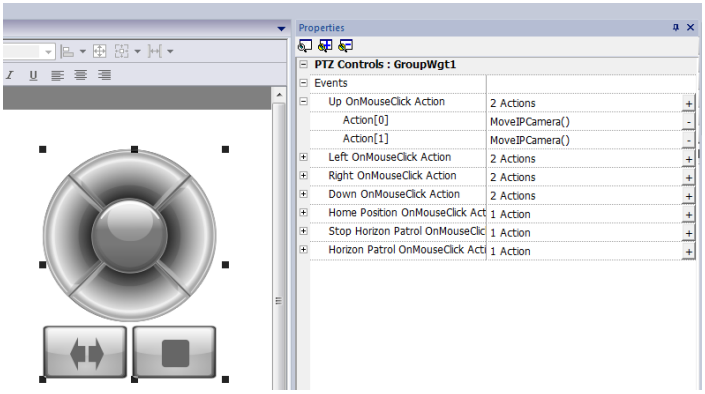
IPCamera	Protocol	URL
Apexis APM-J901-Z-WS PTZ IP Camera	MJPEG HTTP	http://{ip_address}/videostream.cgi http://{ip_address}/snapshot.cgi
AXIS M3027-PVE Network Camera	MJPEG HTTP	http://{ip_address}/axis-cgi/mjpg/video.cgi http://{ip_address}/axis-cgi/jpg/image.cgi
DAHUA DH-IPC-HD2100P-080B 1.3mp Outdoor Vandalproof	HTTP	http://{ip_address}:9988/onvif/media_service/snapshot
D-Link DCS-5605 PTZ	MJPEG	http://{ip_address}/video/mjpg.cgi
D-Link DCS-900W IP Camera	MJPEG	http://{ip_address}/video.cgi
D-Link DCS-932L	MJPEG	http://{ip_address}/video.cgi

IPCamera	Protocol	URL
Edimax IC-7100P PTZ	MJPEG HTTP	http://{ip_address}/mjpg/video.mjpg http://{ip_address}/picture.jpg
Foscam FI8916W	MJPEG HTTP	http://{ip_address}/videostream.cgi http://{ip_address}/snapshot.cgi
Foscam FI9803 EP	MJPEG	http://{ip_address}:88/cgi-bin/CGIStream.cgi?cmd=GetMJStream&usr={user}&pwd={pass} NOTE: • port 88 may be different as per IP Camera settings • {user} = username defined into IP Camera settings • {pass} = password defined into IP Camera settings
Hamlet HNIPCAM IP Camera	MJPEG HTTP	http://{ip_address}/video.cgi http://{ip_address}/image.jpg
MOXA VPort 254 (Rugged 4-channel MJPEG/MPEG4 industrial video encoder)	MJPEG HTTP	http://{ip_address}/moxa-cgi/mjpeg.cgi http://{ip_address}/moxa-cgi/getSnapShot.cgi?chindex=1
NVS30 network video server	MJPEG HTTP	http://{ip_address}:8070/video.mjpeg http://{ip_address}/jpg/image.jpg
Panasonic WV-Series Network Camera	MJPEG	http://{ip_address}/cgi-bin/mjpeg
Ubiquiti UniFi Video Camera	HTTP	http://{ip_address}:7080/images/snapshot/camera/{camera_guid}?force=true NOTE: • {camera_guiID} can be found into IP Camera Webpage • port 7080 may be different as per IP Camera settings
Zavio F3210 2MP Day & Night Compact IP Came	MJPEG HTTP	http://{ip_address}/stream?uri=video.pro3 http://{ip_address}/cgi-bin/view/image?pro_0 NOTE: • MJPEG video streaming can be configured selecting "video profile 3" with 640x480 resolution into IP Camera settings.

PTZ Controls widget

PTZ (pan–tilt–zoom) cameras are cameras capable of remote directional and zoom control.

The PTZ Controls widget uses the MoveIPCamera action to send HTTP/cgi commands to the PTZ IP Camera.



Parameter	Description
Camera URL	URL of IP Camera
User Name	Name of user allowed to access the camera. Set this parameter when access to the camera is password protected.
Password	Password to access the camera.
Command	Command to send to the PTZ controller (for example, decoder_control.cgi?command=0)

Authentication methods

The authentication method is automatically set by the camera web server to which the widget connects. Authentication methods supported are:

- Basic
- NTLM version 1
- Digest-MD5

Web Browser

On the Web Browser, only the "Basic Authentication" mode is supported. When used, the IP Camera with authentication shows a pop dialog to enter login and password.

Widget is supported by Chrome and Firefox, we found issues using the current version of the Edge browser.

Table widget

Path: **Widget Gallery**> **Basic**> **Table**

Use this widget to create a table with data provided from a data source.

To configure a table:

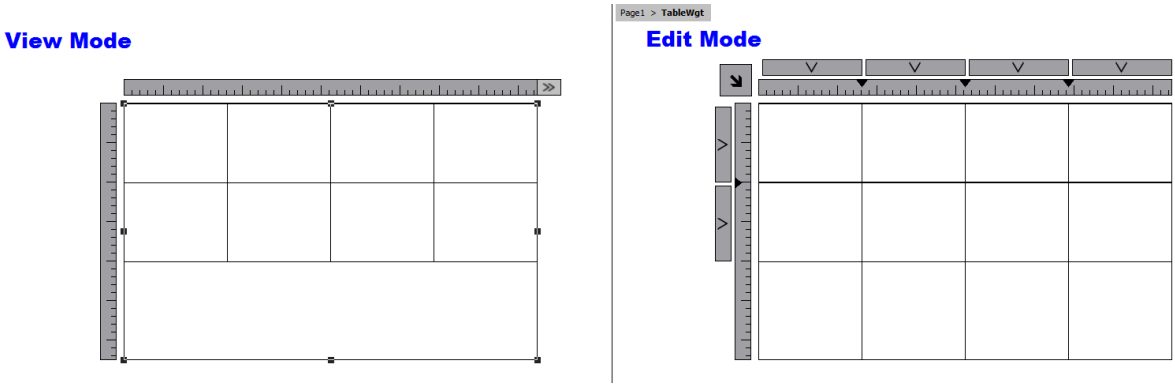
1. Put a table widget on the screen and configure the template of the table.
2. Add widgets into cells to configure one or more rows that will be used as row templates when the table will be filled with data provided from the data source.
3. Select a data source that will be used to fill the rows of the table
4. Define the links from widgets and data source.

Configure the table widget

Table widget has two states:

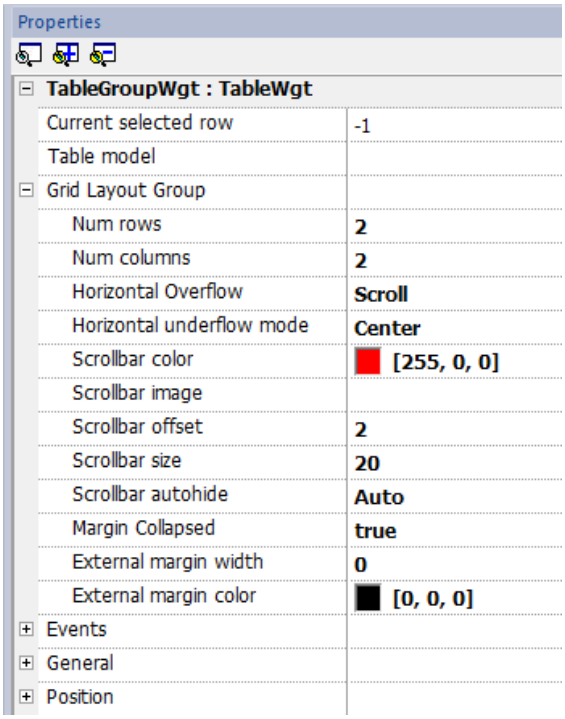
- View mode
- Edit mode.

Click on the table to manage the widget in view mode, double click to enter in the edit mode. To exit and return to view mode click outside the table.



View Mode

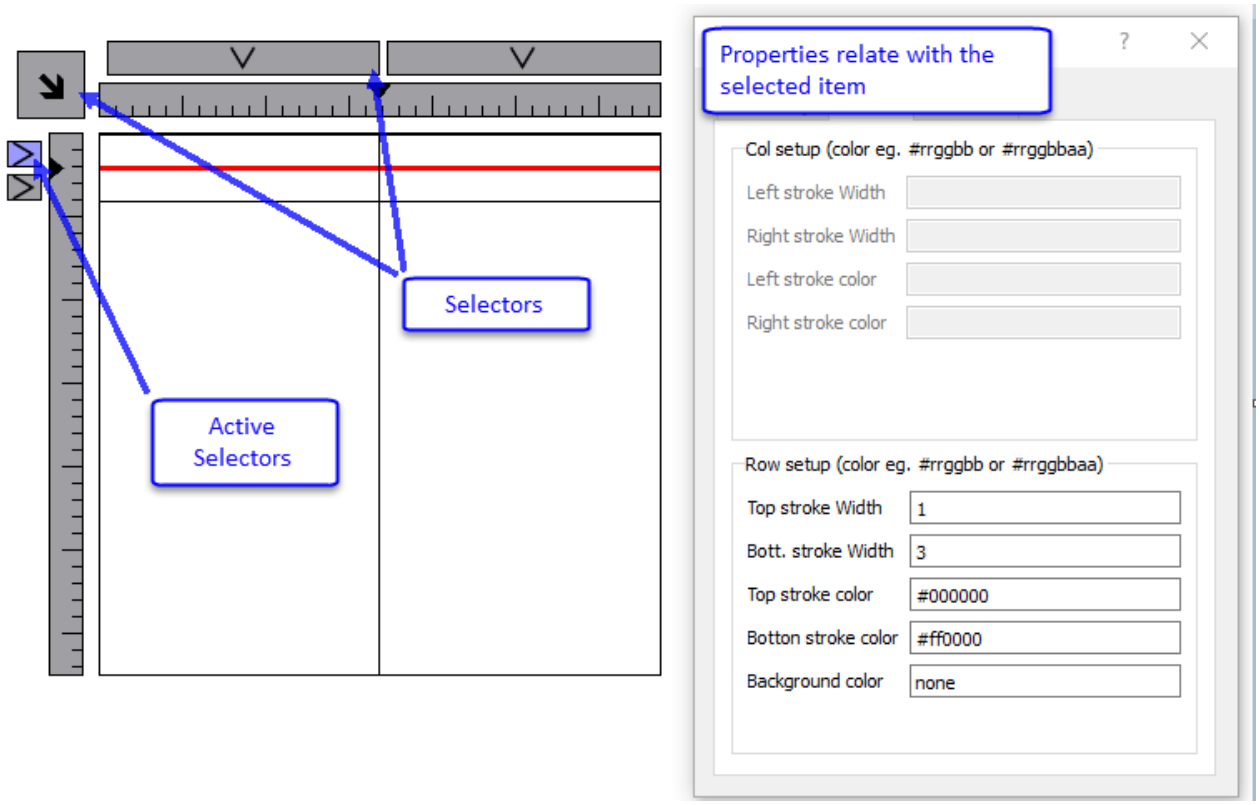
In view mode, you can configure the table layout. Drag and drop the table onto the page, resize the table, define number of template rows, number of columns and the main table properties.



Edit Mode

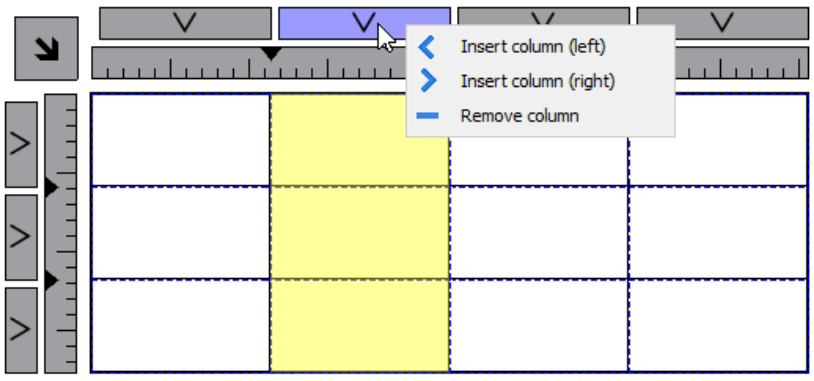
In edit mode, it is possible configure the format and the content of each cell of the table. Each row of the table will act as a row template.

To configure the look of the table, click on table's selectors to select the item to configure.



Add or remove rows or columns

To add or remove rows or columns, double click over the grid to enter in edit mode and right click over column or row selector to open the context menu.



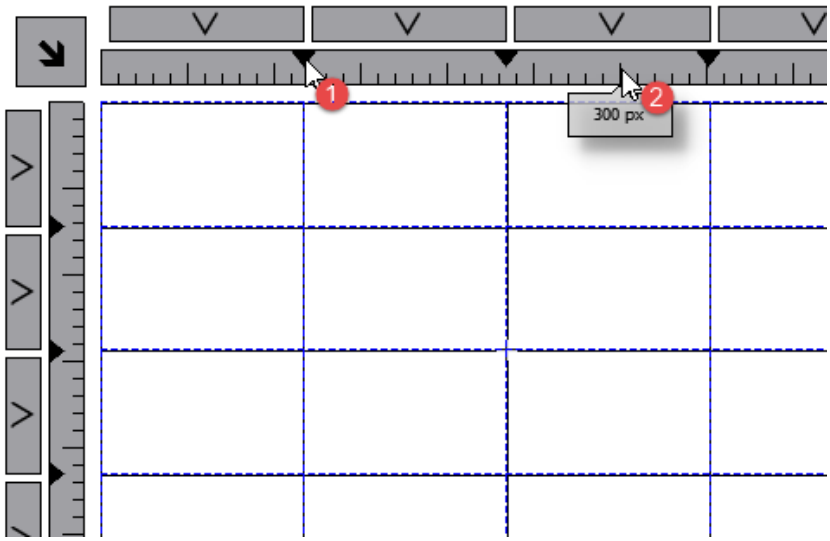
Merge or split rows or columns

To merge or split rows or columns, double click over the grid to enter in edit mode and move the cursor over the ribbons:

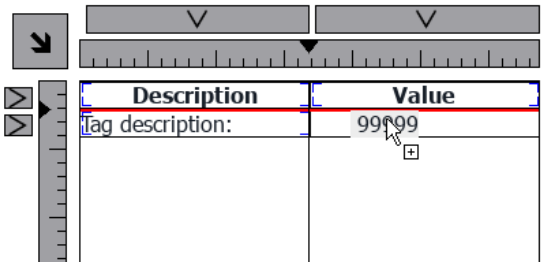
- Double click the black triangle to merge the two adjacent rows or columns (1)

  Note that merge is possible only with an empty row or column.

- Double click on ribbon to split the selected row or column (2)



To configure the contents of cells, drag and drop the widgets inside the cells.



If you need more widgets inside a single cell, create a group of widgets and copy the group from the page to the cell.

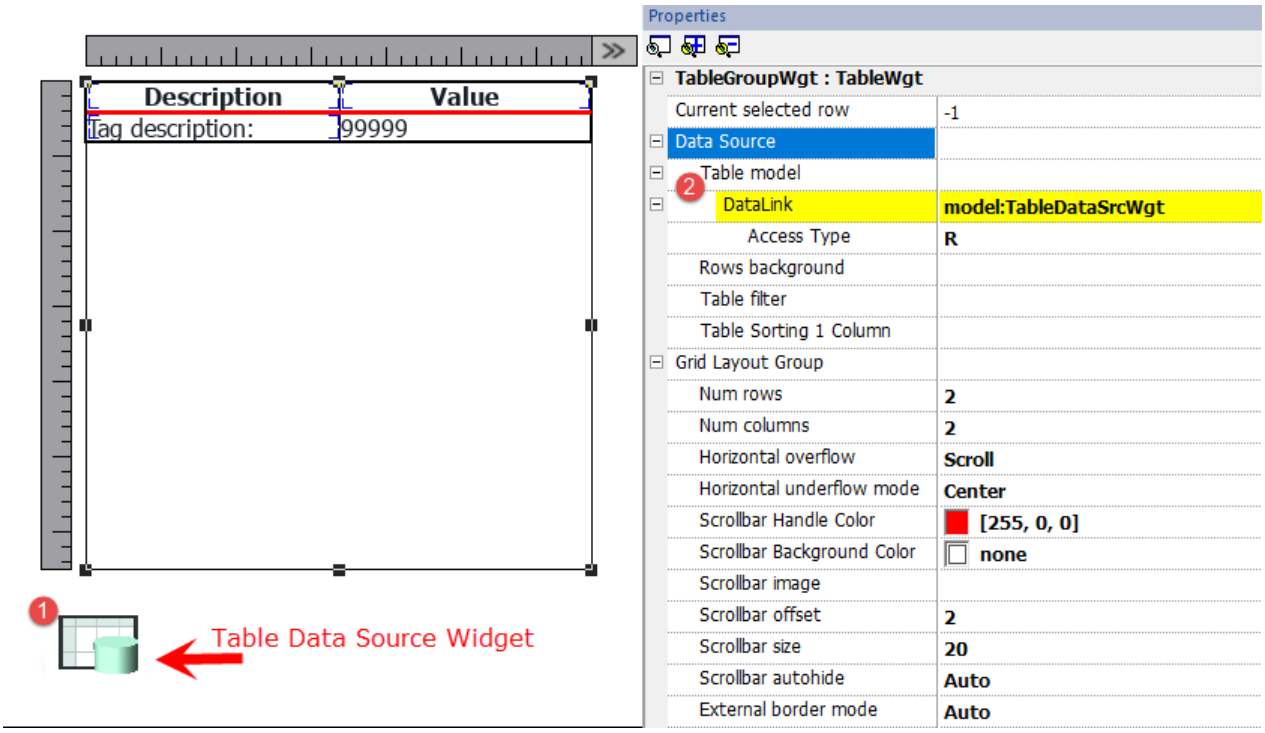
Configuring the data source

The data source, that provide the data to fill the table, could be a Table Data Source Widget or a JavaScript Object.

Table Data Source Widget

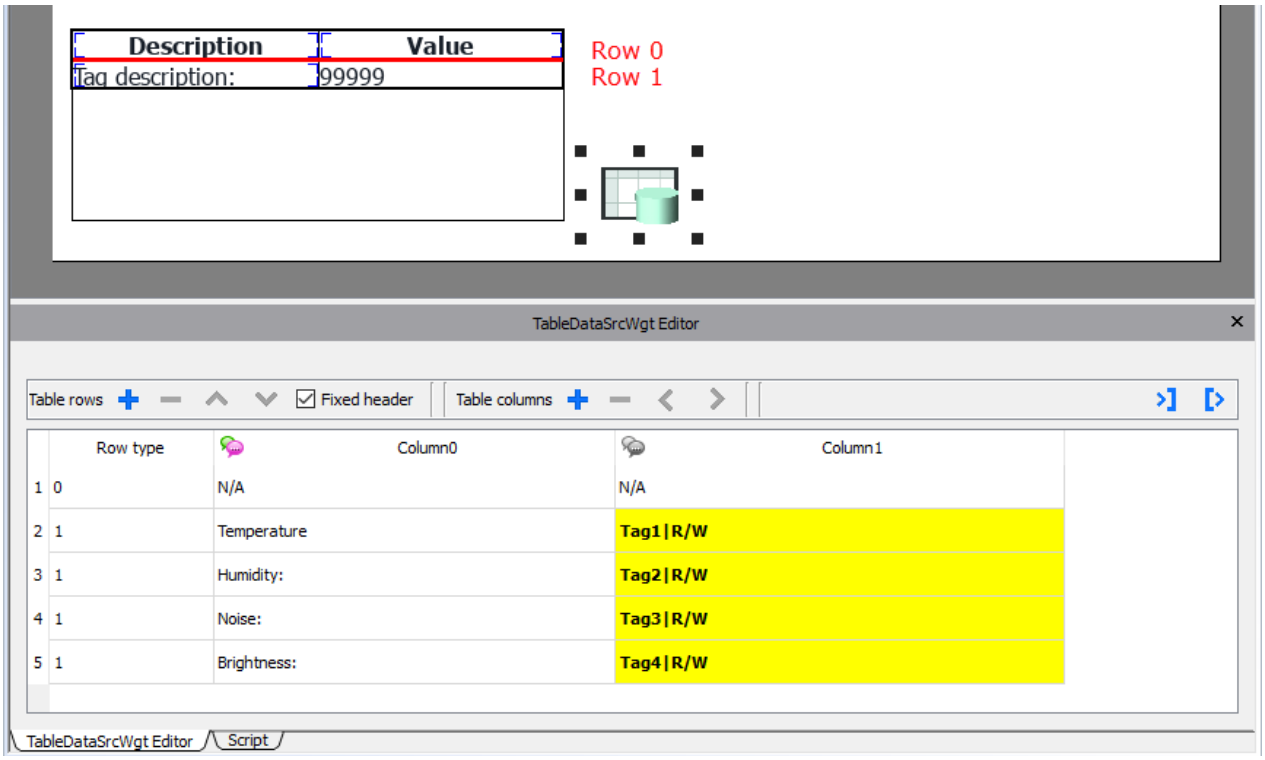
Path: **Widget Gallery**> **Basic**> **Table**

1. Drag and drop a *Table Data Source Widget* onto the page
2. Set the *Table Model* parameter to link at the data source.



Select the Data Source and inside the TableDataSrcWgt Editor add the rows and columns that are needed. In the following example, we have defined two row templates:

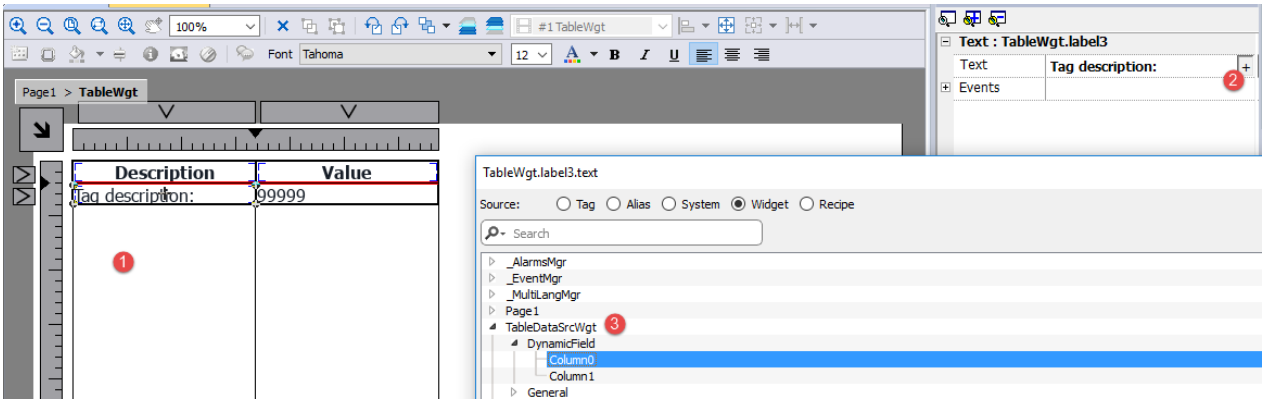
- Row 0
  Header of the table. Contains only static text.
- Row 1
  Template of rows with data. On the first column we added a label that will contain the description and on the second column a field that will contain the value.



Each row must be assigned a row type. The row will take on the format of the corresponding row template. Widgets that were placed in each cell of the row template will appear in rows of that type.

Define links with data source

1. Double click over the Table widget to enter in edit mode and select a widget
2. Select the property that is to be read from the data source
3. Select the column of the data source that will provide the data



The below picture is showing how our example will be rendered at runtime

Description	Value
Temperature	111
Humidity:	222
Noise:	333
Brightness:	444

Fixed Header

If you want the first row will be not scrollable, check the *"Fix Header"* check box on Data Source toolbox or set true the "Show Header" propert inside the Data Source properties panel (note the parameter is available only in advanced view).

Column override

You can use an array of integers to define or modify the columns order at runtime. When you use this property, be sure to attach an array of integer and set the index to -1 (to select the entire array).

Column override (array of int):

0	1	2	3	4	5	6	7	8

Description:	Col 1	Col 2	Col 3	Col 4
00	1	2	3	4
Row 1	Data 1	Data 11	Data 111	Data 1111

Column override (array of int):

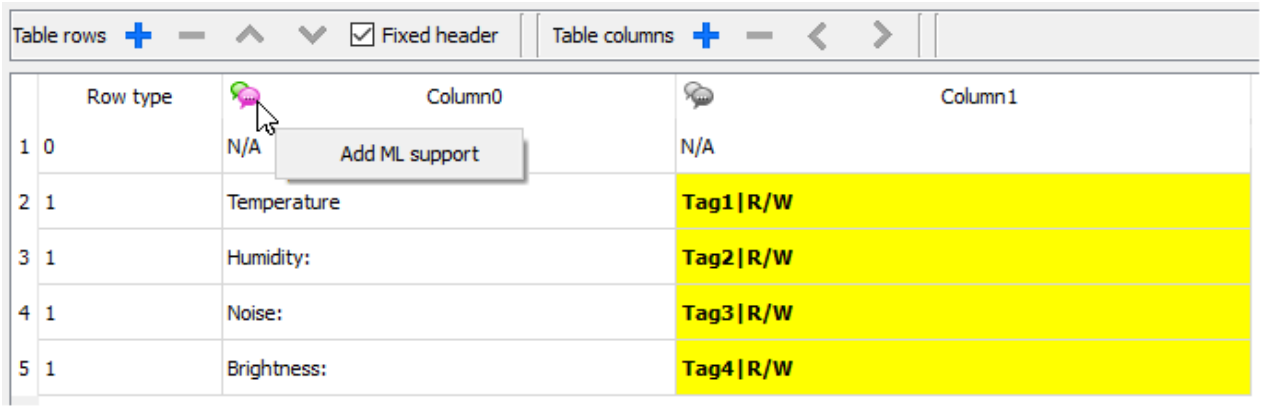
0	1	3	3	7	5	6	7	8

Description:	Col 1	Col 3	Col 3	Col 7
00	1	3	3	7
Row 1	Data 1	Data 111	Data 111	fdgfd

Multilanguage

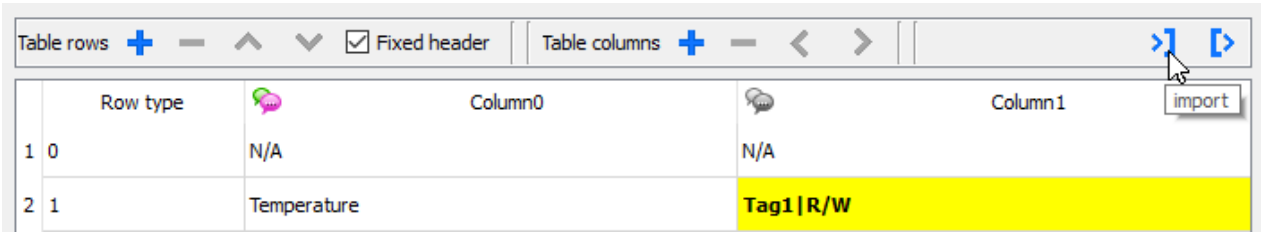
To enable the Multilanguage support right click on the Multilanguage icon of the column. The icon will change color to indicate that the support is enabled.

Avoid enabling the Multilanguage support when not necessary to better performance.



Import/Export Data Source

The configuration of the Data Source can be imported/exported using xml files



JavaScript Object

In alternative to the Data Source Widget, for data to fill the table could be provided from a JavaScript Object. In this case, we have to fill an array of elements with the data to use and assign the array to the table widget.

```
var myTable = page.getWidget("TableWgt1");
```

```
myTable.model = model;
```

model is an array of elements with the table definition and data. The first element of the array will contain the template of the rows while the other elements will contain the data to fill in the rows of the table

```
model[0] = row_templates;     // row templates
model[1] = row_data1;         // data of the row1
model[2] = row_data2;         // data of the row2
model[3] = row_data3;         // data of the row3
model[4] = row_data4;         // data of the row4
model[5] = row_data4;         // data of the row5
```

The **row templates** is a multi dimensional array where each array defines the datalink of one template row.

On the below example, we have a template for two rows.

```
var row_templates = {
    _h : [
            [ []       , []        ],   //rowType = 0
            [ ["text"] , ["value"] ]    //rowType = 1
        ]
}
```

The first row has two columns that do not contain data links. We use this template for the header on the first row of our table.

The second row defines the template of one row with the “text” property of the widget into the first column and the “value” property of the widget into the second column. They will be dynamically filled using the data provided inside the model variable.

On the below example we define a **row of data**

```
var row_data = {
    _t : 1,
    _v : ["Temperature:", { _c : "dl" , s : "_TagMgr", a : "Tag1", i: 0, m : 2 }]
}
```

The first element is the row template to use while the second element is the array with the data to use. In our example “Temperature:” is the text to use inside the widget on the first column, while the other element is a datalink that will provide the value to fill the value property of the widget into the second column.

The datalink element:

Parameter	Description
_c : "dl"	Identify the element as a Datalink
s : "_TagMgr"	Specify the source of data is the Tag Manager
a : "Tag1", i: 0, m:2	Specify tag name and index (necessary when the tag is an array) and the read/write mode

Parameter	Description
	• m=0 is Read Only • m=1 is Write Only • m=2 is Read/Write

The below JavaScript code will generate the same table of the previous example using the Table Data Source Widget

```
var model = [];

var row_templates = {
    _h : [
            [ []       , []        ],   //rowType = 0
            [ ["text"] , ["value"] ]    //rowType = 1
        ]
}

var row_data1 = {
    _t : 0,
    _v : [],
    _h : true
}

var row_data2 = {
    _t : 1,
    _v : ["Temperature:", { _c : "dl" , s : "_TagMgr", a : "Tag1", i: 0, m : 2 }]
}

var row_data3 = {
    _t : 1,
    _v : ["Humidity:",    { _c : "dl" , s : "_TagMgr", a : "Tag2", i: 0, m : 2 }]
}

var row_data4 = {
    _t : 1,
    _v : ["Noise:",       { _c : "dl" , s : "_TagMgr", a : "Tag3", i: 0, m : 2 }]
}

var row_data5 = {
    _t : 1,
    _v : ["Brightness:",  { _c : "dl" , s : "_TagMgr", a : "Tag4", i: 0, m : 2 }]
}

model[0] = row_templates;
model[1] = row_data1;
model[2] = row_data2;
model[3] = row_data3;
model[4] = row_data4;
model[5] = row_data5;
```

```
var myTable = page.getWidget("TableWgt1");
myTable.model = model;
```

Note the first row (row_data1) contains the directive _h: true to avoid the first line will be scrollable.

```
var row_data1 = {
    _t : 0,
    _v : [],
    _h : true
}
```

Multilanguage

A multi languages text can be entered using the below element:

```
{ _c : "ml" , mltext : {"en-US" : "Temperature:" , "it-IT" : "Temperatura:"} }
```

Parameter	Description
_c : "ml"	Identify the element as a Multilanguage text
mltext : { ... }	List of couples: "ID Language":"Text" Example: • "en-US" : "Temperature:" • "it-IT" : "Temperatura:"

Example:

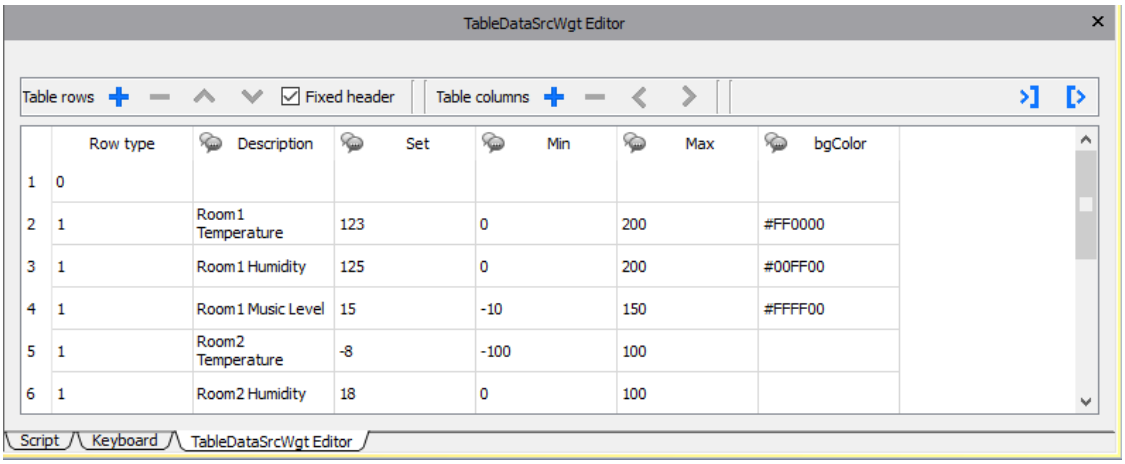
```
var row_data2 = {
    _t : 1,
    _v : [ { _c : "ml" , mltext : { "en-US" : "Temperature:",
                                    "it-IT" : "Temperatura:"}   },
           { _c : "dl" , s : "_TagMgr", a : "Tag1", i: 0, m : 2 }
         ]
}
```

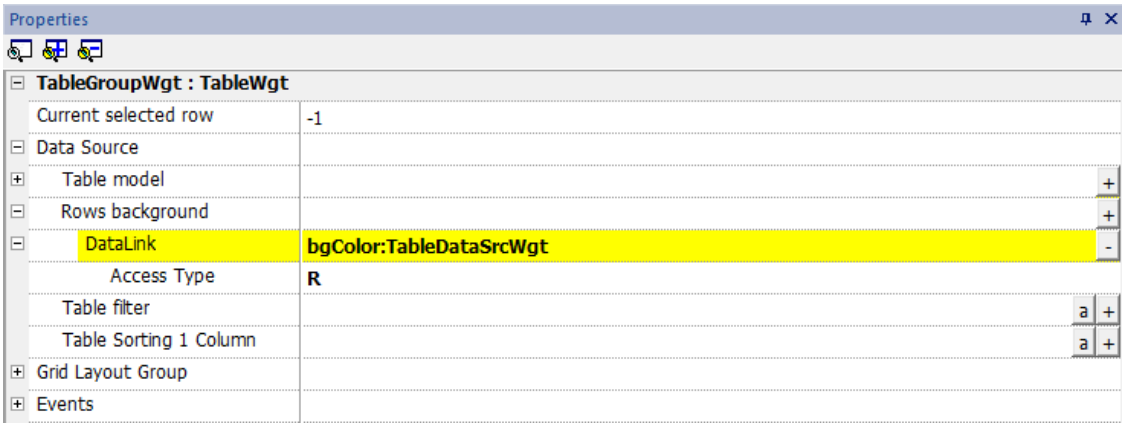
Row background color

Using the Rows background parameter is possible define the column of the Data Source Widget that will contains the background color of the associate row.

To configure the background color of the rows of the table:

1. Add a new column inside the Data Source Widget to contain the background color of each row
2. Configure the “Row background” color parameter of the Table to point to the color column of the Data Source Widget





Filter:

Description	Set	Min	Max
Room1 Temperature	123	0	200
Room1 Humidity	125	0	200
Room1 Music Level	15	-10	150
Room2 Temperature	-8	-100	100
Room2 Humidity	18	0	100
Room2 Music Level	12	0	150
Room3 Temperature	15	0	150
Room3 Humidity	134	0	500

Table Filter

Content visible inside the table can be filtered using the “Table Filter” property. On datalink you can use a formula (see "Formula" on page 44 chapter for additional details) to define the criteria to use to filter the data.

Each row of the table will be visible only when the Datalink of the Table Filter return true value.

Example 1

If you want choice to see only the rows that contain “something” inside the Description column, you can use the below formula:

```
=$Contains($('Description:TableDataSrcWgt'),$('value:SearchOnTable'))
```

Where

- 'Description:TableDataSrcWgt' is a Dynamic Field of the Data Source Widget used from the table to identify the column to check
- 'value:SearchOnTable' is the value of a text field that will contains the string to search

Filter:

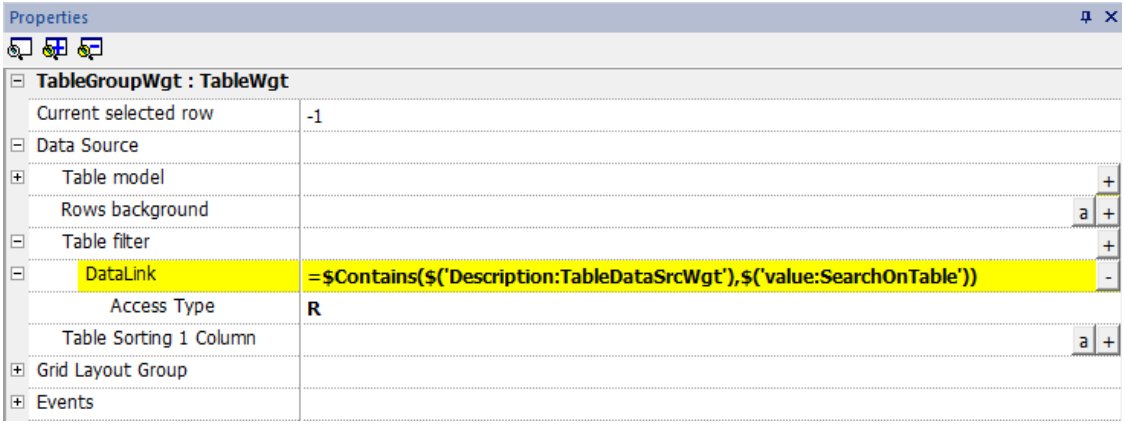
Description	Set	Min	Max
Room1 Temperature	123	0	200
Room1 Humidity	125	0	200
Room1 Music Level	15	-10	150
Room2 Temperature	-8	-100	100
Room2 Humidity	18	0	100
Room2 Music Level	12	0	150
Room3 Temperature	15	0	150
Room3 Humidity	134	0	500

Filter: Room2

Room2 Temperature	-8	-100	100
Room2 Humidity	18	0	100
Room2 Music Level	12	0	150

Filter: Temperature

Room1 Temperature	123	0	200
Room2 Temperature	-8	-100	100
Room3 Temperature	15	0	150
Room4 Temperature	2	0	10



Example 2

To use flags to define the parameters to expose inside the table:

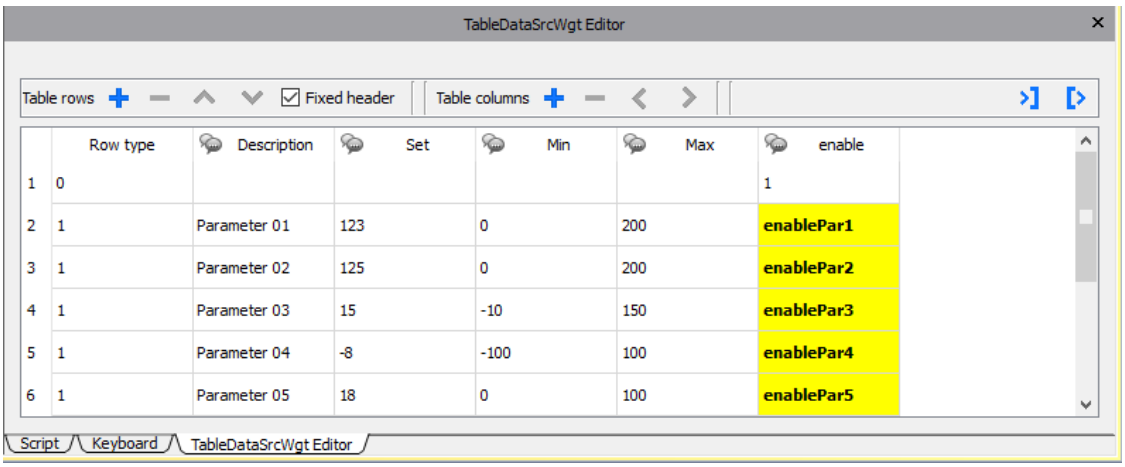
First, add a new column inside the Data Source that will contains the flags that will enable the associate row. Then, link the datalink of the table filter to the new column that contains the flags

enablePar1 = 1
enablePar2 = 1
enablePar3 = 1
enablePar4 = 1
enablePar5 = 1

Description	Set	Min	Max
Parameter 01	123	0	200
Parameter 02	125	0	200
Parameter 03	15	-10	150
Parameter 04	-8	-100	100
Parameter 05	18	0	100
Parameter 06	12	0	150
Parameter 07	15	0	150
Parameter 08	134	0	500

enablePar1 = 1
enablePar2 = 1
enablePar3 = 0
enablePar4 = 0
enablePar5 = 1

Description	Set	Min	Max
Parameter 01	123	0	200
Parameter 02	125	0	200
Parameter 05	18	0	100
Parameter 06	12	0	150
Parameter 07	15	0	150
Parameter 08	134	0	500
Parameter 09	44	0	50
Parameter 10	2	0	10



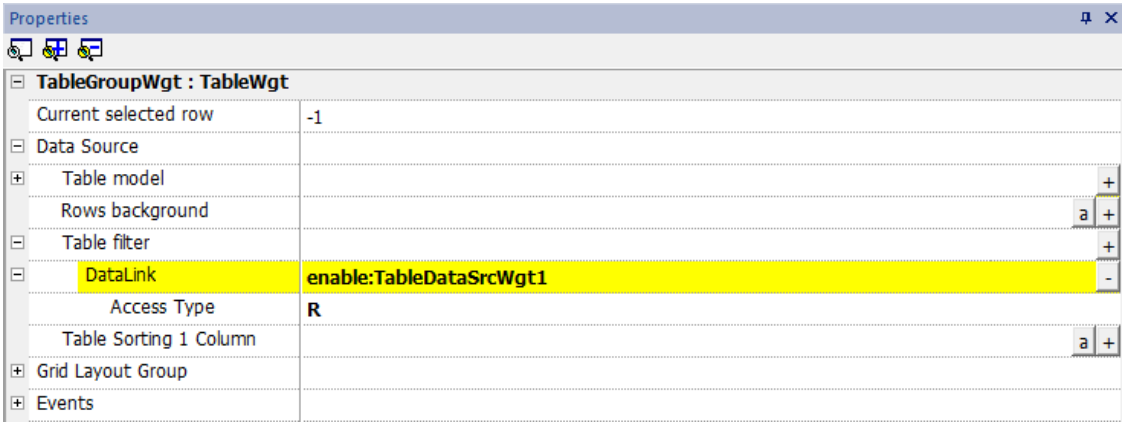
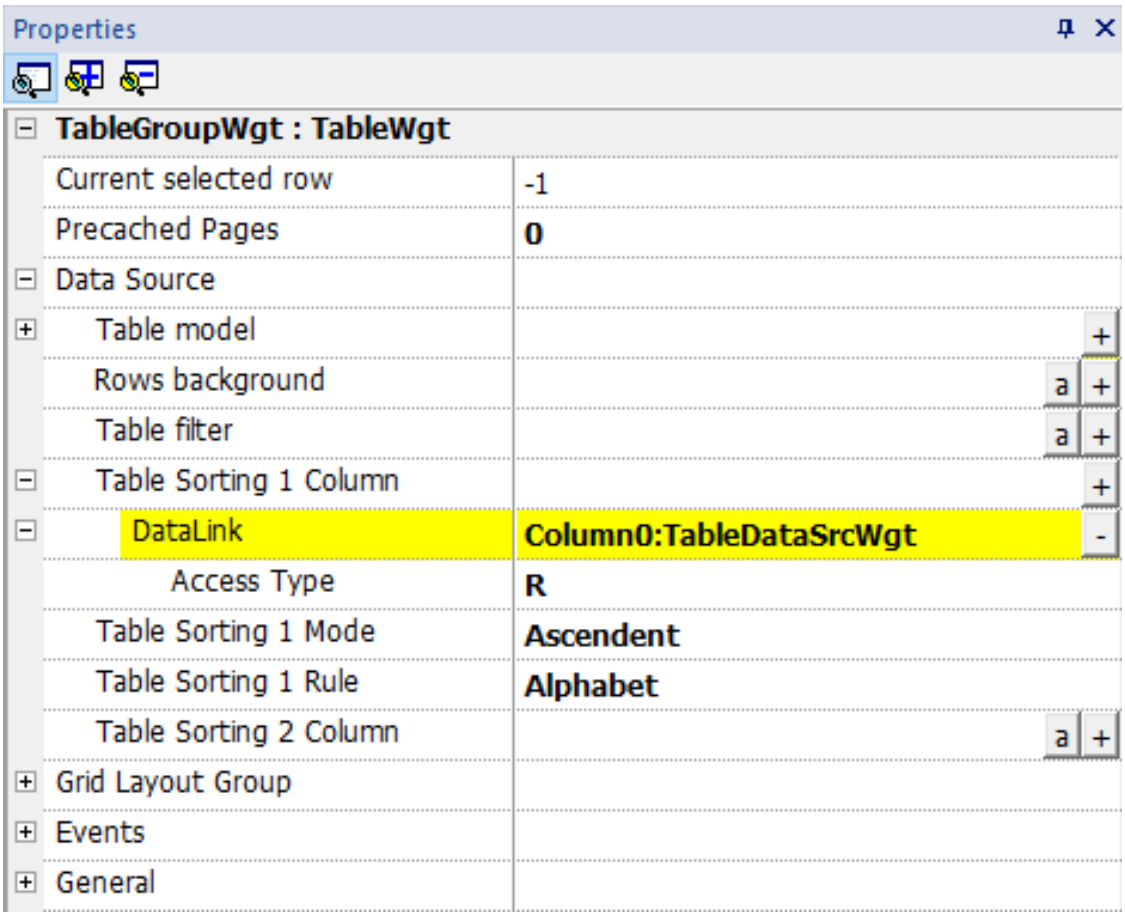


Table Sorting

To sort the rows of the table, select the column of the Data Source that you want to use to sort the table

- Sorting mode can be Ascendent or Descendent
- Sorting Rule can be Alphabetic or Numeric



Multiple sorting (STABLE sorting) is useful when a column has repetitions. You can use up to three sort columns.

Example of sorting:

Alphabetic

Value
1
10
15
7

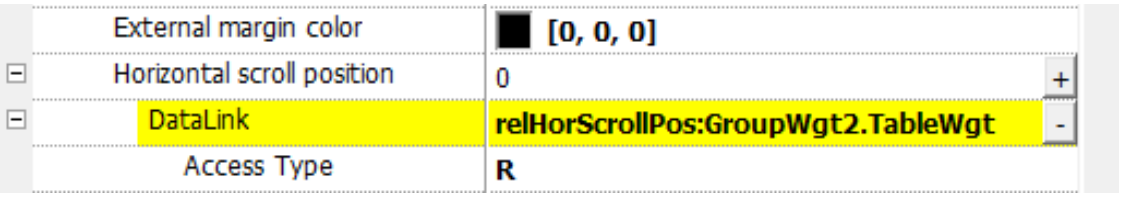
Numeric

Value
1
7
10
15

The table can be ordered even using the SetTableSortingColumn macro (see "SetTableSortingColumn" on page 215 for details).

Horizontal scroll position

The “Horizontal scroll position” give the possibility to keep synchronized the horizontal scroll movements of two tables.



Horizontal scroll position parameter is available only in Advanced Proprieties View mode

Precached Pages

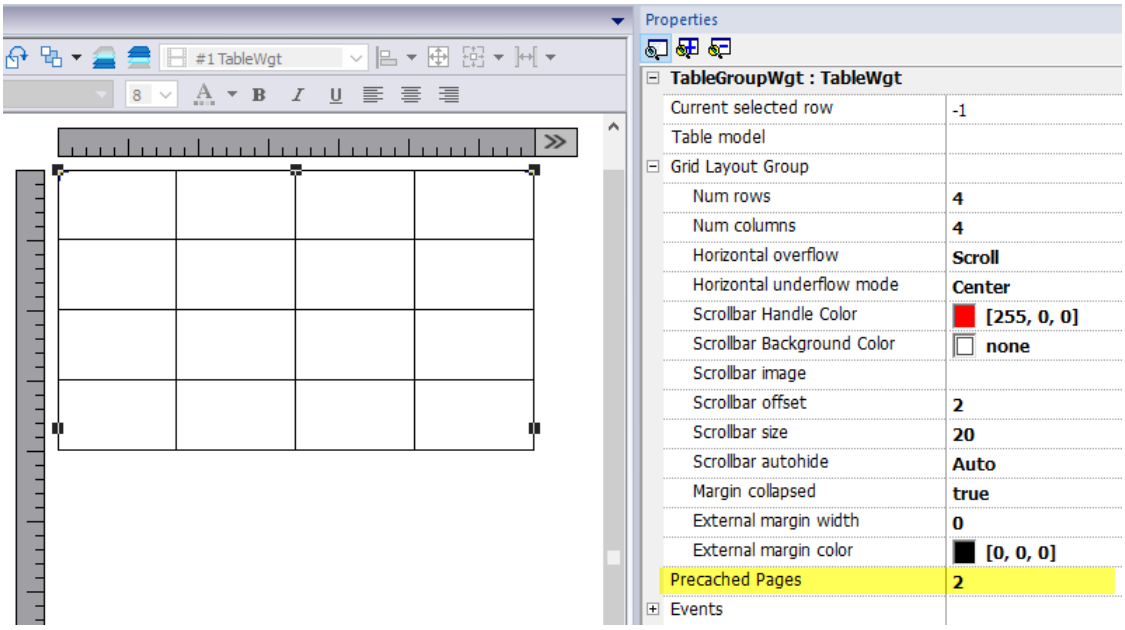
Normally the LRH SW HMI Runtime retrieve only the data that will be visible into the display. To make table scrolling more pleasant, it may be useful to preload the data of the next and previous rows of the displayed ones. Using the Precached Pages parameter is possible define how many pages will be preloaded

- 0 = no pages preloaded
- N = number of pages to preload

Example:

Using a table with 4 rows and Precache Pages = 2

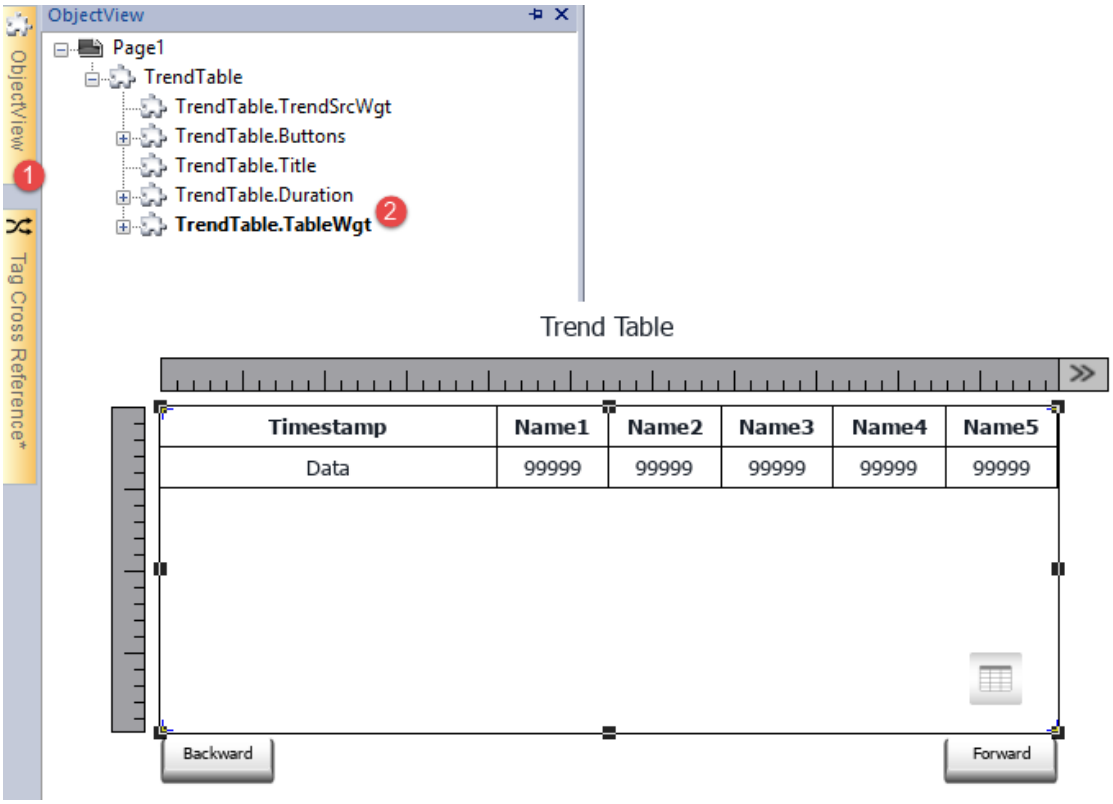
- Number of row to preload are 8 (2 pages x 4 rows)
- 4 rows before (to be ready to manage scroll table up)
- 4 row above (to be ready to manage scroll table down)



Precached Pages parameter is available only in Advanced Proprieties View mode

Widgets that contain tables

Inside the gallery, there are widgets that contain tables, e.g. trend table, audit table, etc. To open the table's properties or the data source's properties you can use the Object View tab and select the component that you need to configure.



Printing table

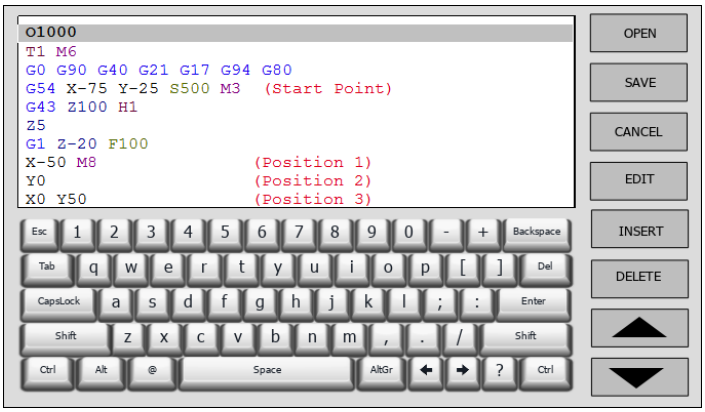
A table widget can be found and used from the print report gallery.

TextEditor widget

Path: ***Widget Gallery> Advanced> Editor***

Use this widget to edit text files. Widget can load the text file from the local HMI device or download the file from a remote device using an ftp connection.

Note: TextEditor widget is available as a plugin (see "Plug-in" on page 79 for details) to download to the LRH SW HMI Runtime only when required.



Widget Buttons

Button	Description
Open	Load text file inside the TextEditor
Save	Save text file
Cancel	Remove all changes from last OPEN or SAVE command
Edit	Enter in edit mode
Insert	Insert a new line
Delete	Delete current line
Up/Down	Move cursor up/down

Widget Properties

Option	Description
Keyboard	TextEditor widget has an embedded keyboard. When widget is used without the embedded keyboard, the alphabetic keyboard will be displayed when enter edit mode. • Hidden • Visible
FTP Config	FTP parameters to download the text file from a remote FTP server. Leave this filed empty to load the text file from the local HMI device.
File Name	File name to edit. When empty a file browser to load a local file will be opened
Syntax Highlight Type	Displays text in different colors according to the selected language • None • GCode

FTP Config

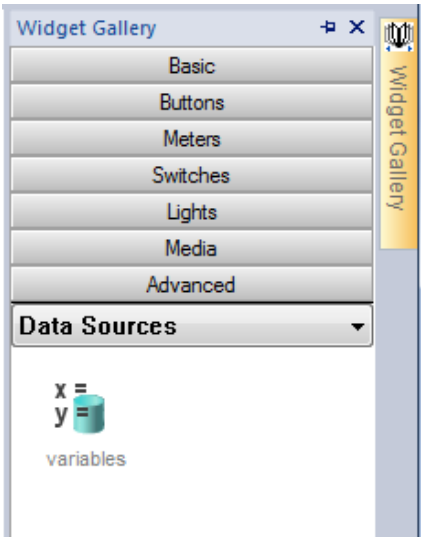
Parameter	Description
FTP Address	FTP server IP Address
Server Port	Port for FTP connection (default = 21).
Authentication	Select the FTP authentication to use: • Normal (Username and password required) • Anonymous
User Name	Username of the remote FTP account
Password	Password of the remote FTP account

Variables widget

Path: ***Widget Gallery> Advanced> Data Sources***

Use this widget to add internal variables for operations such as data transfer or to be used in JavaScript programs.

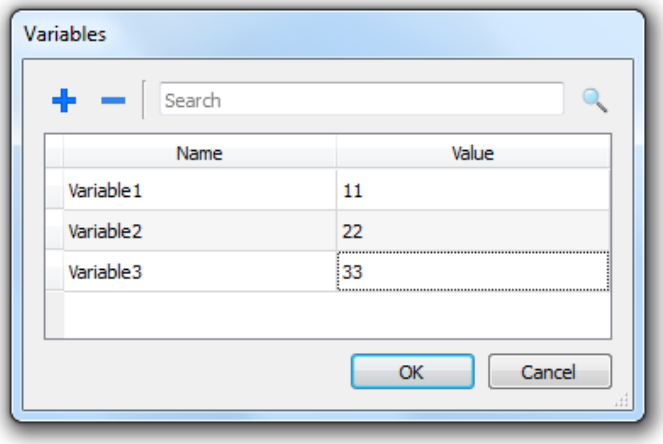
Note: The variables are local to the page where the widget has been inserted.



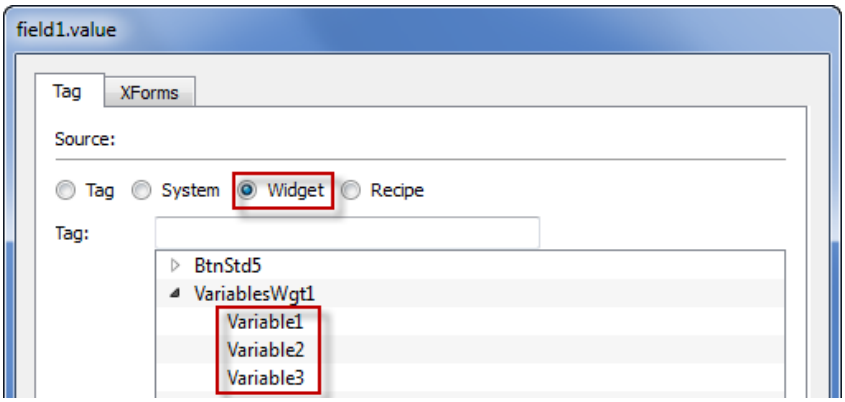
When you drag and drop this widget into you page, a place holder will be displayed to indicate the widget location, but it will not be visible at runtime.

Setting the widget

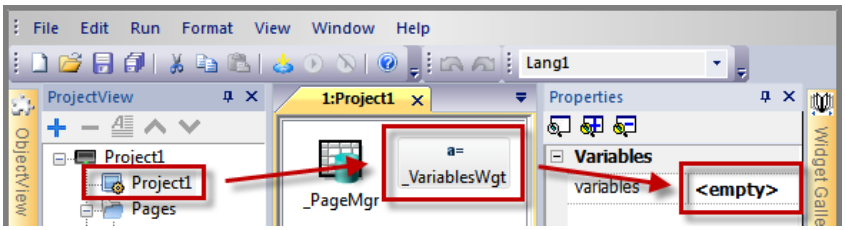
To create variables and assign values to them, open the **Variables** dialog from the **Variables** property in the **Properties** pane.



These variables can then be referenced from the **Attach tag** dialog, from the Page Editor.



If you need global variables, configure them at project level, adding the desired variables to the global variable widget.



Using variables in JavaScript

Variables can be also referenced in JavaScript programs with the following syntax:

For local variables:

```
var varWgt = page.getWidget("_VariablesWgt");
var compVar = varWgt.getProperty("VariableName");
```

For global variables:

```
var varWgt = project.getWidget("_VariablesWgt");
var compVar = varWgt.getProperty("VariableName");
```

38 Custom widgets

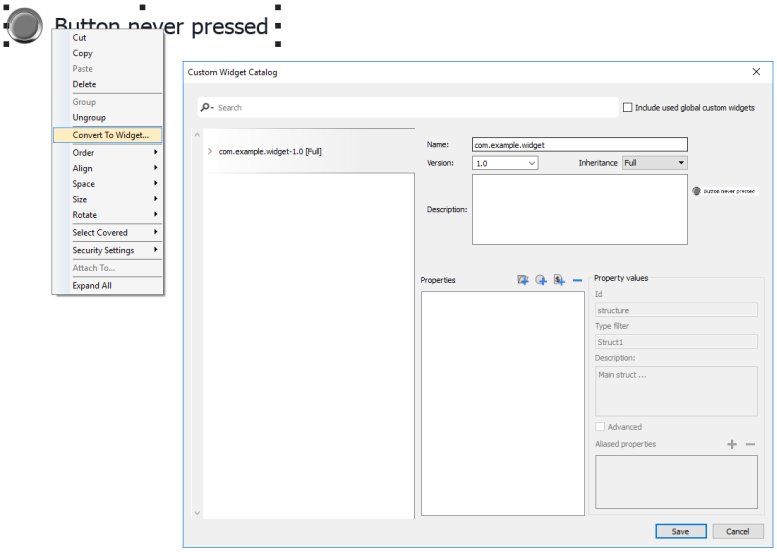
LRH SW has a large widget library which includes predefined dynamic widgets (buttons, lights, gauges, switches, trends, recipes, and dialog items), as well as static images (shapes, pipes, tanks, motors).

You can drag and drop an object from the gallery to the page, and then size, move, rotate or transform it. All widgets in the gallery are vector based, so they do not loose definition when resized.

You can, however, modify any of the pre-defined widgets to create your own custom widget. Custom widgets can be made up of several elements only including the properties needed to their purpose.

Creating a custom widget

1. Drag and drop on a page all the widget you want to use to compose your custom widget.
2. Select and group them.
3. Right-click on the grouped object and select **Convert To Widget**: the **Custom Widgets Catalog** dialog is displayed.

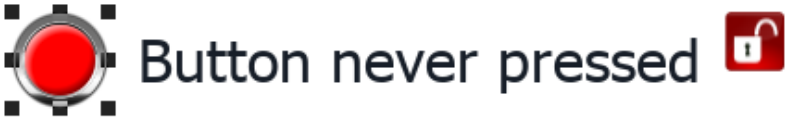


Parameter	Description
Include used custom widgets	When checked, list all the widgets used inside the project. Even system widgets.
Name	You can define everything you prefer, but is common keep a name structure. The folder com.hmi is reserved for the system widgets
Description	Widget description.
Version	Widget version. All widgets that share the same version share the properties defined from the Inheritance parameter.
Inheritance	Properties shared between widgets with the same version • Full (both Graphic and Logic) • Only Graphic • Only Logic • Disable

Modify a custom widget

To modify a custom widget, simple double clicks the custom widget to enter in edit mode.

If the Inheritance flag is enabled, a lock icon will appear to warn you that you are add changes that will be propagated to all the other custom widgets that share the same version. Click the padlock icon to confirm to enter in edit mode, padlock will be open. Click again when modifies are done.

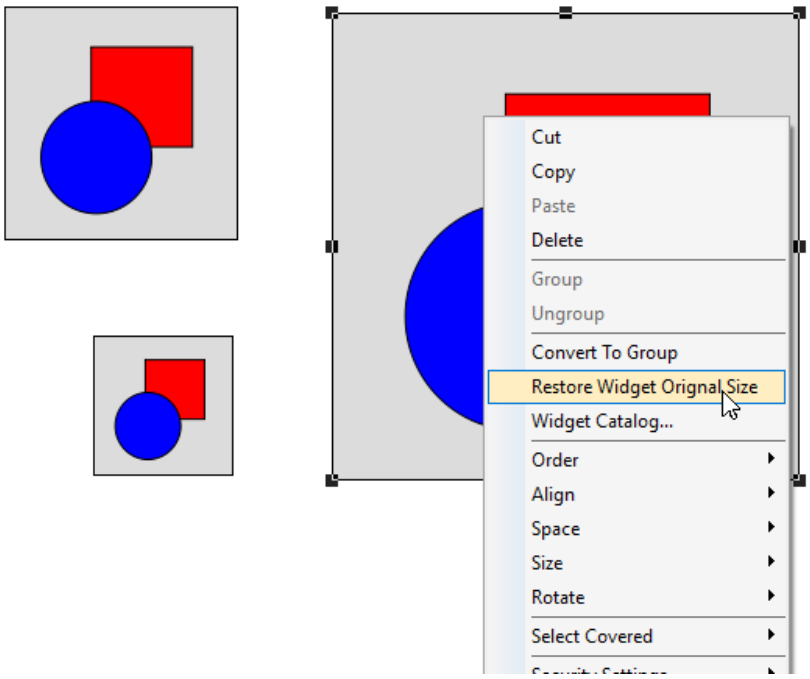


Padlock is showed only when the Inheritance is enabled.

Resize a custom widget

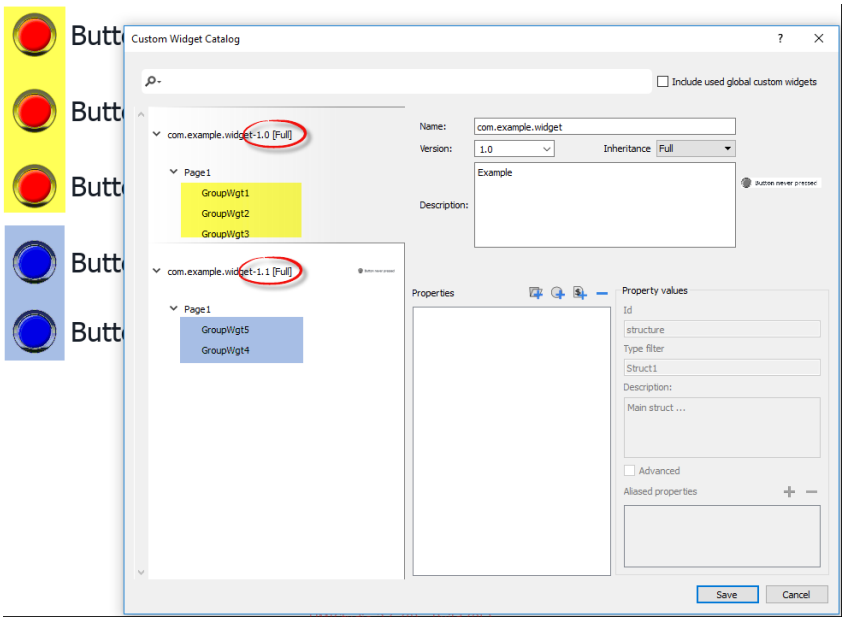
When sizes of custom widget is changed, the new sizes will not be propagated to the other widget instances.

“Restore Widget Original Size” command can be used.



Share properties

When a custom widget is modified, all the modifies will be propagated to all the other custom widgets that share the same version and that are configured to inherit the widget properties.

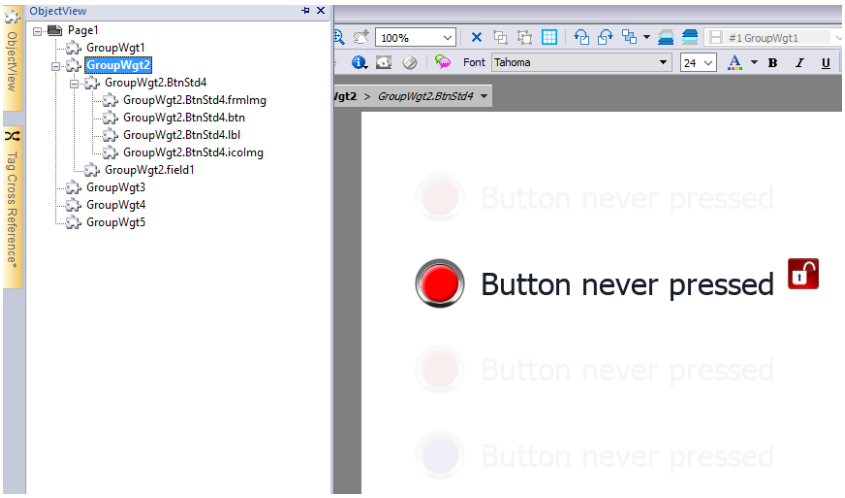


Using widgets components

Widgets are usually made up of many parts, for example a button is a complex widget including two image widgets, a button widget and label.

To display a list of all the elements that are part of a widget, select the widget, open the padlock and open the **ObjectView** pane: all the element making up a complex widget are listed in hierarchical order.

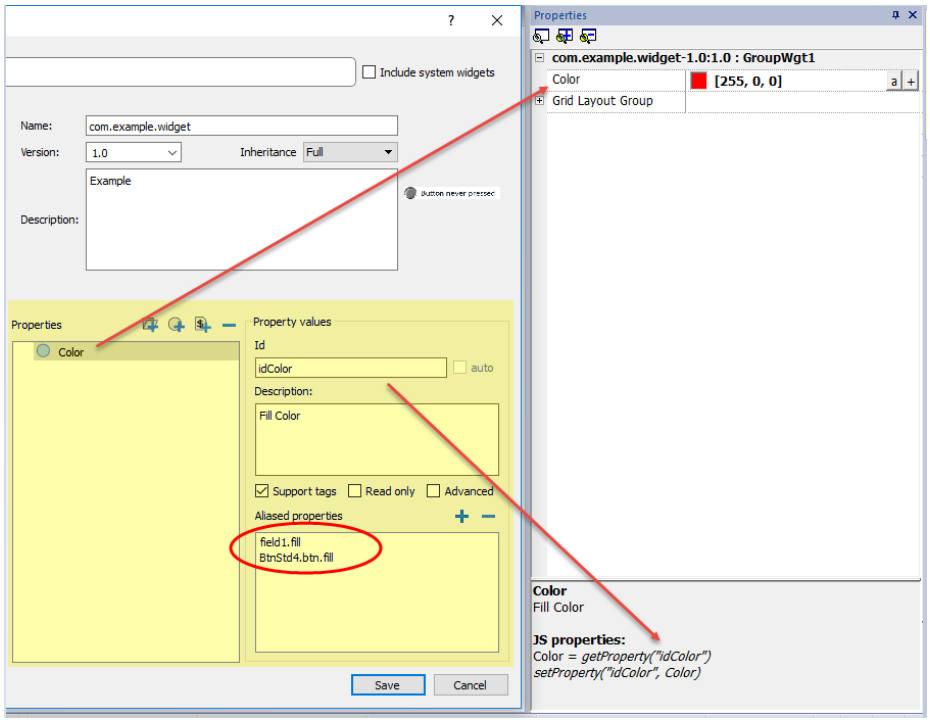
To select a single widget, select it directly form the **ObjectView** pane.



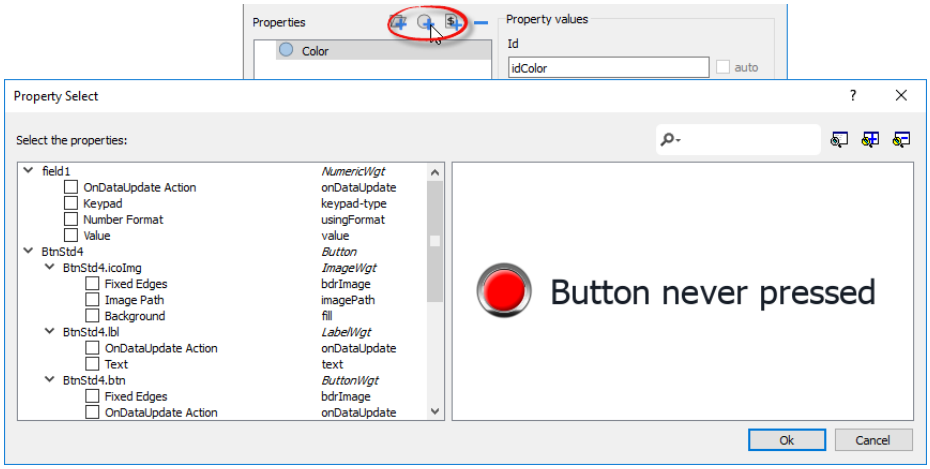
Adding properties to a custom widget

When you create a custom widget, you need to define the properties that will be displayed for it in the **Properties** pane.

1. Right-click on the grouped object and select **Widget catalog**: the properties dialog is displayed.



2. Click **+** to open the **Property Select** dialog: this lists all the properties of all the grouped widgets.



3. Select the properties you want to define for your custom widget.
4. Define each property's details.

Note that you can create folders and use drag & drop to move or reorganize the **Properties** list

Parameter	Description
Properties	Name shown in the **Properties** pane.
Description	Any comment on the property to be displayed in the **Properties** pane.

Parameter	Description
Id	The name exposed by LRH SW, to JavaScript functions and Attach Tag dialog.
Support Tags	Specifies if the property supports the “Attach to" attribute.
Read only	Property exposed only in read mode
Advanced	Specifies whether each property should appear in the advanced, or in the simple view mode of the **Properties** pane.
Aliased properties	Internal properties linked with the exposed property

Combining properties

To combine two or more properties:

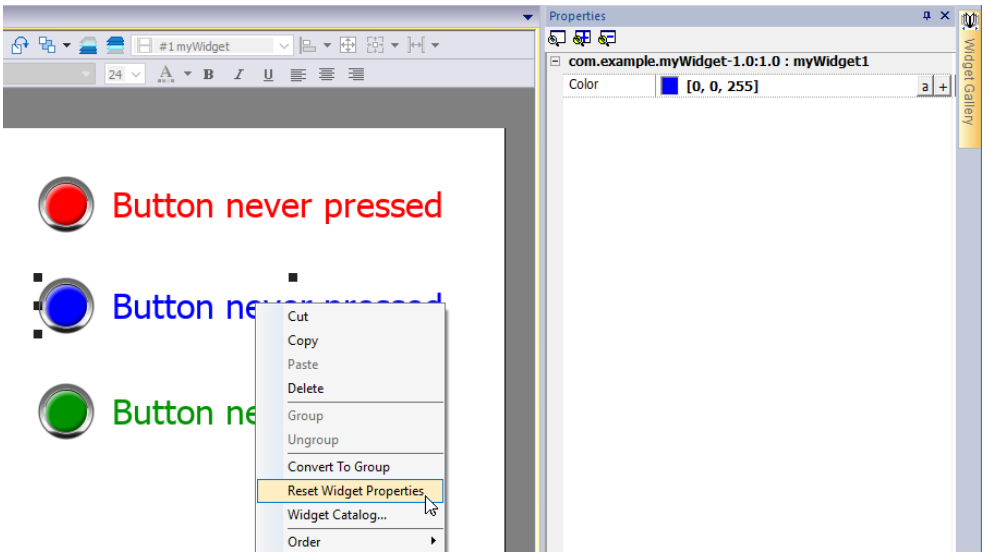
1. Select the primary property in the **Properties** list dialog.
2. Click **+** in the **Aliased properties** toolbar: the **Property Select** dialog is displayed.
3. Select the properties you want to combine.
4. Click **OK:** the combined attributes will be shown in the **Aliased properties** list box.

Example

If you insert into a “Color” property the fill color of all widgets (e.g. filed1.fill and BtnStd4.btn.fill) when you set the exposed Color property of the custom widget all colors of the included widgets will changes.

Reset Widget Properties

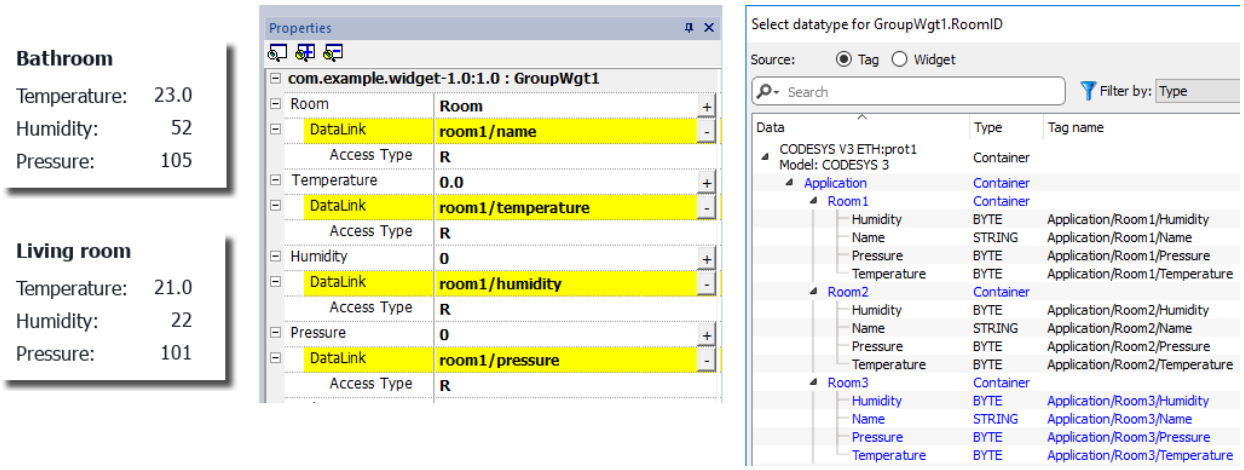
The "Reset Widget Properties" reset the modified properties values to original values.



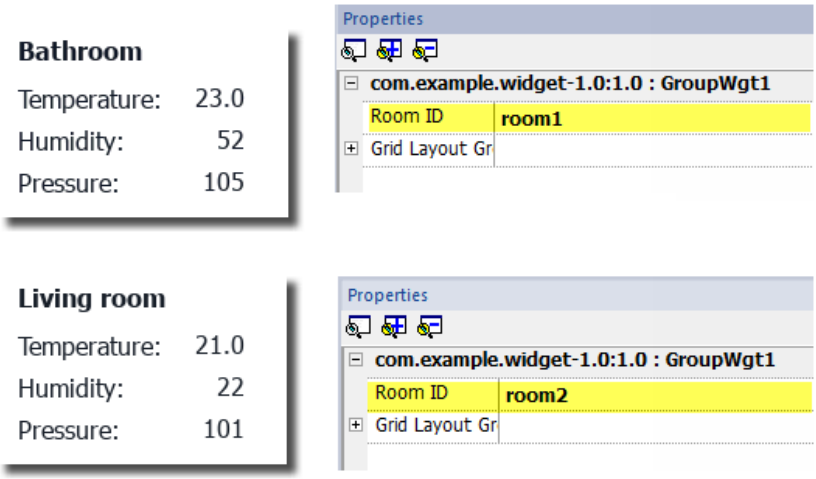
Using structured tags

A common problem using a widget that use many tags is the need to create instances of the widget by giving only the tag name of the structure that contains all the tags instead to configure each single tag.

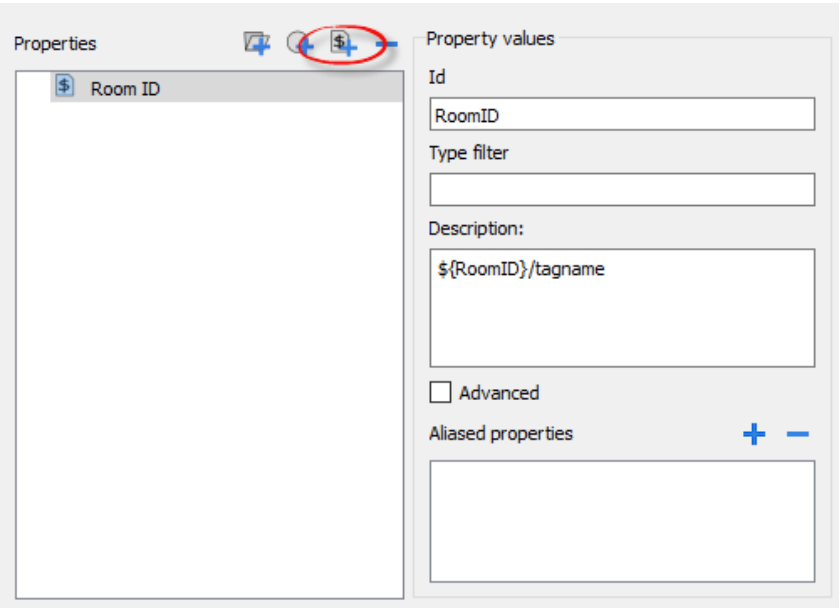
For example, think about the below widget. It use four tags, the room name, temperature, humidity and pressure. If we want use two instances of this widget for two different rooms we have to configure eight tags, four tags for each room.



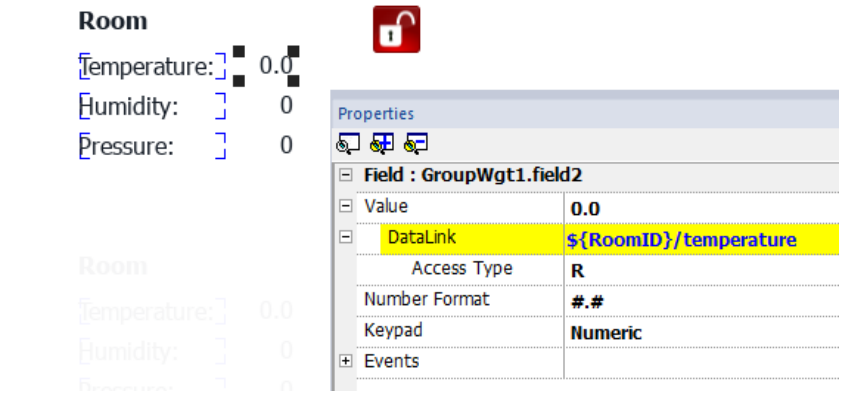
By using a **Parameter** property, is possible to set all the data links of the widget by giving only the name of the structure.



A “Parameter” field can be added inside the custom widget using the *"Add Parameter"* icon:

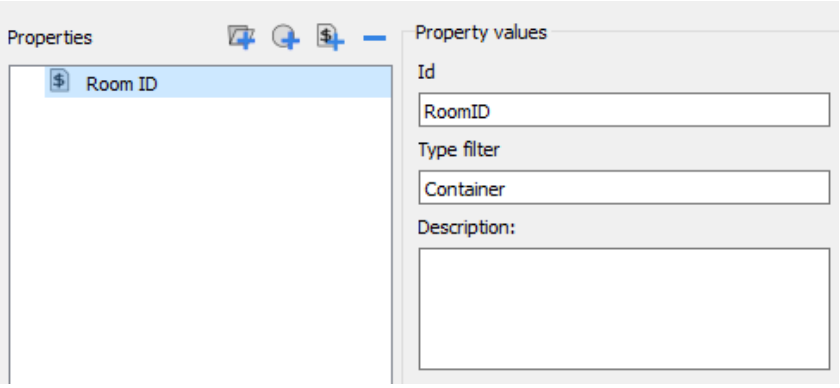


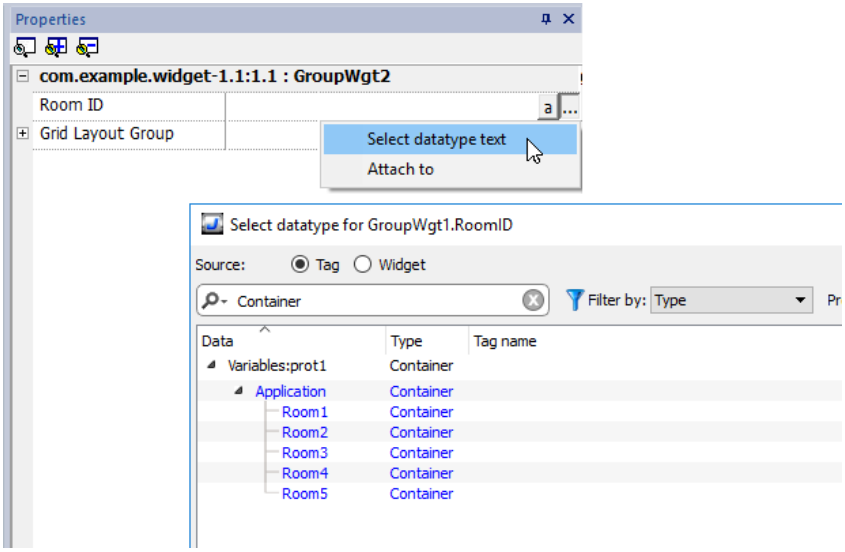
To configure the data links of the custom widget the keyword ${RoomID} can be used to reference at the structure instance



Type filter

Typically, value of the parameter will be an element of a structured tag. Using the *"Type filter"* parameter, the *"Select datatype text"* will list filtered tags.





The *"Select datatype text"* will return a string while the *"Attach to"* will return a datalink to a tag that will contains the string to use.

getParameter

From JavaScript you can read the parameters' value using the getParameter()

```
object getParameter(paramID)
```

Example:

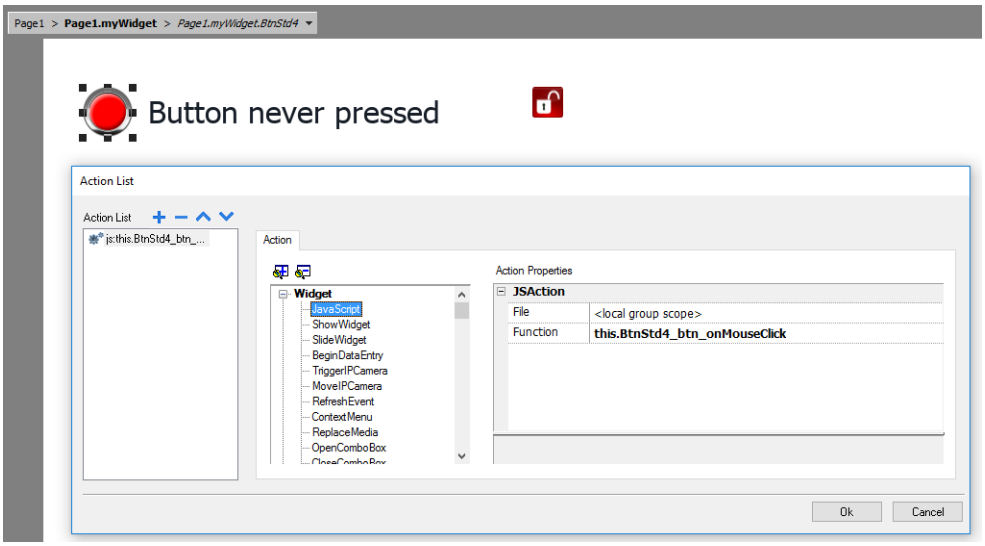
```
var myWidget = page.getWidget("myWidget");
function BtnStd3_btn_onMouseClick(me, eventInfo)
{
     alert("Room is: " + myWidget.getParameter("RoomID"));
}
```

You can also use getProperty(), but getParameter() is more efficient to read custom widget parameters

JavaScript in custom widgets

JavaScript functions can be embedded in custom widgets.

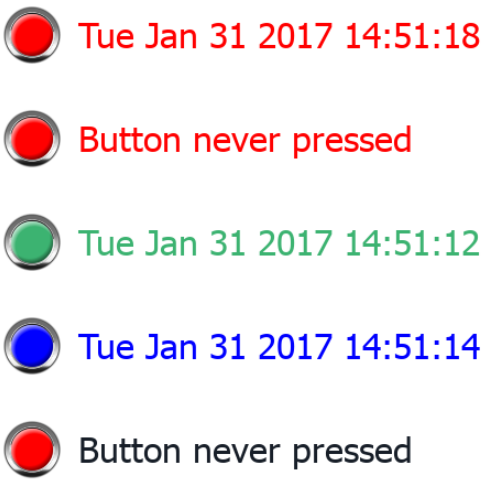
After doing a double click on the custom widget and clicked on the padlock, the edit mode is active and it is possible to associate the JavaScript code to the available events.



Note the usage of the operator **this.** that is necessary to allow the multiple instance of the custom widget.

If you need to reference to an element of the widget, you can use the keyword **wgt.**. For example, use wgt.id to reference at the id of the active widget instance.

If you cut and paste some instances of the custom widget of the above example and execute it, e.g. inside the simulator, you will obtain the below result.



onActivate property

To initialize the custom widget is possible to define the onActive property with an initializing function as for the below example.

The onActivate() function will be execute when the page is loading

Script

```
/*!
javascript module: widget-1.0.js
javascript source file path: lib\com\example\widget-1.0\widget-1.0.js
*/

this.wMSG = wgt.getWidget(wgt.id+".field1")

this.BtnStd1_btn_onMouseClick = function (me, eventInfo)
{
    var now = new Date();
    this.wMSG.setProperty("value", now.toString().slice(0, 24));
}


this.onActivate = function()
{
    this.wMSG.setProperty("value", "Button never pressed");
};
this.onActivate();
```

Note that the custom widget can also past inside the User's Gallery for later reuse.

The JavaScript code used inside the examples of this chapter

```
/*!
javascript module: widget-1.0.js
javascript source file path: lib\com\example\widget-1.0\widget-1.0.js
*/

this.wMSG = wgt.getWidget(wgt.id+".field1")

this.BtnStd1_btn_onMouseClick = function (me, eventInfo)
{
    var now = new Date();
    this.wMSG.setProperty("value", now.toString().slice(0, 24));
}

this.onActivate = function()
{
    this.wMSG.setProperty("value", "Button never pressed");
};
this.onActivate();
```

User's Gallery

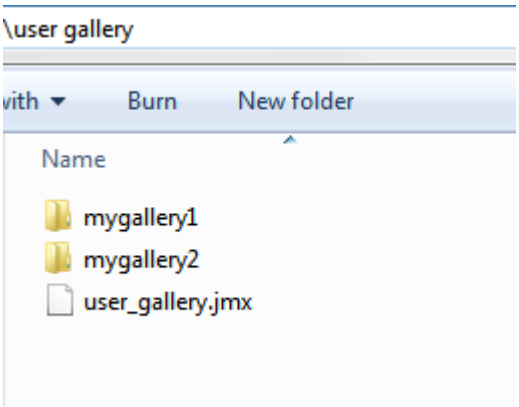
Widgets created from the developers can be saved inside the Widgets Gallery to be available during development of new projects.

User widgets toolbar

Command	Description
	Open the selected widgets folder into the LRH SW editor
	Add a new widgets folder
	Delete current selected folder
	Select the user widgets folder

To add a new widget into the user gallery, open the widget folder and then edit the gallery page creating or adding the new widget.

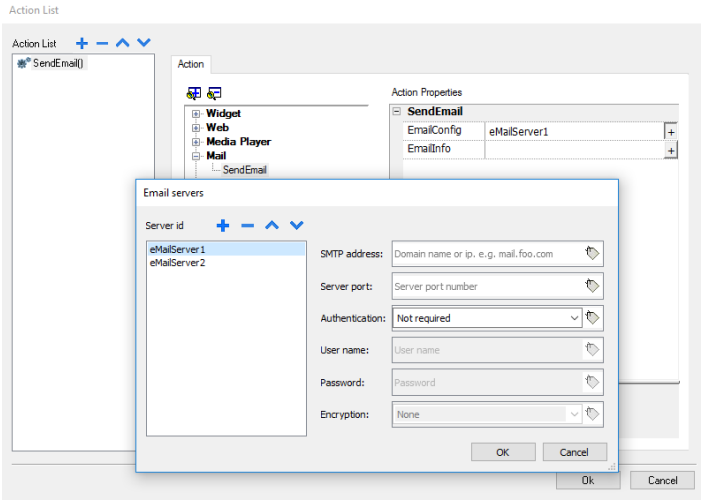
Tip: To import a user gallery sub folder, simply copy the folder to import inside the main user gallery folder.



39 Sending an email message

Send emails using the SendMail action, including tags in the email body and attachments.

The SendMail action has been created for working with alarms and schedulers but can be triggered and executed by many other events.



Configuring the email server

To configure the email server, enter the following information for the **EmailConfig** setting:

Parameter	Description
SMTP Address	SMTP server address.
Server Port	Port for SMTP server connection (default = 25).
Require Auth	Select if the SMTP server requires authentication.
User Name	Username for sending mail using SMTP server.
Password	Password for sending mails using SMTP server.
Encryption	Encryption type (none or SSL).

Click **+** to add more email servers.

Tip: Use tags if you want change the server parameters dynamically from the LRH SW HMI Runtime.

Configure emails

Enter the following information for the **EmailInfo** setting:

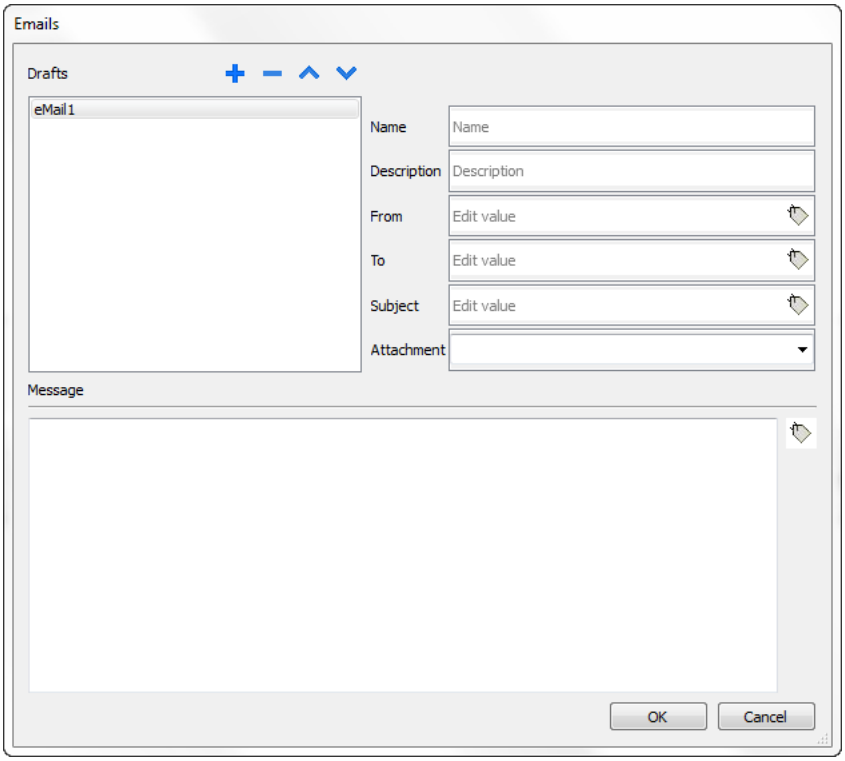
Parameter	Description
Name	Optional, this information is only for the log.
Description	Optional, this information is only for the log.
From	Sender email address (for example, John@domain.com).
To	Recipient e-mail addresses. To enter multiple addresses, separate them with a semi-colon.
Subject	Subject of email.
Attachment	Path of the file to be sent as attachment. Only one attachment at a time can be sent. Note: The maximum size of the attachments is usually set by the SMTP server.
Body	Main content of the email. Here you can insert live tags if you include them in square brackets. For example, a message body as “Tag1 value is [Tag1]”, will be sent as “Tag1 value is 45”, if the current value of Tag1 is 45.

Tip: Attach a string tag to the **From**, **To** and **Subject** fields so that their value can be changed in the LRH SW HMI Runtime.

WARNING: The maximum size for the message body is 4096 bytes, the exceeding text will be truncated.

Adding email templates

Click **+** to add more templates.



40 JavaScript

The purpose of this section is to describe how JavaScript is used in the LRH SW applications, not to explain the JavaScript language.

LRH SW JavaScript is based on the ECMAScript programming language http://www.ecmascript.org , as defined in standard ECMA-262.

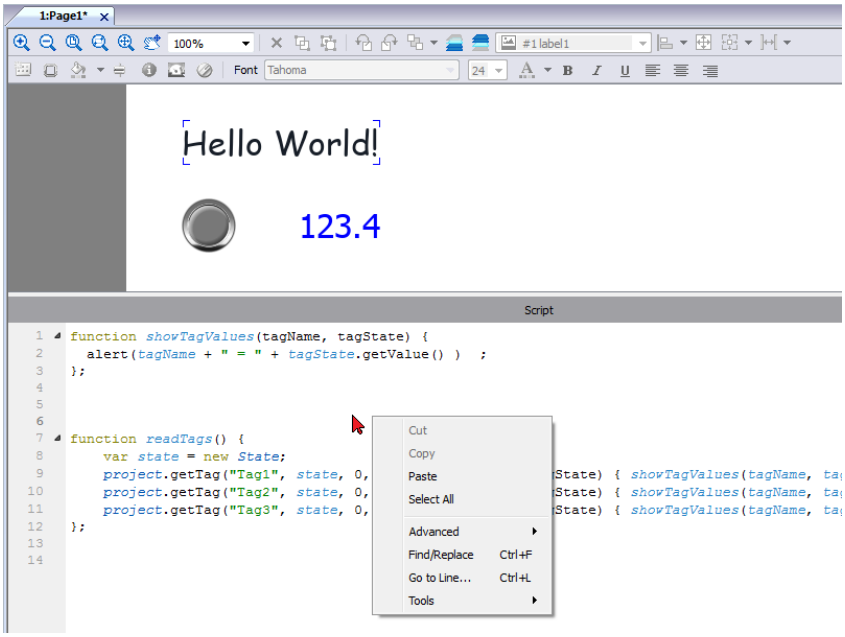
If you are familiar with JavaScript, you can use the same type of commands in LRH SW as you do in a web browser. If you are not familiar with the ECMAScript language, refer to:

https://developer.mozilla.org/en/JavaScript

JavaScript editor

LRH SW includes a powerful JavaScript editor.

Right-click in the editor to display available commands.



Execution of JavaScript functions

JavaScript functions are executed when events occur. For example, a user can define a script for the OnMouseClick event and the JavaScript script will be executed when the button is pressed on the HMI device.

JavaScript functions are executed only when the programmed event occurs and not cyclically. This approach minimizes the overhead required to execute logic in the HMI device.

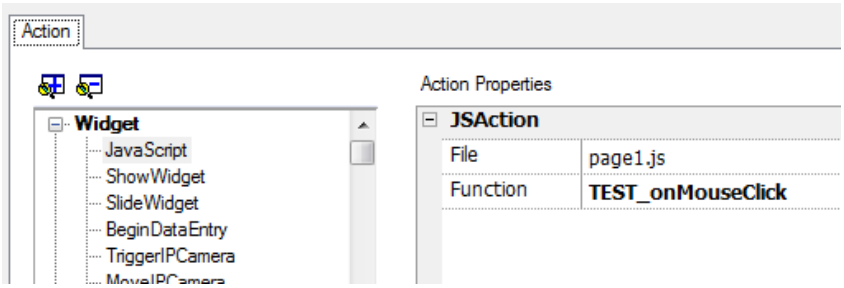
LRH SW provides a JavaScript engine running on the client side. Each project page can contain scripts having a scope local to the page where they are added; global scripts can be created to be executed by scheduler events or alarm events.

In both cases scripts are executed on the client. This means that if more than one client is connected to the HMI device (for external computer running the LRH SW Client), each client will run the same script, providing different output results depending on the input, since inputs provided to different clients may be different.

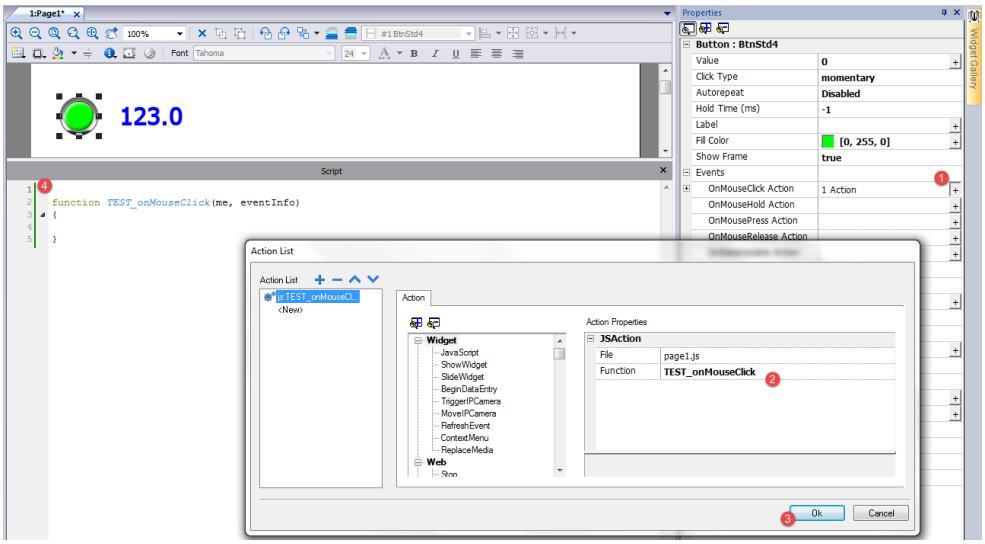
For example, if a script acts according to the position of a slider and this position is different on the different clients, the result of the script will be different on each client.

JavaScript functions for page events

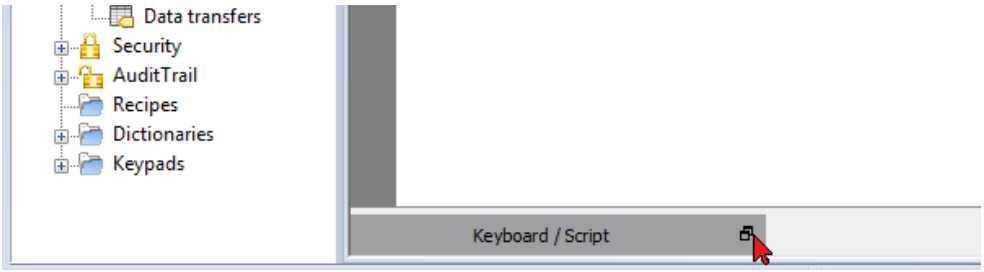
JavaScript editor will open when you add a JavaScript action inside an action list.



1. Select the even that will execute the action.
2. Add a **JavaScript** action from the **Widget** category.
3. Either leave the default function name, or type a new one.
4. Click **OK** to confirm: the JavaScript editor displays your function structure.



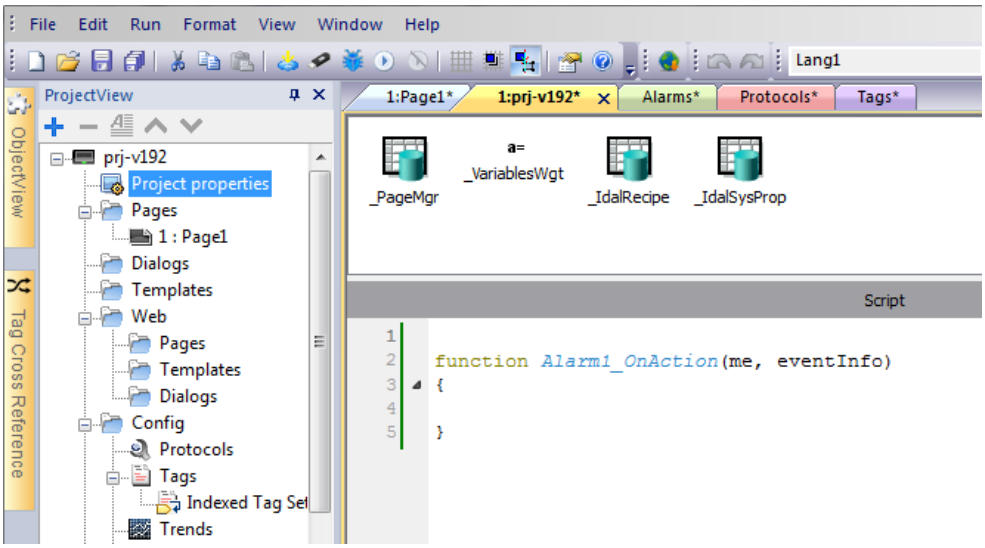
You can also open the JavaScript editor from the **Script** tab at the bottom of the workspace.



JavaScript functions for alarms and scheduled events

JavaScript code associated with alarms and scheduled events and not associated with a specific page, can be edited from the main **Project properties** page.

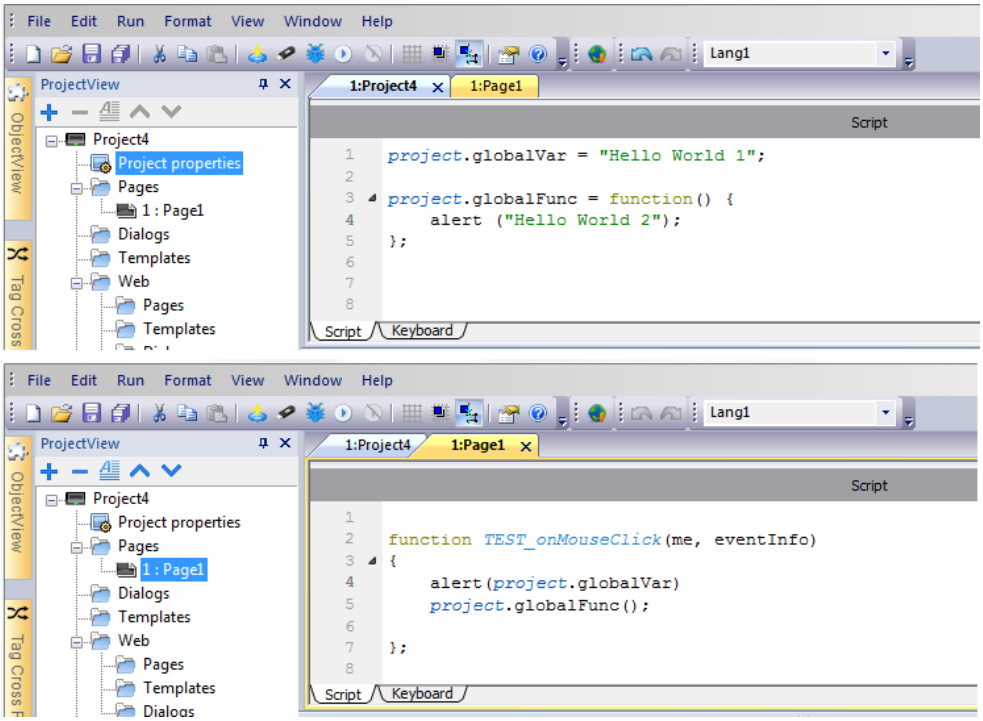
*Path: **ProjectView**> double-click **Project properties***



Note: JavaScript actions are client actions so they are executed only when a client is logged in.

Shared JavaScript code

The **project** global variable can be used to share JavaScript code between the pages. Variables are created/initialized from the main JavaScript code from the main **Project properties** page and can then be used from the project pages.



Events

You can add JavaScript to the following categories of events:

- Widget events
- Page events
- System events

For events of type:

- OnMousePress
- OnMouseRelease
- OnMouseClick
- OnWheel

JavaScript **eventinfo** parameter contains the following additional properties:

Parameter	Description
eventInfo.posX	Local mouse/touch X coordinate with respect to widget coordinates
eventInfo.posY	Local mouse/touch Y coordinate with respect to widget coordinates
eventInfo.pagePosX	Page X mouse/touch coordinate
eventInfo.pagePosY	Page Y mouse/touch coordinate
eventInfo.wheelDelta	Mouse wheel delta. Integer value with sign representing the rotation direction. The actual value is the rotation amount in eighths of a degree. The smallest value depends on the mouse resolution. Typically this is 120, corresponding to 15 degrees.

Widget events

onMouseClick

```
void onMouseClick (me, eventInfo)
```

This event is available only for buttons and it occurs when the button is pressed and released quickly.

Parameter	Description
me	Object triggering the event
eventInfo	Details of triggered event

```
function buttonStd1_onMouseClick(me, eventInfo) {
     //do something…
}
```

onMouseHold

```
void onMouseHold (me, eventInfo)
```

This event is available only for buttons and it occurs when the button is pressed and released after the number of seconds set as **Hold Time** in the widget properties.

Parameter	Description
me	Object triggering the event
eventInfo	Details of triggered event

```
function buttonStd1_onMouseHold(me, eventInfo) {
     //do something…
}
```

onMousePress

```
void onMousePress(me, eventInfo)
```

This event is available only for buttons and it occurs when the button is pressed.

Parameter	Description
me	Object triggering the event
eventInfo	Details of triggered event

```
function buttonStd1_onMousePress(me, eventInfo) {
     //do something…
}
```

onMouseRelease

```
void onMouseRelease (me, eventInfo)
```

This event is available only for buttons and it occurs when the button is released.

Parameter	Description
me	Object triggering the event
eventInfo	Details of triggered event

```
function buttonStd1_onMouseRelease(me, eventInfo) {
     //do something…
}
```

onDataUpdate

```
boolean onDataUpdate (me, eventInfo)
```

This event occurs when data attached to the widget changes.

Parameter	Description
me	Object triggering the event
eventInfo	An object with the fields listed below (you can refer fields using "." - dot notation) **oldValue** = Widget value before the change **newValue** = Value which will be updated to the widget **attrName** = Attribute on which the event is generated **index** = Integer attribute index if any, default = 0 **mode =** W when the user is writing to the widget. R in all others status.

The event is triggered before the value is passed to the widget. A JavaScript code can intercept the event and decide to avoid to update the widget by return true value.

Note: if there are additional macros associate at the event, all macros will be execute regardless of the return value used inside the JavaScript code.

```
function buttonStd1_onDataUpdate(me, eventInfo) {
if ( eventInfo.newValue > 100) {
   //do something...
   return true;  // To avoid to continue and update
                 // the widget (e.g. not update the linked tag)
}
   return false; // To continue and update the widget
                 // (e.g. update the linked tag)
}
```

Page events

onActivate

```
void onActivate( me, eventInfo )
```

This event occurs each time the page is displayed.

Parameter	Description
me	Object triggering the event
eventInfo	Reserved for future use

JavaScript will be executed when the page is active, that is when the page is loaded.

```
function Page1_onActivate(me, eventInfo) {
     //do something…
}
```

onDeactivate

```
void onDeactivate( me, eventInfo )
```

This event occurs when leaving the page.

Parameter	Description
me	Object triggering the event
eventInfo	Reserved for future use

```
function Page1_onDeactivate(me, eventInfo) {
     //do something…
}
```

onWheel

```
void onMouseWheelClock( me, eventInfo )
```

This event occurs when a wheel device is moving (for example, a mouse wheel).

Parameter	Description
me	Object triggering the event
eventInfo	Details of triggered event

```
function Page1_onMouseWheelClock(me, eventInfo) {
     //do something…
}
```

System events

System events can be related to:

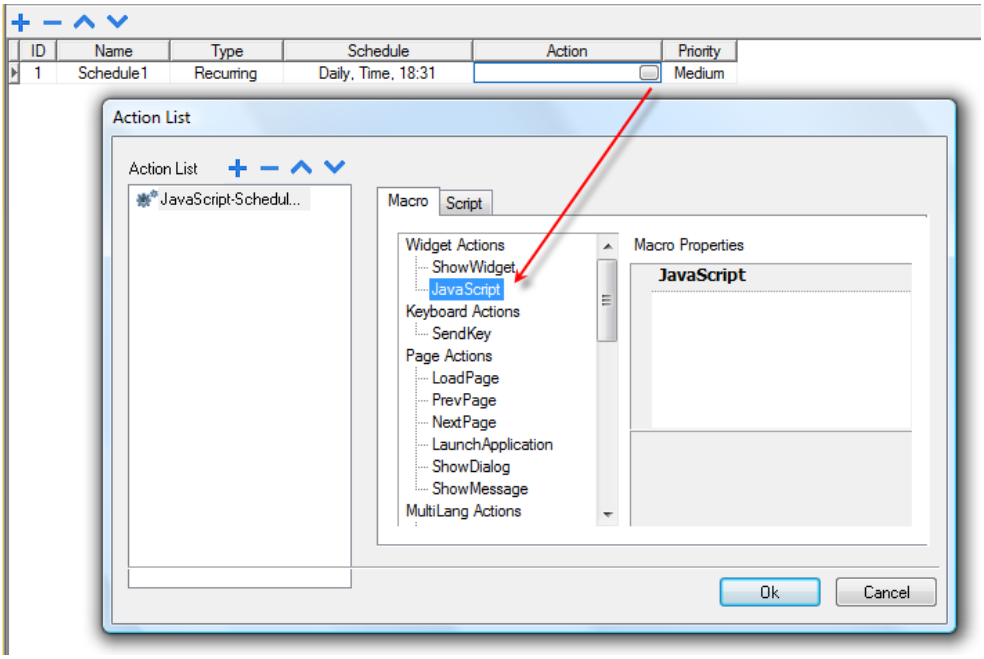
- scheduler
- alarms
- a wheel device

Important: Make sure you do not duplicate JavaScript function names at page and project level. When a conflict happens, that is two functions with the same name in current page and at project level, the system execute the JavaScript callback at page level.

When a JavaScript callback is not found in the current page, the system automatically searches for it at project level.

Scheduler events

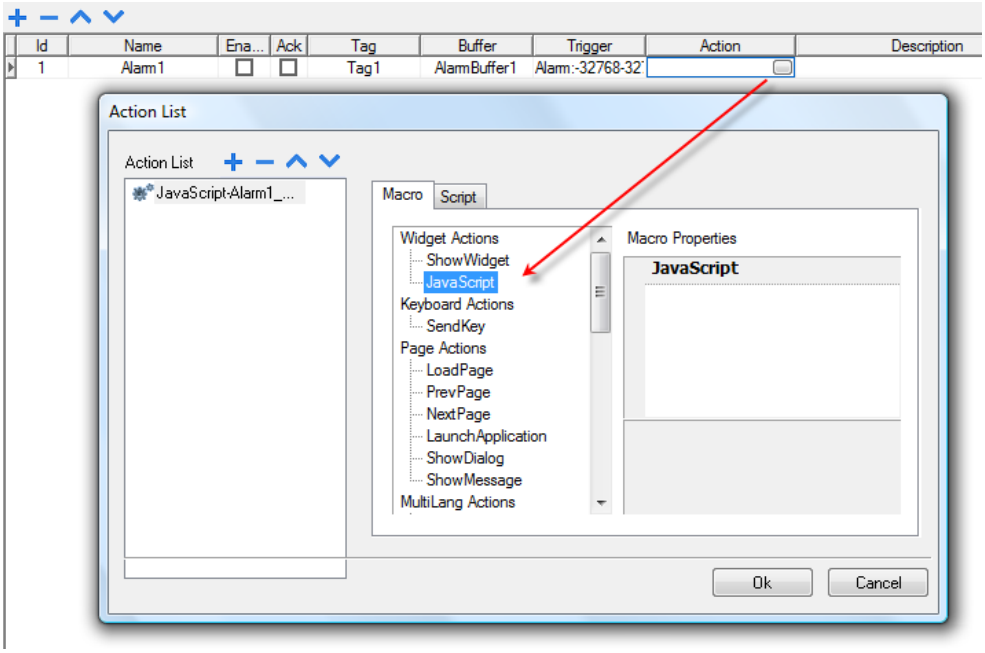
These events occur when triggered by the associated action in the scheduler.



You can edit the JavaScript from the **Project Properties** tab.

Alarm events

These events occur when triggered by the associated alarm condition.



You can edit the JavaScript from the **Project Properties** tab.

onWheel

```
void onMouseWheelClock( me, eventInfo )
```

This event occurs when a wheel device is moving (for example, a mouse wheel).

Parameter	Description
me	Object triggering the event
eventInfo	Details of triggered event

```
function Project1_onMouseWheelClock(me, eventInfo) {
     //do something…
}
```

Objects

LRH SW uses JavaScript objects to access the elements of the page. Each object is composed of properties and methods that are used to define the operation and appearance of the page element. The following objects are used to interact with elements of the HMI device page:

Object	Description
Widget	This is the base class for all elements on the page including the page element
Page	This object references the current HMI device page. The page is the top-level object of the screen.
Group	This object associates a set of tags to allow uniform operation on a set of logically connected tags
Project	This object defines the project widget. The project widget is used to retrieve data about the project such as tags, alarms, recipes, schedules, tags and so on. There is only one widget for the project and it can be referenced through the project variable.
State	This object is the class holding the state of a variable acquired from the controlled environment. Beside the value itself, it contains the timestamp indicating when the value was collected and flags marking the quality of the value.

Widget class objects

The Widget class is the base class for all the elements on a page including the page element.

Widget, in this case, is not used to indicate a specific screen object but a JavaScript class.

Changing widget properties with JavaScript

If you want to change the properties of widgets with JavaScript set the widget property **Static Optimization** to **Dynamic**.

Important: If the widget property Static Optimization is not set to Dynamic, changes to properties will be ignored.

Whenever a call to `getWidget` fails, the remote debugger reports the following error:

“Trying to access static optimized widget "label1". Disable widget static optimization to access widget from script.”.

This error is visible also using following code fragment:

```
var wgt;
try {
wgt = page.getWidget('label1');
} catch(err) {
alert("" + err);
}
```

Widget properties

Some properties are common to all widgets.

objectName

`string objectName`

Gets the name of the widget, a unique id.

```
function btnStd04_onMouseRelease(me) {
     var wgt = page.getWidget("rect1");
     var name = wgt.objectName;
}
```

(Available on web pages)

x

`number x`

Gets or sets the widget ‘x’ position in pixels.

```
function btnStd1_onMouseRelease(me) {
     var wgt = page.getWidget("rect1");
     wgt.x = 10;
}
```

(Available on web pages)

y

`number y`

Gets or sets the widget ‘y’ position in pixels.

```
function btnStd1_onMouseRelease(me) {
     var wgt = page.getWidget("rect1");
     wgt.y = 10;
}
```

(Available on web pages)

width

`number width`

Gets or sets the widget width in pixels.

```
function btnStd1_onMouseRelease(me) {
     var wgt = page.getWidget("rect1");
     wgt.width = 10;
}
```

(Available on web pages)

height

`number height`

Gets or sets the widget height in pixels.

```
function btnStd1_onMouseRelease(me) {
     var wgt = page.getWidget("rect1");
     wgt.height = 10;
}
```

(Available on web pages)

visible

`boolean visible`

Gets or sets the widget visible state.

```
function btnStd4_onMouseRelease(me) {
     var wgt = page.getWidget("rect1");
     wgt.visible = false;
}

function btnStd5_onMouseRelease(me) {
     var wgt = page.getWidget("rect1");
     wgt.visible = true;
```

```
}
```

value

`number value`

Gets or sets the widget value.

```
function btnStd6_onMouseRelease(me) {
     var wgt = page.getWidget("field1");
     wgt.value = 100;
}
```

opacity

`number opacity (range from 0 to 1)`

Gets or sets the widget opacity. Values are decimals from 0 to 1, where 1 is 100% opaque.

```
function btnStd8_onMouseRelease(me) {
     var wgt = page.getWidget("rect1");
     wgt.opacity = 0.5;
}
```

(Available on web pages)

rotation

`number rotation (in degrees)`

Gets or sets the rotation angle for the widget. The rotation is done clockwise and by degrees, starting at the East position.

```
function btnStd9_onMouseRelease(me) {
     var wgt = page.getWidget("rect1");
     wgt.rotation = 45;
}
```

(Available on web pages)

userValue

`string userValue`

Gets or sets a user-defined value for the widget. This field can be used by JavaScript functions to store additional data with the widget.

```
function btnStd9_onMouseRelease(me) {
     var wgt = page.getWidget("rect1");
     wgt.userValue = "Here I can store custom data";
```

```
}
```

Every widget has some specific properties that you can access using dot notation. For an up-to-date and detailed list of properties you can use the JavaScript Debugger inspecting the widget methods and properties.

Widget methods

Some methods are common to all widgets.

getProperty

```
object getProperty( propertyName, [index] )
```

Returns a property.

Parameter	Description
propertyName	String containing the name of property to get
index	Index of the element to get from the array (default = 0)

Almost all properties that are shown in the LRH SW **Properties** pane can be retrieved using the `getProperty` method. The index value is optional and only used for widgets that support arrays.

```
function buttonStd1_onMouseRelease(me, eventInfo) {
     var shape = page.getWidget("rect2");
     var y_position = shape.getProperty("y");
}
```

```
function buttonStd2_onMouseRelease(me, eventInfo) {
     var image = page.getWidget("multistate1");
     var image3 = image.getProperty("imageList", 2);
     //…
}
```

(Available on web pages)

setProperty

```
boolean setProperty( propertyName, value, [index] )
```

Sets a property for the widget.

Parameters

Parameter	Description
propertyName	String containing the name of property to set
value	String containing the value to set the property.
index	Index of the element to set in the array (default = 0)

Almost all properties that are shown in the LRH SW **Properties** pane can be set by this method. The index value is optional and only used for Widgets that support arrays (for example, a MultiState Image widget). The `setProperty` method returns a boolean value (true or false) to indicate if the property was set or not.

```
function buttonStd1_onMouseRelease(me, eventInfo) {
     var setting_result = shape.setProperty("y", 128);
     if (setting_result)
     alert("Shape returned to start position");
}
```

```
function buttonStd2_onMouseRelease(me, eventInfo) {
     var image = page.getWidget("multistate1");
     var result = image.setProperty("imageList", "Fract004.png", 2);
     //…
}
```

(Available on web pages)

Page object

This object references the current HMI device page. The page is the top-level object of the screen.

Page object properties

Properties available at page level.

backgroundColor

`string backgroundColor (in format rgb(xxx, xxx, xxx) where xxx range from 0 to 255)`

Page background color.

```
function btnStd11_onMouseRelease(me) {
     page.backgroundColor = "rgb(128,0,0)";
}
```

(Available on web pages)

width

```
number width
```

Page width in pixels.

```
function btnStd05_onMouseRelease(me) {
     var middle_x = page.width / 2;
}
```

(Available on web pages, get only)

height

```
number height
```

Page height in pixels.

```
function btnStd05_onMouseRelease(me) {
     var middle_y = page.height / 2;
}
```

(Available on web pages, get only)

userValue

```
string userValue
```

Gets or sets a user-defined value for the widget. This field can be used by JavaScript functions to store additional data with the page.

```
function btnStd9_onMouseRelease(me) {
     page.userValue = "Here I can store custom data";
}
```

(Available on web pages)

Page object methods

Methods that can be used at page level.

getWidget

```
object getWidget( wgtName )
```

Returns the widget with the given name.

Parameter	Description
wgtName	String containing the widget name

Return value

An object representing the widget. If the widget does not exist, null is returned.

```
function btnStd1_onMouseRelease(me) {
     var my_button = page.getWidget("btnStd1");
}
```

(Available on web pages)

setTimeout

```
number setTimeout( functionName, delay )
```

Starts a timer to call a given function after a given delay.

Parameter	Description
functionName	String containing the name of function to call
delay	Delay in milliseconds

Return value

A number corresponding to the timerID.

```
var duration = 3000;
var myTimer = page.setTimeout("innerChangeWidth()", duration);
```

(Available on web pages)

clearTimeout

```
void clearTimeout( timerID )
```

Stops and clears the timeout timer with the given timer.

Parameter	Description
timerID	Timer to be cleared and stopped

```
var duration = 3000;
var myTimer = page.setTimeout("innerChangeWidth()", duration);
// do something
page.clearTimeout(myTimer);
```

(Available on web pages)

setInterval

```
number setInterval( functionName, interval )
```

Starts a timer that executes the given function with the given interval.

Parameter	Description
functionName	String containing the name of function to call
interval	Interval in milliseconds

Return value

A number corresponding to the timerID.

```
var interval = 3000;
var myTimer = page.setInterval("innerChangeWidth()", interval);
```

(Available on web pages)

clearInterval

```
void clearInterval( timerID )
```

Stops and clears the interval timer with the given timer.

Parameter	Description
timerID	Timer to be cleared and stopped

```
var interval = 3000;
var myTimer = page.setInterval("innerChangeWidth()", interval);
// do something
page.clearInterval(myTimer);
```

(Available on web pages)

clearAllTimeouts

```
void clearAllTimeouts()
```

Clears all the timers started.

```
page.clearAllTimeouts();
```

(Available on web pages)

Project object

This object defines the project widget. The project widget is used to retrieve data about the project such as tags, alarms, recipes, schedules, tags and so on. There is only one widget for the project and it can be referenced through the project variable.

Project object properties

Properties to be set at project level.

startPage

```
string startPage
```

Page shown when the project is started.

```
var startPage = project.startPage;
project.startPage = "Page2.jmx";
```

Project object methods

Methods to be used at project level.

nextPage

```
void nextPage()
```

The script executes the Next page action.

```
project.nextPage();
```

(Available on web pages)

prevPage

```
void prevPage()
```

The script executes the previous page action.

```
project.prevPage();
```

(Available on web pages)

lastVisitedPage

```
void lastVisitedPage()
```

The script executes the last visited page action.

```
project.lastVisitedPage();
```

(Available on web pages)

homepage

```
void homePage()
```

The script executes the Home page action.

```
project.homePage();
```

(Available on web pages)

loadPage

```
void loadPage(pageName)
```

The script executes to load the set page defined in the script.

```
project.loadPage("Page5.jmx");
```

(Available on web pages)

WARNING: When page change, all active time events are forced to removed and the JavaScript procedure will run until the end before switch to the new page.

showDialog

```
void showDialog(pageName)
```

The script executes to show the dialog page.

```
project.showDialog("Dialog.jmx");
```

(Available on web pages)

closeDialog

```
void closeDialog()
```

The script executes to close the currently-opened dialog page.

```
project.closeDialog();
```

(Available on web pages)

showMessage

```
void showMessage( message )
```

The script executes to display the message popup.

```
project.showMessage("Hi This is test message");
```

(Available on web pages)

getGroup

```
number getGroup( groupName, groupInstance, [callback] )
```

Fast read method; this gets the values of all tags in a group.

Parameter	Description
groupName	String containing the names of the groups. The and/or expression to retrieve tags list from multiple group is supported. | OR operator & AND operator (...) The brackets can be used to define how evaluate the expression Examples: • project.getGroup("one", group); • project.getGroup("(one|two)", group); • project.getGroup("((one&two)*three)", group);
groupInstance	Group element to be filled
callback	String containing the name of the function to be called when the group is ready

Return value

A number value that is the status: 1 for success, 0 for fail.

```
var group = new Group();
var status = project.getGroup ("enginesettings", group);
if (status == 1) {
     var value = group.getTag("Tag1");
     if (value!=undefined) {
     // do something with the value
     }
}
```

```
var g = new Group();
var status = project.getGroup ("enginesettings", g,
     function (groupName, group) { fnGroupReady(groupName, group);} );

function fnGroupReady(groupName, group) {
     var val = group.getTag("Tag1");
     if (val!=undefined) {
     // do something with the value
     }
}
```

(Available on web pages)

getTag

```
object getTag( tagName, state, index, forceRefresh)
```

```
void getTag( tagName, state, index, callback, forceRefresh)
```

It returns the tag value or the complete array if index value is -1 of the given tagName.

Parameter	Description
tagName	String of tag name
state	State element to be filled
index	Index if the tag is of array type. -1 returns the complete array. Default = 0.
callback	Function name if an asynchronous read is required. Default = “”.
forceRefresh	(Optional parameter) True = the Runtime will read an updated value of the tag directly from the device. Default is false.

Return value

Tags value is returned. If tag is array type and index = -1 then the complete array is returned. For non-array tags provide index as 0.

```
var state = new State();
var value = project.getTag("Tag1", state, 0);
//
//for non array type
//tags index is not considered, so can be left as 0
//
if (value!=undefined) {
//...do something with s
}
```

```
var state = new State();
project.getTag("Tag1", state, -1,
     function(tagName, tagState) { fnTagReady(tagName, tagState); });
function fnTagReady(tagName, tagState) {
     if (tagName=="Tag1") {
     var myValue = tagState.getValue();
     }
}
```

(Available on web pages)

setTag

```
number setTag( tagName, tagValue, [index], [forceWrite] )
```

Sets the given tag in the project. Name and value are in strings.

Parameter	Description
tagName	String of tag name
tagValue	Object containing the value to write
index	Index if the tag is of array type. -1 pass the complete array. Default = 0.
forceWrite	Boolean value for enabling force write of tags, the function will wait for the value to be written before it returns back. Default = false.

Return value

Interger value for denoting success and failure of action when forceWrite is true. 0 means success and -1 means failure. If forceWrite is false, returned value will be undefined.

```
var val = [1,2,3,4,5];
var status = project.setTag("Tag1", val, -1, true);
if (status == 0) {
     // Success
} else {
     // Failure
}
```

```
var val = "value";
project.setTag("Tag1", val);
```

(Available on web pages)

updateSystemVariables

`void project.updateSystemVariables()`

Force system variables to refresh.

```
project.updateSystemVariables()
```

selectAllAlarms

void project.selectAllAlarms(bool selected)

Select/unselect all alarms

```
project.selectAllAlarms(true)
```

(Available on web pages)

ackAlarms

void project.ackAlarms()

Acknowledge all selected alarms

```
project.selectAllAlarms(true);
project.ackAlarms();
project.selectAllAlarms(false);
```

(Available on web pages)

resetAlarms

void project.resetAlarms()

Reset all selected alarms

```
project.selectAllAlarms(true);
project.resetAlarms();
project.selectAllAlarms(false);
```

(Available on web pages)

enableAlarms

void project.enableAlarms()

Enable all selected alarms

```
project.selectAllAlarms(true);
project.enableAlarms();
project.selectAllAlarms(false);
```

(Available on web pages)

getRecipeItem

```
object getRecipeItem (recipeName, recipeSet, recipeElement)
```

Gets the value of the given recipe set element.

Parameter	Description
recipeName	String representing the recipe name
recipeSet	String representing the recipe set, can be either the recipe set name or 0 based set index.
recipeElement	String representing the recipe Element, can be either the element name or 0 based element index.

Return value

An object with the value of the recipe. undefined is returned if invalid. If of type array, an array object type is returned.

```
var value = project.getRecipeItem("recipeName", "Set", "Element");
```

setRecipeItem

```
number setRecipeItem (recipeName, recipeSet, recipeElement, value )
```

Gets the value of the given recipe set element.

Parameter	Description
recipeName	String representing the recipe name
recipeSet	String representing the recipe set, can be either the recipe set name or 0 based set index.
recipeElement	String representing the recipe Element, can be either the element name or 0 based element index.
value	An object containing the value to store in the recipe. It can be an array type.

Return value

Interger value for denoting success and failure of action. A '0' means success and '-1' means failure.

```
var val = [2,3,4];
project.setRecipeItem("recipeName", "Set", "Element", val);
if (status == 0) {
     // Success
} else {
     // Failure
}
```

downloadRecipe

```
void downloadRecipe (recipeName, recipeSet )
```

Downloads the recipe set to the corresponding tag.

Parameter	Description
recipeName	String representing the recipe name
recipeSet	String representing the recipe set, can be either the recipe set name or 0 based set index.

```
project.downloadRecipe("recipeName", "Set");
```

uploadRecipe

```
void uploadRecipe (recipeName, recipeSet )
```

Uploads the value of tags into the provided recipe set.

Parameter	Description
recipeName	String representing the recipe name
recipeSet	String representing the recipe set, can be either the recipe set name or 0 based set index.

```
project.uploadRecipe("recipeName", "Set");
```

launchApp

```
void launchApp( appName, appPath, arguments, singleInstance)
```

Executes an external application.

Parameter	Description
appName	String containing the application name
appPath	String containing the application absolute path
Arguments	String containing the arguments to be sent to application
singleInstance	true = only single instance allowed, false = multiple instances allowed

Note the pathname's syntax depend from the native OS format (see "HMI devices capabilities" on page 541).

On **Linux devices**, the pathname's syntax need slash character (even double slash character is permitted).

```
project.launchApp
("pdfViewer","/mnt/data/hmi/qthmi/deploy","/mnt/usbmemory/test.pdf","true");
```

getClientType

```
string getClientType()
```

Return the client type

Client Type	Description
local	Running on HMI device
remote	Running on LRH SW Client client
web	Running on Web client

```
var clientType = project.getClientType();
if (clientType=="web") {
       // Currently running on web client
} else  if (clientType=="remote") {
       // Currently running on LRH SW Client
} else  if (clientType=="local")  {
       // Currently running on HMI Device
}
```

(Available on web pages)

login

```
int project.login("username", "password")
```

Access to the system with the given credentials

```
var ReplyCode;
ReplyCode = project.login("admin", "admin");
if (ReplyCode != 0) {
    alert("Access denied");
}
```

Return value	
0	No Error
1	Error: You are not authorized.
2	Error: Connection lost with the Runtime.
3	Error: The username or password you entered is incorrect
4	Error: The password entered is incorrect
5	Error: Action cannot be executed
6	Error: Passwords do not match
7	Error: Password length too short
8	Error: Password must contain numbers
9	Error: Password must contain special characters
10	Error: Password must be different than previous passwords
11	Error : User already exist
12	Error: Password cannot be empty
13	Error: Your password has expired
14	Warning: Your password will expire soon

logout

`project.logout(AllowDefaultUser)`

Exiting the system

```
project.logout();       // Logout even from default user
project.logout(true);   // Logout even from default user
project.logout(false);  // Logout only if not logged as default user
```

Project object widgets

getCurrentPageName

```
string getCurrentPageName()
```

Return the name of current active page

```
// Get PageMgr widget
var pageMgr = project.getWidget( "_PageMgr" );

// Show Current Page
var currentPageName = pageMgr.getCurrentPageName();
project.showMessage( "Current active page is: " + currentPageName );
```

(Available on web pages)

hasPage

```
boolean hasPage(string pageName)
```

Return true if the page exist, false otherwise

```
// Get PageMgr widget
var pageMgr = project.getWidget( "_PageMgr" );

//Page exists
var pageExists = pageMgr.hasPage( "Page10" );
if (pageExists) {
    project.showMessage( "Page10 exists" );
} else {
    project.showMessage( "Hei Page10 not exists!" );
}
```

(Available on web pages)

curLangCode

```
string curLangCode
```

Property of MultiLangMgr widget. Contains the code of the active language.

```
// Get MultiLangMgr widget
var MultiLangMgr = project.getWidget( "_MultiLangMgr" );

// Show curLangCode
var curLangCode = MultiLangMgr.curLangCode;
project.showMessage( "Current active language is: " + curLangCode );
```

Print reports object

printGfxReport

```
void printGfxReport( reportName, silentMode)
```

Prints the graphic report specified by reportName.

Parameter	Description
reportName	String containing the report name
silentMode	True = silent mode enabled. No printer settings dialog is displayed.

```
project.printGfxReport("Report Graphics 1", true);
```

emptyPrintQueue

```
void emptyPrintQueue()
```

Empties the print queue. Current job will not be aborted.

```
project.emptyPrintQueue();
```

pausePrinting

```
void pausePrinting();
```

Suspends printing operations. Will not suspend the print of a page already sent to the printer.

```
project.pausePrinting();
```

resumePrinting

```
void resumePrinting();
```

Resumes previously suspended printing.

```
project.resumePrinting();
```

abortPrinting

```
void abortPrinting();
```

Aborts current print operation and proceed with the next one in queue. This command will not abort the print of a page already sent to the printer.

```
project.abortPrinting();
```

printStatus

`project.printStatus;`

Returns a string representing current printing status.

Status string	Description
error	An error occurred during printing
printing	Ongoing printing
idle	System is ready to accept new jobs
paused	Printing has be suspended

```
var status = project.printStatus;
project.setTag("PrintStatus",status);
```

printGfxJobQueueSize

`project.printGfxJobQueueSize;`

Returns the number of graphic reports in queue for printing.

```
var gfxqueuesize = project.printGfxJobQueueSize;
project.setTag("printGfxJobQueueSize",gfxqueuesize);
```

printTextJobQueueSize

`project.printTextJobQueueSize;`

Returns the number of text reports in queue for printing.

```
var textjobqueuesize = project.printTextJobQueueSize;
project.setTag("printTextJobQueueSize",textjobqueuesize);
```

printCurrentJob

`project.printCurrentJob;`

Returns a string representing current job being printed

```
var currentjob = project.printCurrentJob;
project.setTag("printCurrentJob",currentjob);
```

printActualRAMUsage

`project.printActualRAMUsage;`

Returns an estimate of RAM usage for printing queues

```
var myVar = project.printActualRAMUsage;
alert(" actual ram usage is   "+ myVar);
```

printRAMQuota

`project.printRAMQuota;`

Returns the maximum allowed RAM usage for printing queues

```
var ramquota = project.printRAMQuota;
project.setTag("printRAMQuota",ramquota);
```

printActualDiskUsage

`project.printActualDiskUsage;`

Returns the spool folder disk usage (for PDF printouts)

```
var myVar1 = project.printActualDiskUsage;
alert(" actual disk usage is   "+ myVar1);
```

printDiskQuota

`project.printDiskQuota;`

Returns the maximum allowed size of spool folder (for PDF printouts).

```
var ramquota = project.printRAMQuota;
var diskquota = project.printDiskQuota;
```

printSpoolFolder

`project.printSpoolFolder;`

Returns current spool folder path (for PDF printouts).

```
var spoolfolder = project.printSpoolFolder;
project.setTag("printSpoolFolder",spoolfolder);
```

printPercentage

`project.printPercentage;`

Returns current job completion percentage (meaningful only for multipage graphic reports)

```
var percentage = project.printPercentage;
project.setTag("printPercentage",percentage);
```

Group object

A group is a basic logical element that associates a set of logical tags.

Group object methods

Methods that can be used with group objects.

getTag

```
object getTag( TagName )
```

Gets the tag specified by TagName from the group object.

Parameter	Description
TagName	String representing the tag name

Return value

An object that is the value of the tag or, if tag value is an array, the complete array. If you need to retrieve an element of the array, check the method `getTag` available in the project object. Undefined is returned if tag is invalid.

```
var group = new Group();
project.getGroup("GroupName", group);
var value = group.getTag("Tag1");
```

(Available on web pages)

getCount

```
number getCount()
```

Returns total number of tags in this group.

```
var group = new Group();
project.getGroup("GroupName", group);
var value = group.getCount();
```

(Available on web pages)

getTags

```
object getTags()
```

Returns the list of all tags in group.

```
function {
var group = new Group();
```

```
project.getGroup("enginesettings", group);
var tagList = group.getTags();
for(var i = 0; i < tagList.length; i++){
     var tagName = tagList[i];
     //do something…
};
```

(Available on web pages)

State object

This is the class holding the state of a tag acquired from the controlled environment.

State object methods

Methods to be used with state objects.

getQualityBits

```
number getQualityBits()
```

Returns an integer - a combination of bits indicating tag value quality.

```
var state = new State();
var value = project.getTag("Tag1", state, 0);
var qbits = state.getQualityBits();
```

(Available on web pages)

getTimestamp

```
number getTimestamp()
```

Returns time the value was sampled.

Return value

A number containing the timestamp (for example 1315570524492).

Note: Date is a native JavaScript data type.

```
var state = new State();
var value = project.getTag("Tag1", state, 0);
var ts = state.getTimestamp();
```

isQualityGood

```
boolean isQualityGood()
```

Returns whether the value contained in this state object is reliable.

Return value

A Boolean true if quality is good, false otherwise.

```
var state = new State();
var value = project.getTag("Tag1", state, 0);
if (state.isQualityGood()) {
     // do something…
}
```

(Available on web pages)

Keywords

Global objects are predefined and can be referenced by the following names.

page

```
object page
```

References the page object for the current page.

```
function btnStd04_onMouseRelease(me) {
     var wgt = page.getWidget("rect1");
     var name = wgt.objectName;
}
```

project

```
object project
```

References the project widget.

```
var group = new Group();
project.getGroup("GroupName", group);
var value = group.getCount("Tag1");
```

Global functions

print

```
void print( message )
```

Prints a message to the HMI Logger window.

Parameter	Description
message	Message string

```
print("Test message");
```

alert

```
void alert( message )
```

Displays a pop-up dialog with the given message. The user must press the **OK** button in the dialog to continue with the execution of the script.

Parameter	Description
message	Message string

Note: The alert function may be used for debugging JavaScript functions.

```
alert("Test message");
```

(Available on web pages)

Handling read/write files

Create folder

```
boolean fs.mkdir(strPath);
```

Creates a folder, if not already existing, in the specified path. Returns true on success and false if it fails.

Parameter	Description
strPath	Path string

Remove folder

```
boolean fs.rmdir(dirPath);
```

Remove directory at strPath if exists and empty. Returns true on success and false if it fails.

Parameter	Description
dirPath	Folder string

Read folder content

```
object fs.readdir(dirPath);
```

Reads the contents of a folder. Returns an array of the names of the files in the folder excluding '.' and '..'. Returns empty list if it fails.

Parameter	Description
dirPath	Folder string

Read file

```
object fs.readFile(strfile [,strFlag]);
```

Opens the strFile file in read mode, reads its contents and returns it.

Parameter	Description
strFile	File name string
strFlag	Read file mode: “b” reads and returns as binary file (otherwise returns a text file)

Write file

```
fs.writeFile(strFile, fileData, [strFlag]);
```

Creates the strFile file if not present. Opens the strFile file in write mode and writes the data fileData to the file.

Parameter	Description
strFile	File name string
fileData	Data to be write on the file in byte array
strFlag	Write file mode: • “a”: appends fileData to the end of the text file • “r”: replaces the contents of the file with fileData • “ab”: appends fileData to the end of the binary file • “rb”: replaces the contents of the binary file with fileData

Default flag is for writing text file in append and write mode. File path will be created if not present.

Returns -1 if write error occurs.

Append file

```
int fs.appendFile(strFile, fileData);
```

If the files does not exist creates it, otherwise append to existing file. Returns the number of character written or -1 on error.

Parameter	Description
strFile	File name string
fileData	Data to be write on the file in byte array

File exists

```
boolean fs.exists(strPath)
```

Returns true if the file or folder exists at strPath.

Parameter	Description
strPath	Path string

Remove file

```
boolean fs.unlink(strPath)
```

Removes the given file at strPath from filesystem if exists. Returns true on success and false if it fails.

Parameter	Description
strPath	Path string

File status

```
object fs.stat(strPath)
```

Retrieves information on the file/folder present at the specified path.

Parameter	Description
strPath	File/folder path string

```
var fileStats = var fs.stat(strPath)
```

fileStats.isFile	True if path is a file
fileStats.isDir	True if path is a folder
fileStats.size	Size in bytes of that file
fileStats.atime	Date object representing the last read access time
fileStats.mtime	Date object representing the last write access time
fileStats.ctime	Date object representing the creation time
fileStats.perm	File permissions

If path is invalid both isFile and isDir fields return false.

File permission table

0x4000	File is readable by the owner of the file
0x2000	File is writable by the owner of the file

0x1000	File is executable by the owner of the file
0x0400	File is readable by the user
0x0200	File is writable by the user
0x0100	File is executable by the user
0x0040	File is readable by the group
0x0020	File is writable by the group
0x0010	File is executable by the group
0x0004	File is readable by anyone
0x0002	File is writable by anyone

Important notes on file handling

Path for files and folders are expected to be UNIX style. This means the backslash character (\) is not recognized. Use slash character (/) instead.

File system object is a client side object. So operations are performed on local file system, not on server file system.

Current JavaScript API to get access at the device file system has been designed to manipulate small files. When a file is read, the entire file contents is temporarily stored inside the RAM available for JavaScript environment (16MB) and an exception is raised when there is not enough available memory. Good programming practice is to include the `fs.readFile()` call inside a try/catch block.

Sign in from JavaScript

Using the project.login() and project.logout() function is possible automatize the user sign in from a remote device. This could be useful, e.g., to perform the sign in by reading a user badge with a badge reader device.

This chapter show an example of how configure the application to manage the sign in by a remote device.

The application must have a default user

Since the project's functions are working only when the application is active, the application must start with a default user, maybe with read only privilege. Reading the badge, the application can be switched to a user with additional privilege. Later, the logout command will reactivate the default user without any particular privileges

In the below example we are using three tags to communicate with the remote device:

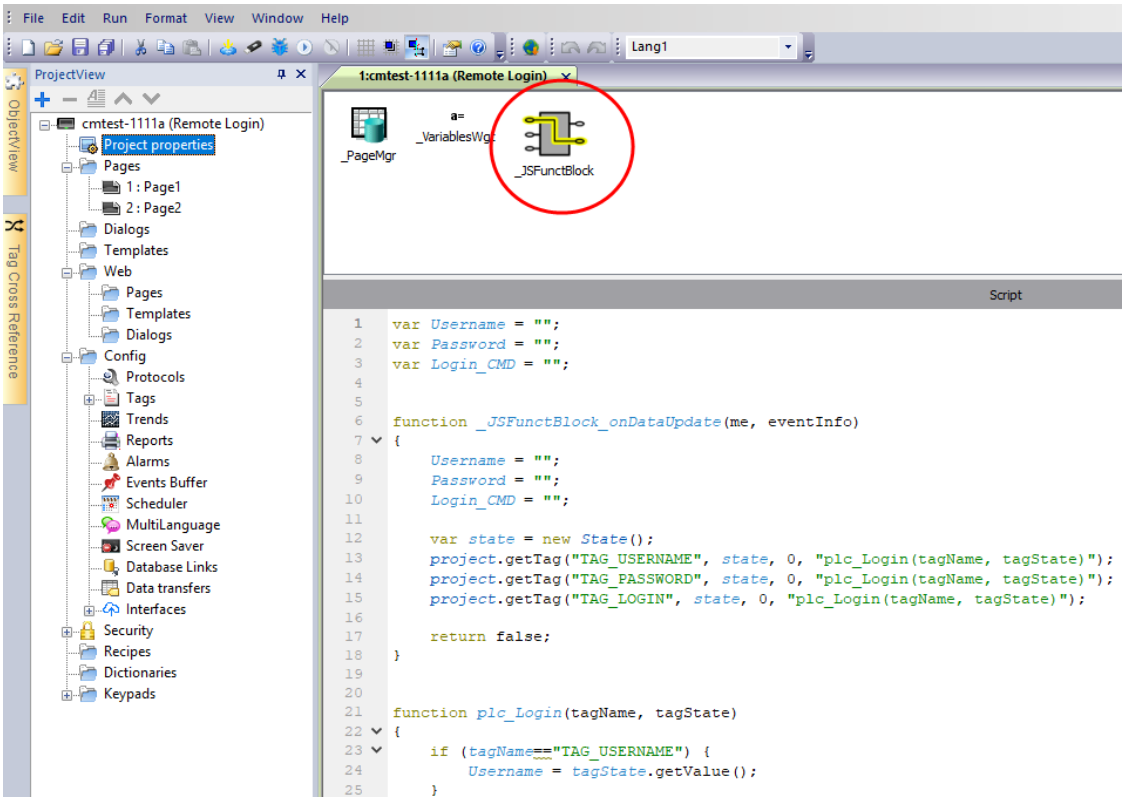
- TAG_USERNAME
- TAG_PASSWORD
- TAG_LOGIN

The TAG_LOGIN will be the command code to execute.

The remote device has to fill the required TAG_USERNAME and TAG_PASSWORD parameters, then fill the TAG_LOGIN parameter with the required login or logout command. Engine on HMI-RUNTIME will detect the TAG_LOGIN changes and perform the required command, then reset the TAG_LOGIN to its idle status.

TAG_LOGIN Commands	
0	Idle
1	Login request
2	Logout request

At the project level, we have to add a JavaScript function block to detect when TAG_LOGIN will changes. The JavaScript code attached at the OnDataUpdate Action of the JavaScript function block will execute the required login/logout command.



The JavaScript code attached at the OnDataUpdate Action

```
var Username;
var Password;
var Login_CMD;

function _JSFunctBlock_onDataUpdate(me, eventInfo)
{
    Username = "";
    Password = "";
    Login_CMD = "";

    var state = new State();
    project.getTag("TAG_USERNAME", state, 0, "plc_Login(tagName, tagState)");
    project.getTag("TAG_PASSWORD", state, 0, "plc_Login(tagName, tagState)");
    project.getTag("TAG_LOGIN", state, 0, "plc_Login(tagName, tagState)");
    return false;
```

```
}

function plc_Login(tagName, tagState)
{
    if (tagName=="TAG_USERNAME") {
        Username = tagState.getValue();
    }
    if (tagName=="TAG_PASSWORD") {
        Password = tagState.getValue();
    }
    if (tagName=="TAG_LOGIN") {
        Login_CMD = tagState.getValue();
    }

    if (Username!="" && Password!="" && Login_CMD!=""){
        if (Login_CMD==1) {
            Reply = project.login(Username, Password);
        };
        if (Login_CMD==2) {
            Reply = project.logout(false); // Logout only if not logged as default
user
        };
        project.setTag("TAG_LOGIN", 0);
        project.setTag("TAG_REPLY", parseInt(Reply));
    }
}
```

See also:

- "login" on page 495

Limitations in working with widgets in JavaScript

Widgets cannot be instantiated by JavaScript, they can only be accessed and changed. If you need additional widgets on the page, you can add hidden widgets on the page, and then display or position them using JavaScript.

Debugging of JavaScript

LRH SW and LRH SW HMI Runtime include a JavaScript debugger.

Two types of debuggers are available:

- Runtime debugger: a debugger running directly on the HMI device
- Remote debugger: a debugger running on a remote computer connected to the HMI device via Ethernet (usually computer running LRH SW)

Enabling debugging

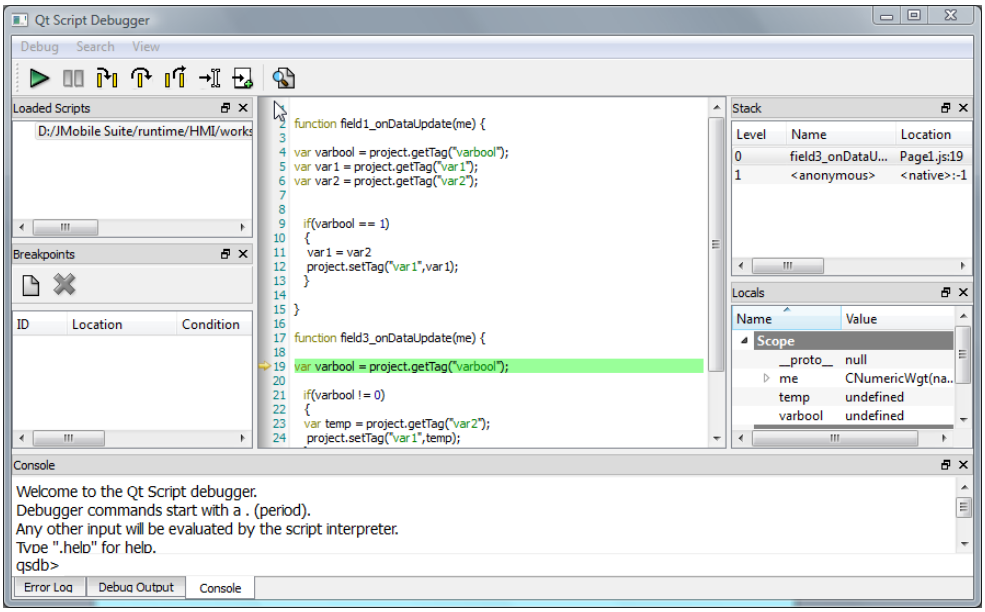
In the **Properties** pane of a page, set **JavaScript Debug** to **true**.

Project Widget	
Id	Project
Full Path	
Version	
Context Menu	on delay
Developer Tools	false
Keyboard	true
JavaScript Debug	true
Allow JavaScript Remote	true

Page	
Id	Page1
Width	1024
Height	768
Background	[255, 255, :
Template	none
Static File Type	png
JavaScript Debug	true

For schedulers and alarms debugging, enable JavaScript Debug in Project properties.

In the LRH SW HMI Runtime, when the events are called, the debugger will show the debug information. In the **Locals** pane you can inspect all variables and elements.



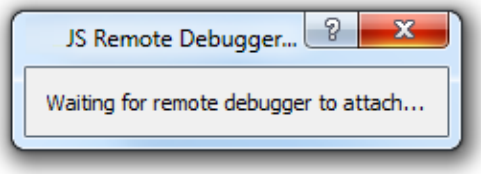
For a complete reference guide about JavaScript Debugger refer to :

http://qt-project.org/doc/qt-4.8/qtscriptdebugger-manual.html

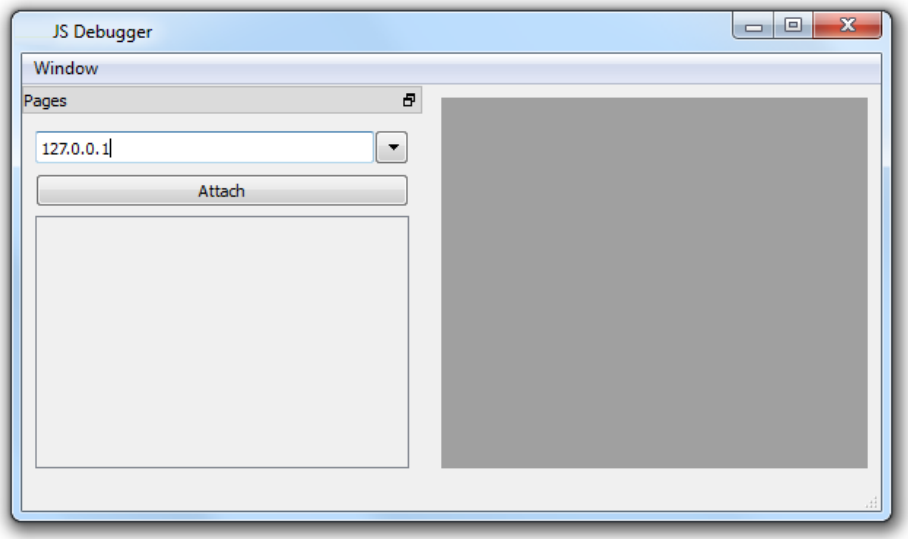
Remote JavaScript Debugger

Path: ***Run> Start JS Remote Debugger***

1. Set the **Allow JavaScript Remote** and the **JavaScript Debug** parameters in the project Properties to true in all the pages where debugging is required.
2. Download the project: the following message is displayed on the runtime.



3. In the **JS Debugger** window, select the IP of the HMI device and click **Attach** to connect the debugger to the HMI device.



Remote JavaScript debugger connects to LRH SW HMI Runtime using port 5100/TCP.

Note: The Remote JavaScript debugger tool is not supported in LRH SW Client.

JavaScript Memory Usage

When the memory exceeds the maximum, an out of memory exception is thrown with a custom message. Please note that we do not have a fine control over the actual memory usage so it is mainly a soft limit. Moreover we can't forbid the allocation (this will break the engine implementation), so exception is thrown only when the memory is already over the limit. Before raising the exception, a garbage collection is forced to see if some memory can be freed.

JavaScrip memory limit can be accessed from the global object **$EngineMemory**. The default is 16MB, which should be enough for the typical JavaScript usage (mainly control, without many allocations).

- $EngineMemory.setLimit()
  set maximum memory allowed for JavaScript (the default limit is 0x00FFFFFF)
- $EngineMemory.getLimit()
  get maximum memory allowed for JavaScript
- $EngineMemory.getSize()
  get currently used memory from JS (fastMallocStat)

Test memory exception

To generate and test memory exception you can use the following snipped. Please note that we need to reset the memory limit to 0xffffffff to be able to run the alert, otherwise the memory allocations required to pop up the alert would fail.

```
try
{
    // Generate out-of-memory error
    var a = [];
    while(1)
    {
        a.push("a");
    };
} catch(e)
{
    // Ensure there is enough memory to pop up error message
    $EngineMemory.setLimit(0xffffffff);
    alert("Exception: " + e);
};
```

41 Handling Gestures

Some widgets have the capability to detect and manage pan and pinch gestures.

- Trends (see "Trend widget gestures" on page 274for details)
- Alarms Widget
- Combo box Widget
- Table Widget
- PDF Viewer
- Gesture Area Widget. Special widget designed to customize handling of gesture events (see "Gesture area widget" on page 414 for details)

For widgets based on table presentation, when the **Scrollbars Type** parameter has been set to "Gesture", the pan gesture is used to smoothly scroll the table.

- Alarms
- Control List

WARNING: Pinch and Rotate gesture requires two fingers. Them are available only with HMI devices supporting multi touch operation (see "HMI devices capabilities" on page 541)

Tip:Using multi touch HMI device you can implement safe commands by programming a command to be executed only when two buttons are pressed at the same time.

42 Web access

LRH SW allows users to access HMI projects from a remote web browser running on a computer or on a mobile device such as a tablet or a phone. With LRH SW, users can create a web project to display at a remote location the same graphical display shown on the HMI device. LRH SW projects are based on HTML5 technology which means that no plug-ins or external software is needed for displaying the information.

This document assumes that you have a basic understanding of how to operate the web browser on your mobile devices as well as how to set up a connection to the HMI device where the server is running. For example, you must know how to set-up Wi-Fi access if you are working with tablet or phone devices to access the LRH SW pages on the HMI device.

Supported platforms and browsers

LRH SW supports 3 platforms:

- web, for desktop browsers,
- phone, for smart phone devices
- tablet, for tablet devices

You can therefore create pages of different content and size for the different platforms. For example, you may want to create a set of smaller pages in your project for phones whereas you will use full size pages for desktop web browsers and tablets.

Working with a computer

LRH SW works with all modern web browsers. The following browsers have been tested for compatibility with LRH SW:

- Mozilla Firefox 52+
- Microsoft Edge 42+
- Apple Safari 11+
- Google Chrome 57+

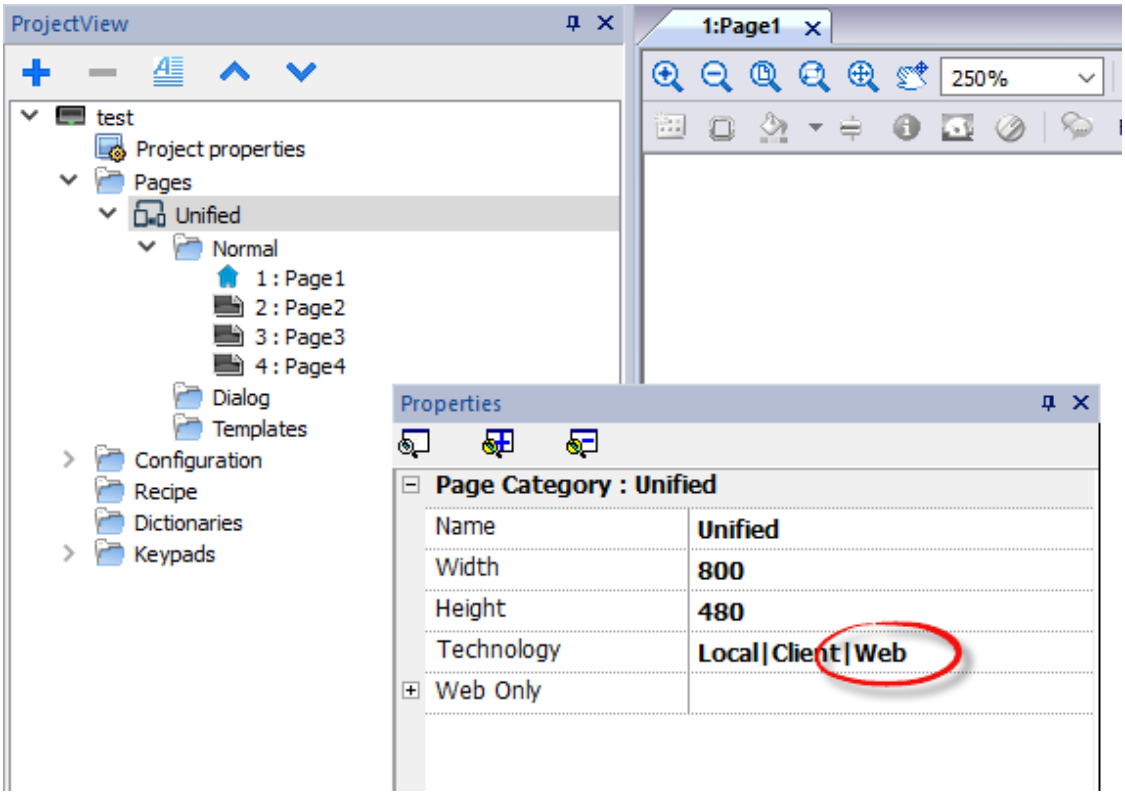
Working with tablets or phones

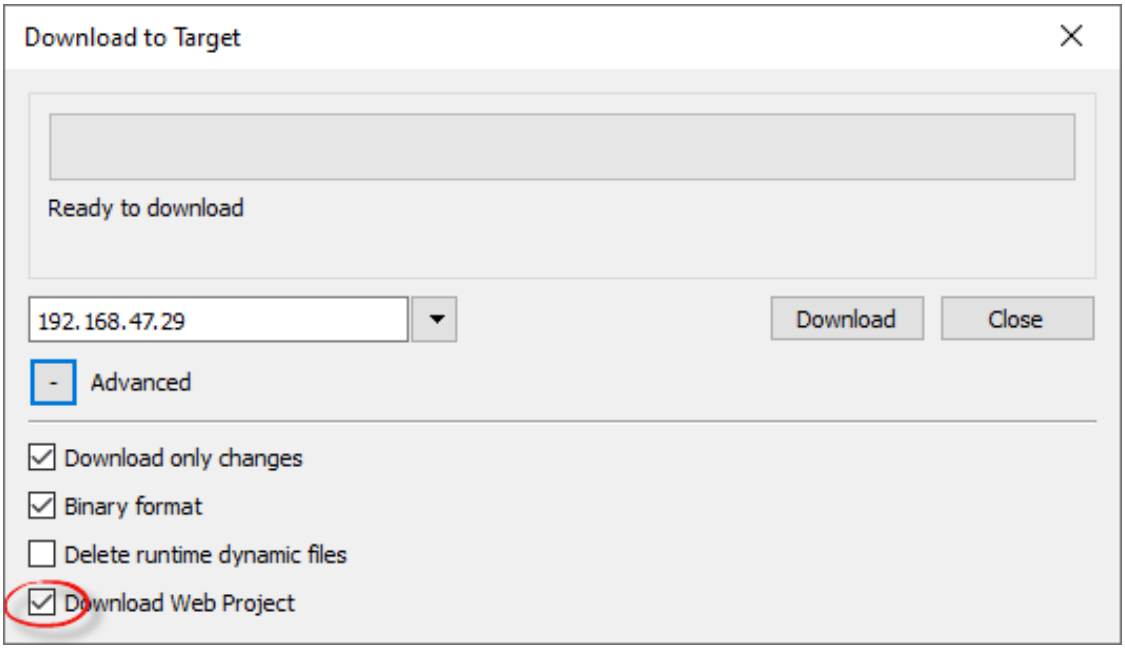
LRH SW works with most tablet and phone devices. The following tablets have been tested for compatibility with LRH SW:

- iOS 10+ - Mobile Safari
- Android 7+ - Chrome for Android 55+

Web pages

To enable web clients to access at the pages is necessary to include the "Web" reference in the Technology parameter of the page's category and make sure that when you download the project to the HMI device the **Download Web Project** option is selected (normally checked by default).





If the application needs to send different pages to different web clients (e.g. Smart phone instead of PC browser, etc.), have a look at the "Differentiated pages " on page 66 chapter.

Web page properties

Any widgets and features can be used in LRH SW; however, not all features are currently available in LRH SW. If the project includes a feature that is not available, LRH SW will still work correctly but the feature will not be available on the

remote client device. See "Web supported features" on page 521 for a list of the features supported in LRH SW and of the existing limitations.

You can use the **Project Validator** tool to check if your project contains widgets configured with properties that are currently not supported in Web technology (see "Project Validator" on page 64)

In addition to the standard page properties, there is an additional property to configure how the page will be adapted to the browser's viewport.

Property	Description
Fit to Screen Size	How the page will adapted to the browser's viewport • None • Fit to Screen Simple modify the zoom level to adapted the page to viewport of the browser • Responsive Design Smart modify the zoom level to adapted the page to viewport of the browser respecting the restrictions defined inside the grid layout

Redirect to specific page using url request

You can access a specific web page by entering an URL with this syntax:

```
http://address/index.html?loadPage=pageName
```

Testing the Web project

You can test your LRH SW project using the online simulator opening a standalone web page directly from a browser.

Testing with the online simulator

LRH SW includes an web server in the online simulator. You can start the simulator and access your LRH SW project from a web browser. The pages will be served from the simulator.

1. Create your project (see "Web pages" on page 516).
2. On the **Run** file, choose **Start Simulator**: the project will start running in a separate window.
3. Open a web browser (see "Supported platforms and browsers " on page 516 for a list of browser compatible with LRH SW).
4. Enter the following address: http://localhost:81: this tells the web browser to read the web pages from the local computer and use port 81, used by default by the online simulator in LRH SW.
5. Test your project in the browser.

Important: If you make any changes to the project pages in LRH SW you must stop and restart the simulator.

Note: If you are using a device (for example, a smartphone) that is not the localhost where the simulator is running, you will be required to enter username and password.

Downloading the Web project

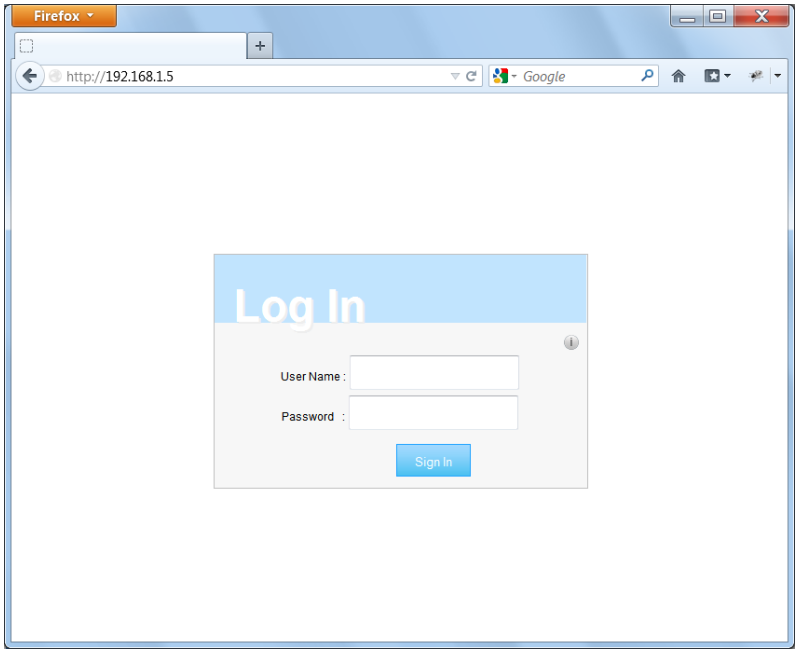
After testing the LRH SW pages, you can download the project to the desired HMI device.

The LRH SW project is downloaded together with the LRH SW project, see "Download to HMI device" on page 92 for details.

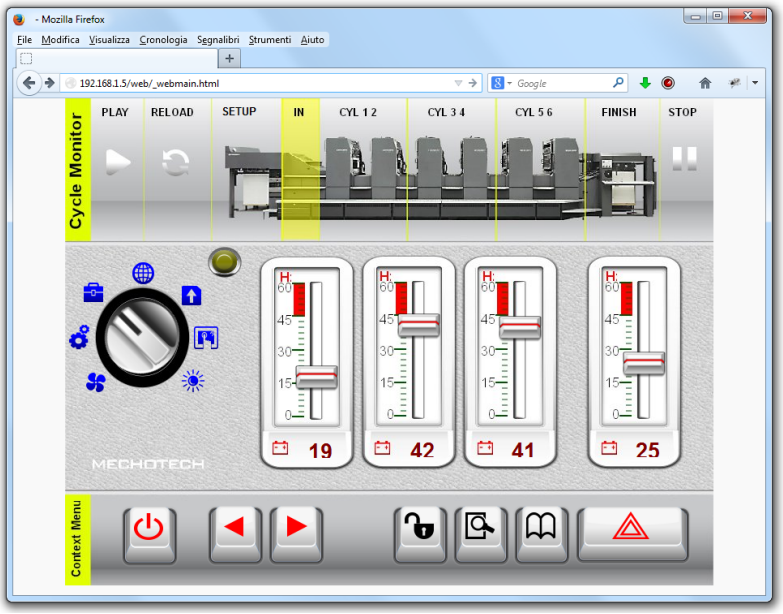
After the download process is completed, the HMI project automatically starts on the HMI device and the LRH SW project is ready to be used.

Running LRH SW from a browser

1. Open a web browser and enter the IP adress of your HMI device: the login page is displayed.



2. Enter **User Name** and **Password** and click **Sign In**: the Home page will be displayed.



See "User management and passwords" on page 325 for details on how to create credentials.

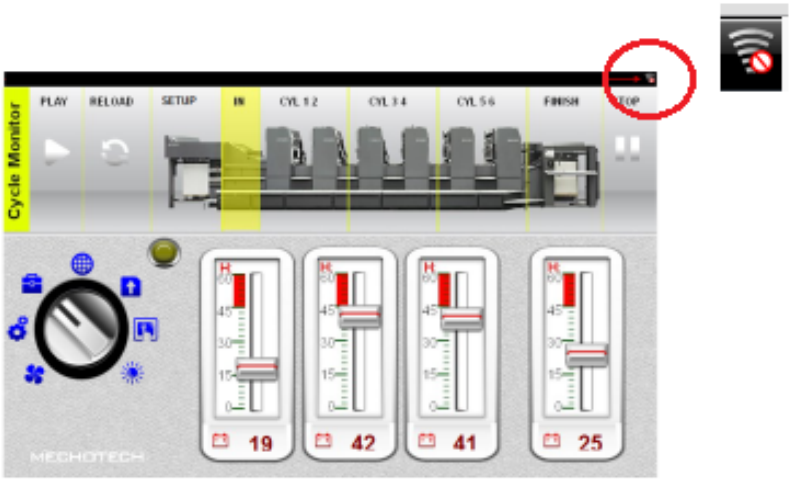
You can interact with the project using the browser in the same way you interact with a device when touching the screen: click buttons to change pages, view indicators and gauges, drag slider handles to change values, and so on. The LRH SW project will manage all communications with the web server while you are interacting with the HMI device remotely.

Web connectivity issues

Here are described the most common issues you might encounter when connecting remotely to your HMI device.

Server disconnection

Since LRH SW runs remotely from the HMI device, the server might disconnect from the browser (for example if the server is stopped or the network cable is unplugged). If this happens, a ‘disconnect’ icon will appear in a toolbar on top of the LRH SW as in this example.



Once the server is back online, the red circle-bar icon will disappear indicating normal communications with the device.

The “Connection status” system variable can be used to know the status of the connection. See "Remote Client variables" on page 133 for additional details.

Note: If you make changes in the LRH SW pages while the server is disconnected, these changes will be visible on the client but will not be transferred to the server until the connection is restored.

Inactivity timeout

LRH SW will require you to re-enter your login credentials if the browser has been inactive for several minutes. If no activity is detected for 10 minutes, the login screen will reappear and you need to enter your login credentials to continue operation. A timeout feature guarantees that no unauthorized access is possible.The web inactivity timeout can be modified from the **Project Properties** table.

User session termination

A user session can be terminated either from the server or from the user.

In specific conditions the server might send a request to the client (browser) to perform the login process. In this case the user is redirected to the login page and then back to the page where he was working. This will happen for example if the user clears the browser cache or browser cookies.

Note: If the user is working in a dialog when redirected to the login page, he will be then redirected to the page from which the dialog was opened.

Non-Active LRH SW Project

The LRH SW page displayed in your browser might come from a project that is no longer active in the device. In this case a confirmation box is displayed and you can return to the active project.

Note: This redirection assumes that the current active project has LRH SW pages in it.

If you choose to stay in the non-active project all the actions you perform in the browser may not be executed properly as the LRH SW cannot perform any server-bound communication.

Web supported features

Some features or widget's properties are not supported by LRH SW. When not supported widgets are used, you will get the widgets only on the pages in the HMI panel, while on the web pages the unsupported widgets will be not visible. Note that you can run the **Project Validator** to check if pages contains unsupported widgets (see "Project Validator" on page 64)

List of widgets that are not supported

- Analog Clocks
- Analog Video
- BACnet
- Button gstArea
- Consumption meter
- Control List
- DateTime Combo (Date or Time can be set using other widgets)
- IP Widget (The IP can be configured via system settings with a browser)
- IPCamera (Supported for Chrome and Firefox. See "Web Browser" on page 431 for additional details)
- Media Player
- Multistate image multilayer (Multistate image widget can be used)
- Rotation menu widget
- RSS Feed
- RSS Scroll
- Scheduler
- Text Editor
- Web Browser
- Hyper Link

List of actions that are not supported

Widget	SlideWidget, BeginDataEntry, TriggerIPCamera, MoveIPCamera, RefreshEvent, ContextMenu, ReplaceMedia, OpenComboBox, CloseComboBox, SelectAlarmsOnSrc, ShiftTableDataSrcColumns, ResetTableDataSrcColumns, SetTableSortingColumn
Web Browser	All actions are not supported
Text Editor	All actions are not supported
MediaPlayer	All actions are not supported
Mail	All actions are not supported
FTP	All actions are not supported
Keypad	All actions are not supported
Page	LaunchApplication, LaunchBrowser, LaunchVNC, LaunchPDFViewer, LaunchUpdater, LaunchHMICloudEnabler, LockScreen, LoadProject, LastVisitedProject
Print	All actions are not supported
Tag	DataTransfer, ActivateGroup, DeactivateGroup, EnableNode, BACnetClearPriority, BACnetClearAllPriorities, BACnetSetPriority, ClearRetentiveMemory, ForceReadTag
Trend/Graph	ConsumptionMeterPageScroll

System	Restart, ResetProtoErrCount, SafelyRemoveMedia, ControlUserLED, SaveEventArchive, LogMessage
Database Actions	All actions are not supported
UserManagement	SwitchUser, ResetPassword, AddUser, DeleteUser, EditUsers, DeleteDynamicFiles, ExportUsers, ImportUsers
RemoteClient	All actions are not supported

List of features that are not supported

- Context menu
- Buzzer on touch
- Javascript debugger
- Wheel actions (Browser use wheel events to manage scroll bars)
- Combo box full-screen mode (Standard "context" mode is supported)
- Keypads
- ScreenSaver
- External Database (SQL4Automation)
- Display Rotation
- Electronic Signature

System Variables

Using the "Attach To", only the system variables listed below are supported, while all system variables are supported using the protocol "System Variables"

- System Time
- X Screen resolution
- Y Screen resolution
- This Client Group-Name
- This Client User-Name
- Connection status
- This Client ID
- Available System Memory
- Current Language Id
- Current Language Name
- Current Language Code

Font files

Font files without "*Font Embeddability* = *Installable*" property (to be verified in the font file properties > details) are not loaded from the LRH SW unless the font is already installed in the operating system of the device running the browser

Allarms

- Alarm color based on trigger condition is not supported in Web
- Can not edit the Alarm widgets in runtime
- On Smartphone/Tablet (in general embedded devices) based on HW a user could expect performance problems with > 500 alarms.
- Page actions are not supported in alarm trigger condition

Others

- The dialog pages support only modal dialogs.
- Some specific widgets properties are not yet supported, in this case, the default value is used. You can use the Project Validator to check if the used widgets contain properties that are not supported (see "Project Validator" on page 64).

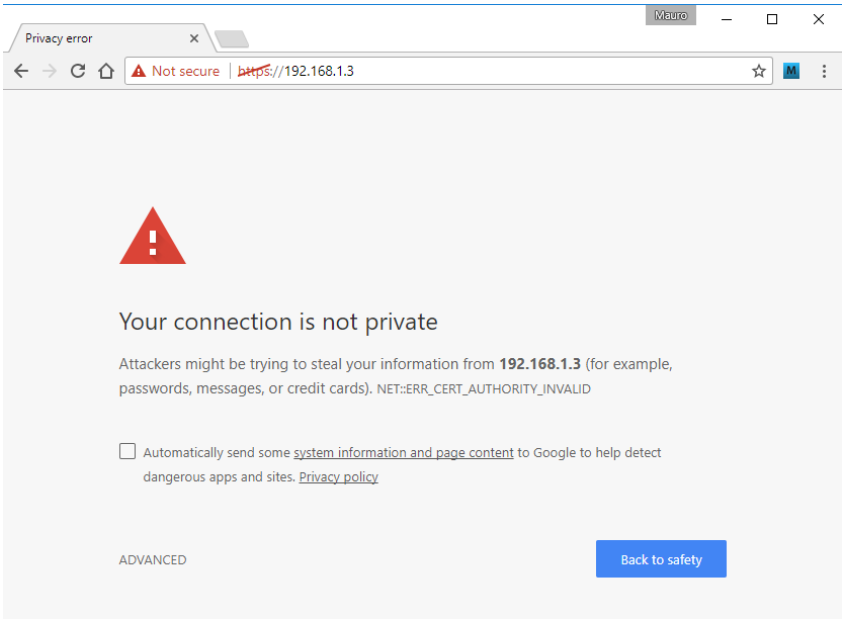
Secure Socket Layer (HTTPS)

Linux devices support the Transfer Protocol over Secure Socket Layer (HTTPS). To use this protocol access at the web page using the below syntax:

https://<device_ip_address>

Note that since the self-certificate provided from the HMI device is not firmed from a known Authority, you will get a warning message.

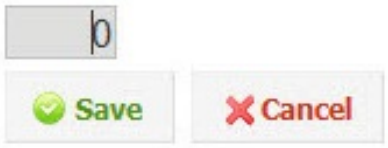
Simple click the ADVANCE button to continue.



Working with keypads in LRH SW

The user can click on the Numeric widget and a text box will be displayed in which the new value can be inserted.

After inserting the value the user can either press **Enter**, or equivalent in touch devices, or click **Save** to make the newly inserted value permanent. Only meaningful numbers will be accepted during the save process. Anything else will be ignored and will not result in a value change.



Troubleshooting and FAQ

Enable JavaScript

LRH SW requires JavaScript to provide interactivity with the server and the user. LRH SW will not work if JavaScript is disabled in your browser.

By default most browsers come with JavaScript enabled. But if you have disabled JavaScript in the past, please re-enable JavaScript before accessing LRH SW pages.

Browser cache

LRH SW includes resources that change infrequently such as CSS files, image files and JavaScript files. These resources take time to download over the network which increases the time required to load the LRH SW page in your browser. Browser caching allows these resources to be saved by a browser and used without requesting them each time from the server. This results in faster loading of LRH SW pages.

Caching is normally enabled by default, for optimal LRH SW performance make sure it has not been disabled.

Note: LRH SW pages will still work properly with disabled browser caching, however resource loading time will be slower compared with normal cached operations.

Using a proxy

Some users may be accessing the LRH SW project through a proxy. The proxies may control the number of parallel connection for the browser.

Make sure that the maximum parallel connections allowed (max connections) is not more than 16 and not less than 12.

Why I'm not able to see changes in the web pages?

Every time a new web page is added edited into the project, you need to download the project to the device. However, when you connect the device IP address, the web browser might display cached pages instead of the latest downloaded pages. To avoid this behavior you can:

- disable cache of your web browser
- force web page refresh
- by-pass browser cache

Privacy

We do not use cookies to collect private information from any user.

A cookie is a piece of data stored on the user's hard drive containing information about the user. Usage of a cookie is in no way linked to any personally identifiable information while on our device. Once the user closes their browser, the cookie simply terminates.

43 Protecting access to HMI devices

The following operations are password protected on the HMI device:

- LRH SW HMI Runtime management: install LRH SW HMI Runtime and update LRH SW HMI Runtime
- Board management: replace main BSP components such as Main OS, Configuration OS, Bootloader, and so on
- Download and upload of project files
- Optional services on Linux devices (e.g. SSH Protocol, VNC Server)

WARNING: For security reasons

- Change the default passwords (See: "Password protection" on page 563 for HMI devices on Linux platform)
- Enable security management (See: "Enable/disable security management" on page 326)
- Force remote login (See: "Force remote login" on page 336)

WARNING: Unauthorized access to the device can cause damage or malfunctions. When connecting the device to a network protect the network against unauthorized access.

Measures for protecting the network include:

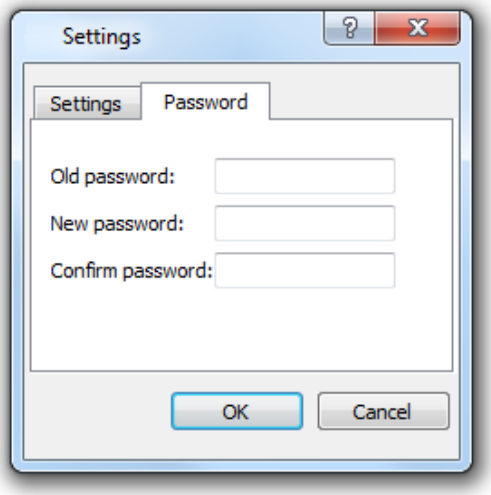
- Firewall
- Intrusion Prevention System (IPS)
- Network segmentation
- Virtual LAN (VLAN)
- Virtual Private Network (VPN)
- Security at physical access level (Port Security).

Further information, guidelines and standards regarding security in information technology: IEC 62443, ISO/IEC 27001.

Changing password on HMI device

To change the password on the HMI device, use one of the following methods:

- From the LRH SW HMI Runtime context menu: **Settings**> **Password** tab.



- Use the **Set Target Password** function in update package: the password is updated by LRH SW HMI Runtime just after the update process is completed.
- Using HMI device "System Settings" on page 543 Tool

Leave “Old password” empty as default if target password is not set.

Ports and firewalls

Here a list of all the ports used by LRH SW components.

Port	Usage	Remote Access	Board Management	Runtime/Project Management
80/tcp	HTTP port	Yes	-	Yes
21/tcp	FTP cmd port	-	-	Yes
2100/tcp	Board port	-	Yes	-
16384-17407/tcp	FTP data port (passive mode)	-	Yes	Yes
990/udp	UDP broadcast (Device discovery)	-	Optional	Optional
991/udp	UDP broadcast (Device discovery)	-	Optional	Optional
998/udp	UDP broadcast (Device	-	Optional	Optional

Port	Usage	Remote Access	Board Management	Runtime/Project Management
	discovery)			
999/udp	UDP broadcast (Device discovery)	-	Optional	Optional
5900/tcp	VNC Server	VNC only	-	-
5100/tcp	JS Remote Debugger	-	-	Optional
48010/tcp	OPC UA Server	-	-	Optional
25/tcp	SMTP Server	-	-	Optional
See your MQTT Broker	MQTT	-	-	Optional

Remote access

Remote access is required to connect to LRH SW HMI Runtime using:

- LRH SW Client
- Internet Browser

Runtime and project management ports

You use these ports to connect to LRH SW HMI Runtime for operations such as update, installation and project download.

Board management ports

You use these ports to connect to the HMI device for Board operations such as BSP update, splash image download and so on.

Note: When broadcast service is not available, for example in VPN networks, type in the exact IP address to connect to the HMI device from LRH SW.

44 Tips and tricks to improve performance

LRH SW allows great flexibility for a project designers.

Follow these guidelines to create projects that perform better in terms of boot time, page change and animations.

Static Optimization

Static optimization is a technique used in LRH SW to improve runtime performance.

Using a lot of images and pictures in a project might degrade performances, static optimization merges several images into a single background image thus reducing rendering and loading times. Using this method only one raster image needs to be loaded and rendered instead of many single raster and/or vector images.

When you create a project in LRH SW, the pages might contain widgets such as texts, images, background images, background colors and so on which can be classified as:

- Static: values or properties do not change at runtime.
- Dynamic: values or properties change at runtime.

Note: Based on security settings, static parts of widgets could be not merged to background. This happens when a widget is configured as “hide” in security settings.

Important: When you change the properties of widgets with JavaScript set the widget Static Optimization to Dynamic, otherwise changes to properties will be ignored.

When downloading or validating a project, LRH SW identifies static components and renders them as background images to .png files. These background images are saved as a part of the project under the folder */opt*.

Background images can be created as follows:

- full page background images, containing all widgets merged to page background
- group background images, containing a group of static widgets merged together to form a group background. For example, the Gauge group is normally composed by a background, a scale, a label and a needle, where background scale and label can all be merged to a single background image.

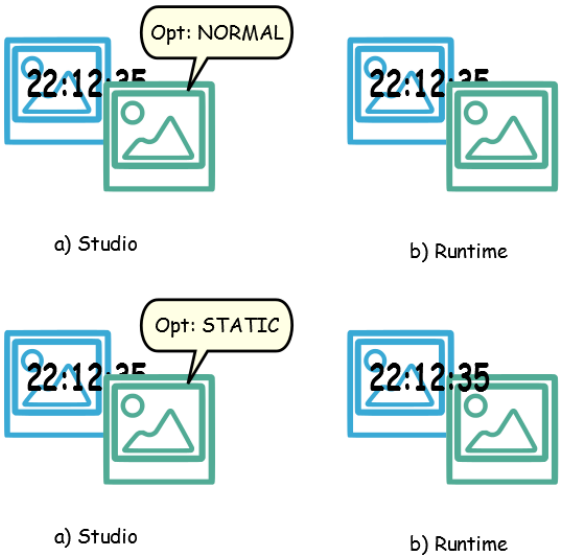
The **Static Optimization** page attribute enables and disables static optimization of the whole page. If it is set to **false** the optimization is totally disabled.

Finer control can be achieved setting the **Static Optimization** attribute of each single widget as follows:

- **Normal**: LRH SW automatically detects if the widget can be merged with the background. This can be used if the widget is not a dynamic widget and does not overlap, that is it is not stacked above, a dynamic widget.
- **Static**: The image is forced to be merged with the background. This can be used when the static widget overlaps a dynamic transparent widget.

Note: In this case the automatic optimization will fail because it does not make any assumption on invisible areas which might be rendered at runtime.

- **Dynamic**: The widget is not optimized at all. Use this flag when a static widget needs to be changed by JavaScript.



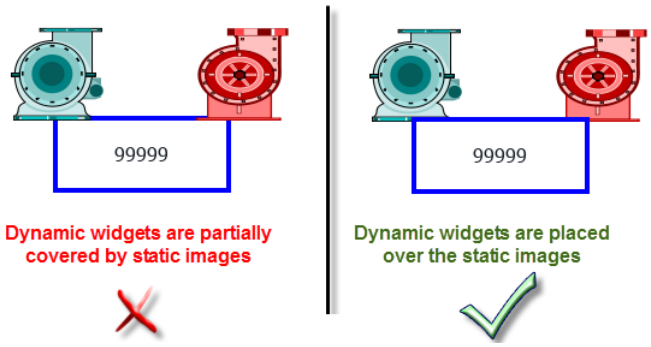
Tips for best performance

1. First of all: avoid placing static widgets over a dynamic widget. The overlapping area is computed considering the bounding rectangles of the widgets, that is the rectangles delimited by editing handles.
2. Do not use static optimization if your pages contain almost only dynamic objects. Static optimization would save many almost identical full size images for each page using up a lot of memory space that could be more effectively used to improve project performance with other techniques (such as, for example, page caching).
3. Bounding rectangles can include transparent areas, minimize transparent areas (for example splitting the image in multiple images) since they can be a waste of resources even when optimized.
4. Optimize image size. The image will be rendered at the size of the image widget containing the image. For best performances the widget needs to be the same size of the image.
5. Avoid using **Scale to fit** for image widgets, since this forces a rescaling at runtime for dynamic images and “hides” the actual image size during editing.
6. Use **Size to fit** to make the widget to the real size of his contents.
7. If overlapping cannot be avoided make sure to place the static widgets in the back, that is behind the dynamic widget.
8. Choose the image file format based on the HMI device you are connecting to.

9. Avoid using too many widgets in a single page. Often widgets are placed outside the visible area or their transparency is controlled by a tag. Since widgets are loaded even if they are not visible, having too many widgets in a page can significantly slow down the page change time.
10. Split a page with many widgets into multiple pages with less widgets.
11. For popping up new graphic elements in a page, prefer dialog pages with controlled positioning to transparent widgets.
12. Check the *opt* folder to see if static optimization is working as expected, the widgets z-order might need to be adjusted.
13. Numeric fields are often used to run JavaScript code on OnDataUpdate event even if the widget doesn't need to be visible on the page. In this case place the widget outside the page visible area instead of making it invisible, altering font color or visibility property. In the latter case you might end up with many left over wedges.
14. Use a HotSpot button if you need a touch area to react to user inputs.
15. If you reuse a widget from the gallery or you create your own, remember to set the correct optimization properties. For example button widgets are dynamic widgets, if you use a button widget just for its frame it won't be optimized since the button widget is dynamic. If you just need the frame you should use the Up image.
16. With many pages having many dynamic widgets and using a common template:
    1. set template static optimization to **true,**
    2. set page static optimization to **false**, since the background is already provided by the template.

    In this scenario the background image can be reused by many different pages thus saving memory space.
17. Do not use dynamic widgets, such as buttons, only for graphic purposes, when the button function is not needed, use image widgets instead to obtain the same graphical effect.

Here is an example of a correct and an incorrect use of static optimization.



Supported image formats

LRH SW supports several raster formats like BMP, PNG, JPEG, TIFF and the vector format SVG. Here a list of pros and cons:

Image format	Pros	Cons
RASTER	• Fast rendering • Well standardized	• Big file size • Fixed resolution
VECTOR (SVG)	• Small file size • Rescale without quality loss • Can handle dynamic properties	• Complex SVG images with many graphic items and layers can be slow to render. • Creating an optimized SVG is not simple. • Only Tiny 1.2 (http://www.w3.org/TR/SVGTiny12/) supported.

Note: Scour software is free tool that can be used to remove foreign code from file (http://www.codedread.com/scour/).

Static optimization of templates

Template pages can have large amounts of static content. However, static optimization cannot be applied to a template page, since where the template is used is based on the page design.

If a huge background image should be repeated in every page that uses the same template, this would increase the footprint of the device as the same static image would be created for each of the pages using the template page.

FAQ on Static Optimization

Q: In a page where there are a few identical widgets, in the *opt* folder I see a PNG for each one of them. If they are really identical, why should the software duplicate them instead of having just one PNG?

A: The software does not know if static images are actually the same since each widget could have different settings/properties altering the actual rendering at runtime.

Q: Why are the static images stored in a separate folder called *opt* instead of storing them directly in the project folder?

A: This avoids name collisions and allows skipping the upload of optimization images

Q: Why are the static images stored as a PNG files instead of common JPEG files?

A: PNG format uses a lossless compression for images and supports transparencies. JPEG files would render fuzzier compared to the PNG files with a different result in LRH SW(not using optimization) and LRH SW HMI Runtime.

Q: What will happen when no optimization is done in the software?

A: Every single widget is rendered at runtime. In particular SVG images may require a lot of time to render in an embedded platform.

Page caching

Once accessed all pages are kept in a RAM cache up to the maximum allowed cache size depending on the actual platform's available RAM. This allows a much faster access since cached pages, once reloaded, only need to re-paint their content without reloading all page resources.

Image DB

Image DB is a technique used to track the usage of image files and reduce the cost of image loading by caching most frequently used images (example, Push Button images, Gauge needles, Slider thumbs and so on). The same image used in many different places is therefore loaded just once.

The image DB function will preload the top most used images at startup until memory limit is reached. This would further improve the individual page loading times.

The file imagecachelist.xml is created in *project/opt* folder, containing relevant information:

- Fill color (in case of SVG images)
- Size of SVG image
- Number of times an image is used in the project
- Number of different sizes for the same image

Tips for using the Image DB function

1. Use uniform size of buttons, gauges and other widgets wherever possible.
2. Use same color themes among widgets of the same kind.

Precaching

The Precache attribute of pages can be used to notify LRH SW HMI Runtime to preload some pages in RAM at boot time for quicker access. Precaching is useful for complex pages having many dynamic widgets.

When this function is enabled on a page, access to the page is faster, however it also slows down boot-time since the system is not ready until all pages to be precached are not saved into the RAM.

Tips to precaching

1. Enable the precache function just for few pages having many dynamic widgets or for pages frequently used by users.
2. Do not enable the precache function for all the pages in the project since you would run out of memory and have no benefit at all.
3. Disable static optimization for pages where the precache function is enabled to reduce memory used.

FAQ on precaching

Page limit for precaching

Based on the size and complexity of a page, the space required for precaching can be from 1,5Mb to 3Mb.

When a project is loaded, LRH SW HMI Runtime proceeds as follows:

1. Page images are preloaded until 76 MB of memory space is still available (imageDBLowMem)
2. Pages where precache is set to **true** are preloaded untill 64 MB of memory space is still available (pageCacheLowMemMax). The images of these pages are loaded in the RAM (into the Image DB).

When the project is ready:

1. Any new page visited is saved in the cache (RAM) with all related images until 40 MB of memory space is still available(pageCacheLowMemMin)
2. When a page change happens and space in RAM is critical (<40MB), the LRH SW HMI Runtime starts emptying the cache (RAM) removing pages and related images until 64 MB of memory space is made available. LRH SW HMI Runtime removes data stored in the cache in the following order:
    1. last visited pages and bigger and unused images (>320x240),
    2. if more memory is needed also the pages in precache and all images loaded in Image DB can be removed.

45 Functional specifications and compatibility

Here is an overview of the supported functions and related limitations. Limitations indicated here represent a safe limitation, beyond that proper operation and state-of-the-art performance of the system is not guaranteed.

Table of functions and limits

Function	Max limit
Number of pages	1.000
Number of basic widgets	2.000 x page
Number of tags	10.000
Number of dialog pages	See "HMI devices capabilities" on the facing page
Number of dialog pages that can be open at the same time	5
Number of Recipes	32
Number of parameter sets for a recipe	1.000
Number of elements per Recipe	1.000
Number of user groups	50
Number of users	500
Number of concurrent remote clients	4
Number of schedulers	30
Number of alarms	See "HMI devices capabilities" on the facing page
Number of data transfers	1000
Number of templates pages	50
Number of actions programmable per button state	32
Number of trend buffers	30
Number of tags per trend buffer	See "HMI devices capabilities" on the facing page
Memory reserved for trend buffer	See "HMI devices capabilities" on the facing page
Number of curves per trend widget	See "HMI devices capabilities" on the facing page
Number of curves per scatter diagram widget	10
Max number of trend table printable rows	10.000 on LRH SW HMI Runtime
Number of messages in a message field	1024
Number of languages	24
Number of events per buffer	See "HMI devices capabilities" on the facing page
Number of event buffers	4
JavaScript file size per page	See "HMI devices capabilities" on the facing page

Function	Max limit
Size of project on disk	See "HMI devices capabilities" below
Number of indexed instances	100
Number of indexed alias	100
Number of indexed tag sets	30
Number of physical protocols	See "HMI devices capabilities" below
Number of reports	See "HMI devices capabilities" below
Number of reports pages	32
Max number of variables in variables widget	255
User folder size (UpdatePackage.zip)	See "HMI devices capabilities" below
Number of concurrent FTP sessions	4
FTP additional folders	5

HMI devices capabilities

See "Table of functions and limits" on the previous page for the standard capabilities.

Panel	Device OS	Touch	Media Player	Media Player Portrait Mode	PDF	Max Project Size	Dialogs	Alarms	Protocols	JavaScript	Reports	Trend Buffers	Max Tags inside a Trend	Curves per Trend Widget	Max Events inside a Buffer	User Folder Size
LRHA04	Linux		na	Yes	Yes	60 MB	50	500	4	64 KB	32	25 Mb	200	5	2 K	100 MB
LRHA07	Linux		MPEG4	Yes	Yes	60 MB	50	2.000	4	64 KB	32	25 Mb	200	5	10 K	100 MB
LRHA10	Linux		MPEG4	Yes	Yes	60 MB	50	2.000	4	64 KB	32	25 Mb	200	5	10 K	100 MB

Compatibility

The following compatibility policy has been adopted:

- LRH SW version must always be aligned with LRH SW HMI Runtime on the device,
- the user is responsible for updating LRH SW HMI Runtime components on the HMI device at any LRH SW update,
- the LRH SW HMI Runtime update can be done directly from LRH SW using the Update Target command available in the Run\Manage Target dialog,
- projects created in a LRH SW version no older than V1.00 (00) can be opened and handled by any newer version,
- projects created with older versions of LRH SW, opened with later versions and deployed to compatible LRH SW HMI Runtime, are ensured to maintain the performance and functionality,
- compatibility between newer versions of LRH SW HMI Runtime and projects created and deployed with older versions of LRH SW is not ensured.

Important: Do not edit projects with a version of LRH SW older than the one used to create them. It can result in a damage of the project and to LRH SW HMI Runtime instability.

Converting projects between different HMI devices

Project conversion from different HMI device models is supported, however, some manual operations may be required if the project uses features not supported in the destination device.

Guideline

Before converting a project have a look if some unsupported features are present (see "HMI devices capabilities" on the previous page), and adjust your project by removing the unsupported features before converting the project.

In particular:

- Verify limitations and features not supported by the new HMI device (see "Table of functions and limits" on page 540 for details).
- Remove unsupported widgets, actions, system variables, protocols, project properties.
- If the project uses external storage, verify if the same storage path is still available.
- Adjust OS-specific external applications or paths.
- If necessary, reduce project size according to the new HMI device type limitations (see "Limitations" for details).
- Since HMI devices are based on different hardware platforms with different CPU speed, RAM memory size, cache size, make sure to check project boot time and page loading time for each page in the project.
- Verify JavaScript code for OS-specific operations.

46 System Settings

System Settings is an internal tool of the HMI device that can be used for the basic device settings or for the system components update.

Note: the system components can be update even from the LRH SW (see "Updating system components in HMI devices" on page 569 for details)

CAUTION: Working with the System Settings tool is a critical operation and, when not performed correctly, may cause product damages requiring service of the product. Contact technical support for assistance.

Linux Devices

Linux products offer a powerful integrated tool called System Settings that allows management and upgrade of system components. Operations can be done directly on HMI or remotely using web browser.

Runtime Installation

HMI devices are delivered from factory without Runtime, at first power up HMI shows the “Runtime Loader” screen.

Runtime can be installed:

- Automatically, via Ethernet on first project download with LRH SW
- Manually via USB Memory, creating an “Update Package”. (See the "Update package" on page 95 to create a runtime package)

Install Runtime via Ethernet

To install Runtime via Ethernet follow the "Download to HMI device" on page 92 procedure.

WARNING: Runtime installation via Ethernet download requires the HMI to have a valid IP address.

The IP address can be assigned in three ways:

- *Automatically via DHCP server.* If a DHCP server is available on the network IP address will be assigned automatically by the server.
- *Automatically via Auto-IP feature.* If DHCP assignment is enabled but no DHCP server is available on the network the HMI assigns itself an IP Address into range 169.254.x.x with subnet mask 255.255.0.0
- *Manually via System Settings.* From System Settings menu, in Network section the IP address can be manually assigned, disabling the DHCP server assignment feature.

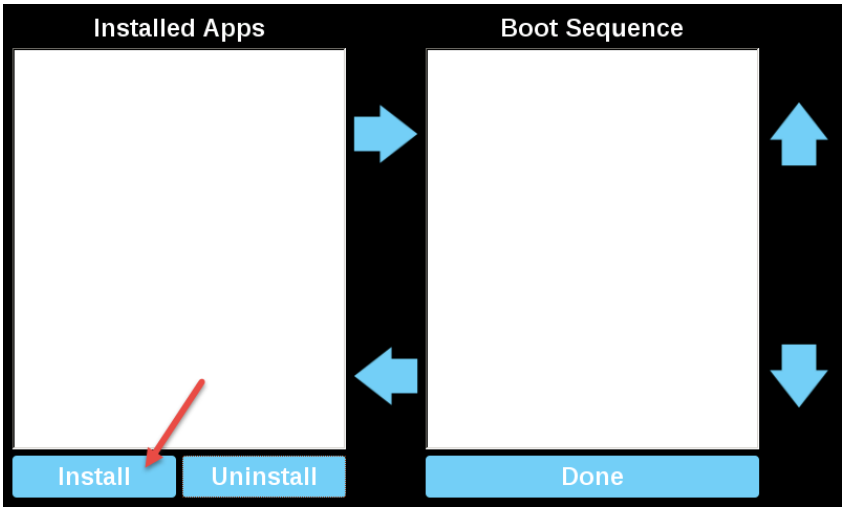
Install Runtime via USB Memory

To install Runtime, UpdatePackage or Backup Package via USB device follow this procedure:

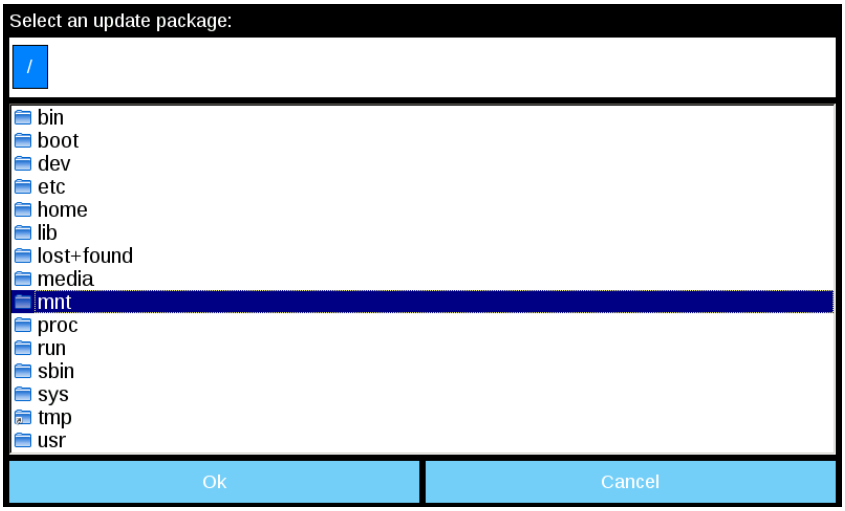
1. Create an Update Package from LRH SW and copy into an empty USB memory stick

Note: File systems supported are FAT16/32 and Linux Ext2, Ext3 and Ext4.

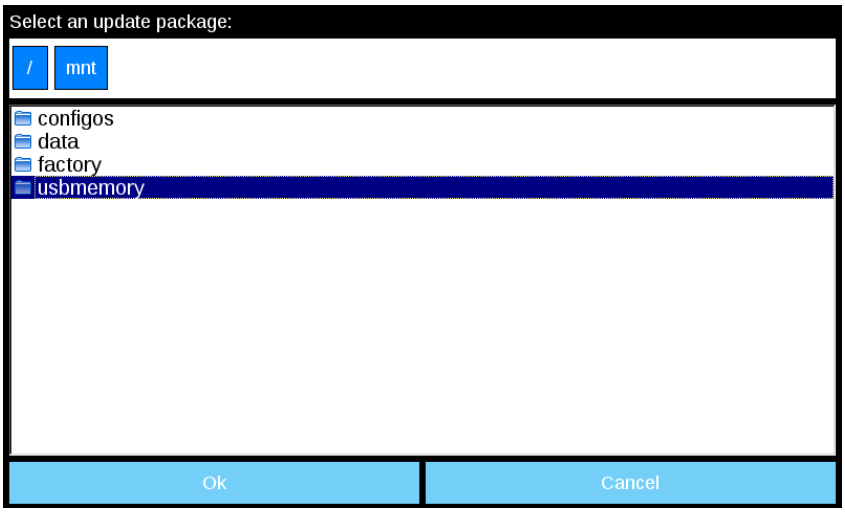
2. On HMI select [Startup sequence], then [Install]



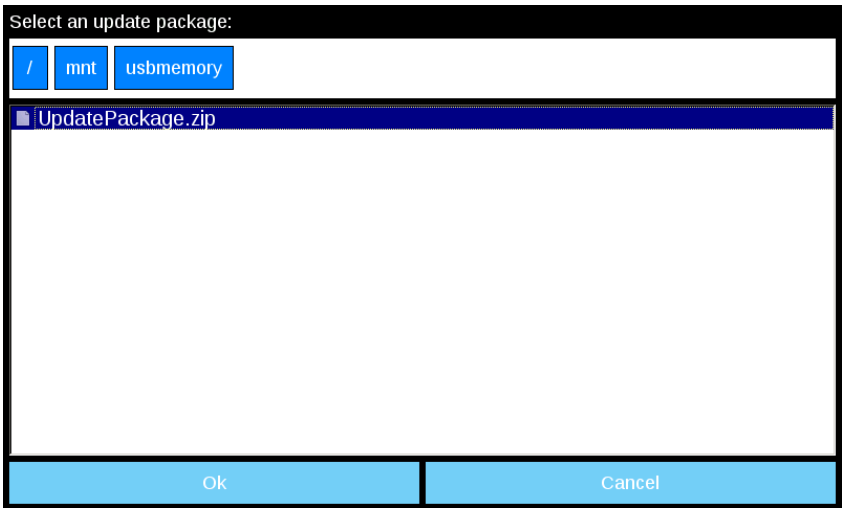
3. Double click on “mnt” to access this folder



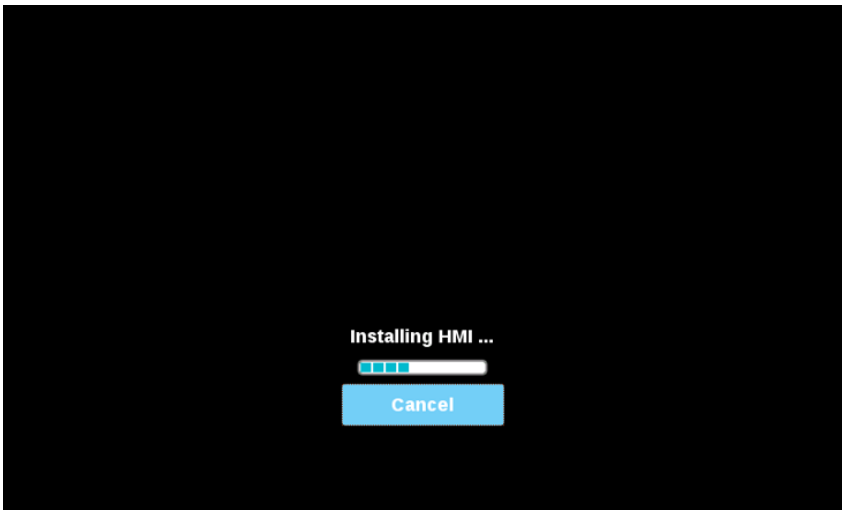
4. Then on “usbmemory”



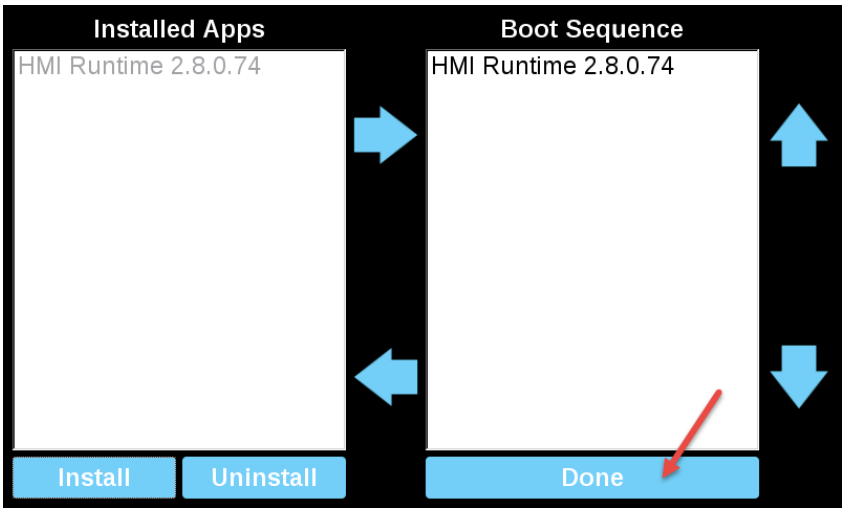
5. Select “UpdatePackage.zip” and confirm with [Ok]



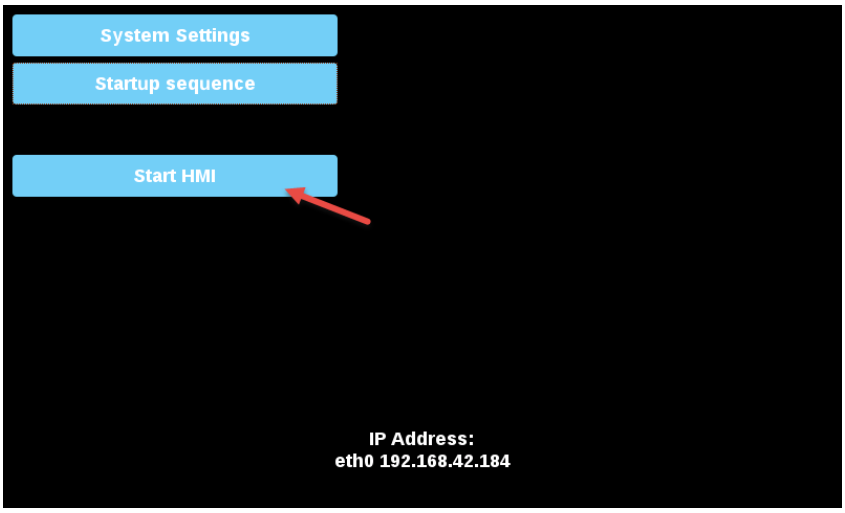
6. The runtime installation begin



7. At the end press "Done" button



8. Then "Start HMI" button



System Settings

The user interface of System Settings is based on HTML pages and can be accessed both locally on the HMI device screen and remotely using a Web browser.

Administrator username with full access right is "admin" with default password "admin". Generic username is “user” with default password “user”

WARNING: For security reasons, change the default passwords for both usernames (passwords can be modified from the “System Settings -> Authentication” command)

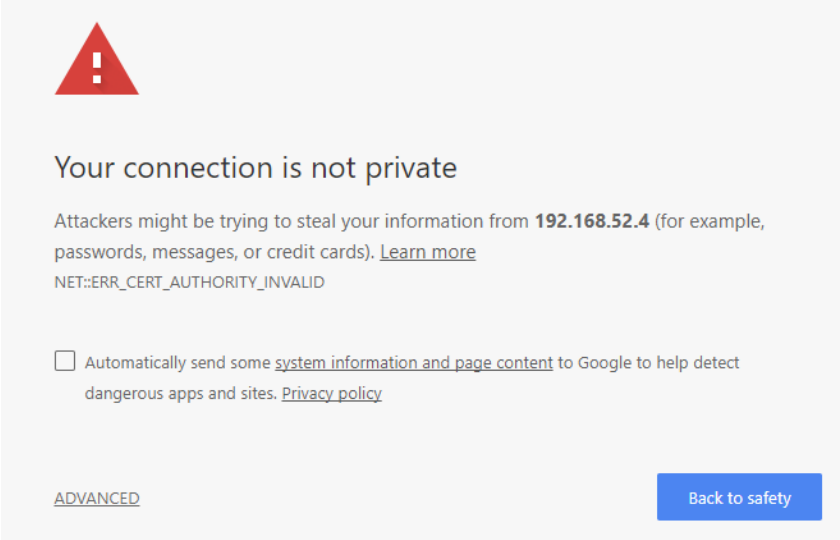
Accessing at the system settings from the HMI device do not require to enter a password until the default "admin" password is not changed.

System Setting access from Web browser

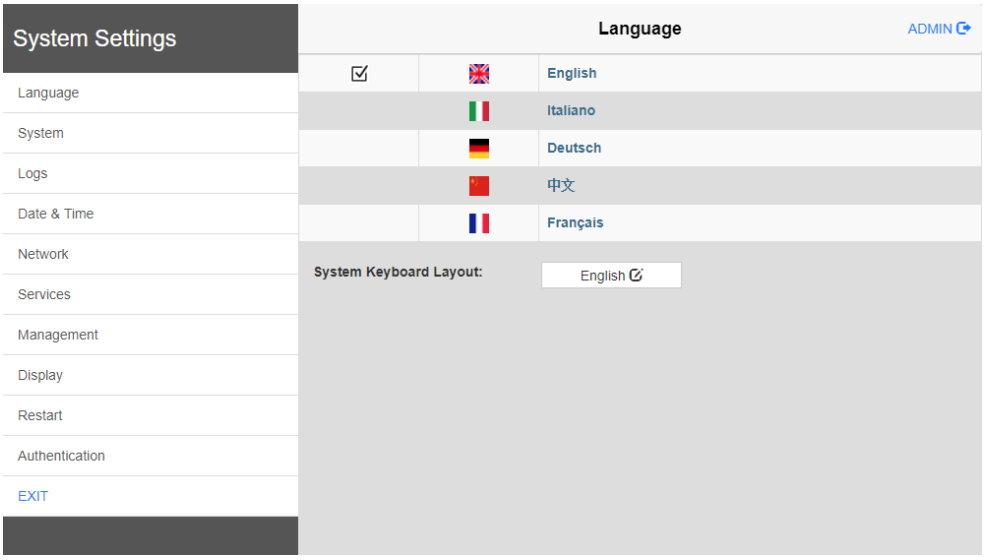
To access System Settings using a Web browser, enter the IP address of the device, in the following format:

https://IP/machine_config

Note the remote access use encrypted https protocol on port 443. When the connection is established, the HMI device send a certificate to use for the encryption. Since the certificate is not signed from a Certificate Authority you will get a warning message. Please, click on advanced options and choice to proceeding.



Browse through the options available in the menu on the left: the active item is highlighted and related information is displayed on the right.



Default security protocols proposed by the HTTPS server in the Linux HMI device are:

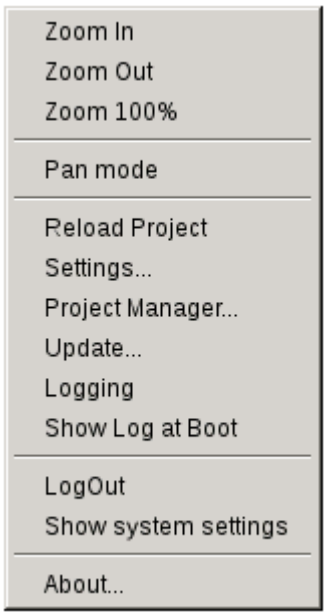
- SSLv3 256 bits ECDHE-RSA-AES256-SHA
- TLSv1 256 bits ECDHE-RSA-AES256-SHA

WARNING: We discourage usage of CBC cyber suites in the context of SSL3 or TLSv1.0 connections since potentially affected by some vulnerabilities.

System Setting access from HMI device

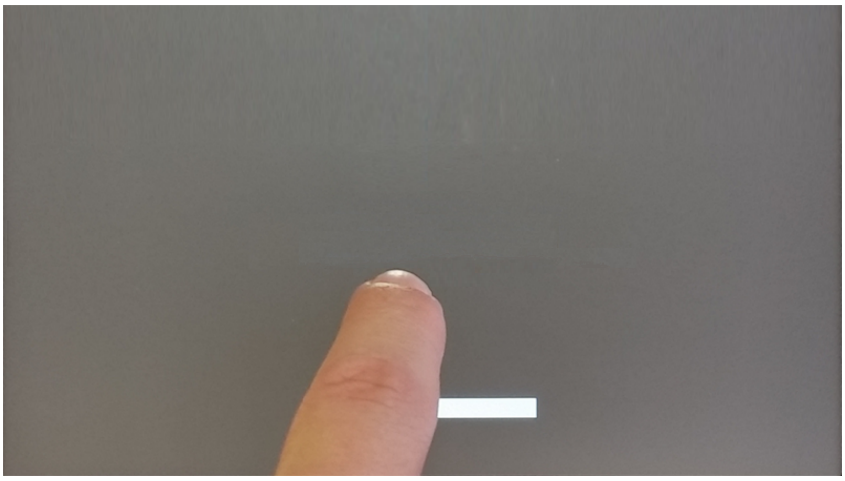
When Runtime is not installed, the System Settings is accessible from the Runtime Loader screen,

When Runtime is installed the System Settings is accessible selecting “Show System Settings” option of Context Menu,



Enter System Settings via tap-tap procedure

Tap-tap consists in a sequence of several touch activations by simple means of the finger tapping the touch screen performed during the power-up phase and started immediately after the HMI is powered on.



When “tap-tap detected” message appears on the top of the screen. Wait for 5 seconds (without touching the screen) to enter System Settings sub menu

Wait for 5 more seconds (without touching the screen) to enter Default Mode

Select "System Setting" from the HMI Default Mode screen

System Settings Sections

To change system settings values, enter in edit mode by click the edit button on the right top.

The edit button is available only inside the dialogs that contains modifiable parameters.

Languages

Select the language for the system settings interface

- System keyboard layout: select the layout of the virtual keyboard

System

Parameter	Description
Info	Device information
Status	Device status (Free RAM, Up time, CPU Load)
Timers	Device timers (System on, Back light on)
PlugIn	Hardware plugins information

Logs

Set the persistent log option if you want maintain the log files saved after a power reset.

Use save button to export a copy of the log files.

The log files manager cyclically fill 3 files of 4Mb

Date & Time

Device date and time.

Parameter	Description
Current Timezone	Timezone region
Current Date **Local Time**	Date and Time can set manually only when the Automatic Update is disabled.
Automatic Update (NTP)	Enable to keep date and time synchronized from a remote server • NTP Server Specify the Internet NTP Server address The NTP Client of the HMI Device is a complete implementation of the Network Time Protocol (NTP) version 4, but also retains compatibility with version 3, as defined by RFC-1305, and version 1 and 2, as defined by RFC-1059 and RFC-1119, respectively. The poll process sends NTP packets at intervals determined by the clock discipline algorithm. The process is designed to provide a sufficient update rate to maximize accuracy while minimizing network overhead. The process is designed to operate in a changeable mode between 8 sec and 36 hr.
Accept NTP requests	When enabled the device will accepts NTP requests from outside. When automatic update is not enabled the device will share the local RTC clock time.

Networks

Network parameters. Available parameter in edit mode:

Parameter	Description
General Settings	Device hostname Avahi Hostname (see "Avahi Daemon" on the facing page)
Network Interface	Network parameters of the available interfaces • DHCP • IP Address • Net Mask • Gateway
DNS	DNS Servers Generally provided from the DHCP servers, but can be modified in edit mode Search Domains Optional domains that will be used in concatenation with the provided urls

Services

Services are available only when logged as admin.

Mouse click on the enable button to enable/disable the service. Click the service name to list the associate parameters.

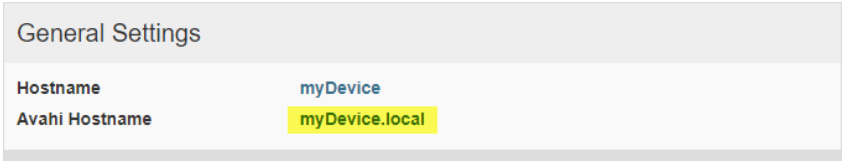
Autorun scripts from external storage

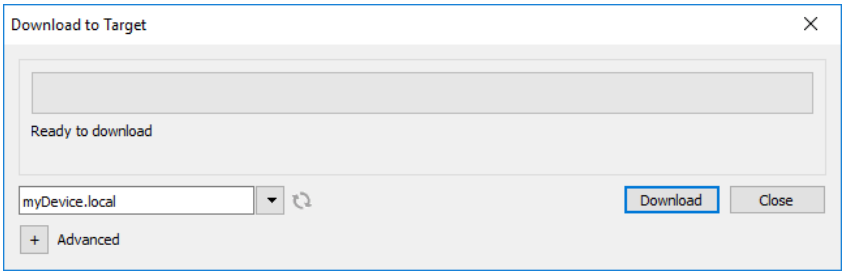
Enable/Disable the possibility to run the "autoexec.sh" script file when a USB key is plugged into the device. Disable this service if you want to prevent unauthorized access through the USB interface.

Required BSP v1.0.212 or greater

Avahi Daemon

Avahi is a system which enables programs to publish and discover services and hosts running on a local network. When it is enabled, the HMI device can be reached even using the device's host name (in alternative to the IP Address).





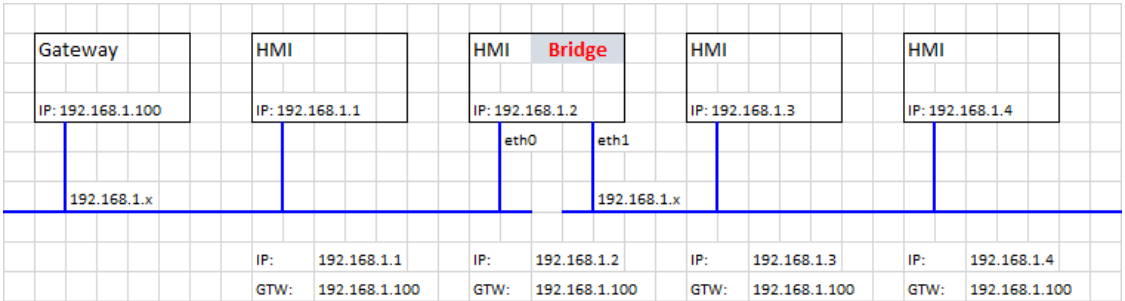
Avahi Daemon runs on UDP port 5353

On Linux and Apple PCs, the Avahi service comes for free with the OS. On Windows PCs instead, you need to install an Avahi service to be able to reach the panel by his Avahi host name (e.g. you need to install the Apple Bonjour application - Bonjour is a trademark of Apple inc.).

Bridge/Switch Service

Using the bridge service is possible connect together the WAN (eth0) network adapter with the other network interfaces. When used, the two Ethernet interfaces are bridged and both Ethernet interfaces are sharing the same IP address.

Bridge Service creates a Linux-based layer-2 Network Bridge between two or more network interfaces. If both WAN and endpoint devices are attached to such bridge, the two networks will be physically joined and endpoints will be available as if they were directly connected to the WAN



DHCP Server

Provide the DHCP Server on the selected interfaces.

Parameter	Description
Enabled	Enable the DHCP Server on the selected interface
Start IP **Stop IP**	IP addresses distributed from the DHCP Server
Gateway	The gateway address
Netmask	The provided netmask
DNS Server	The DNS server address
Lease Time (seconds)	Lease time, default is 86400s (1 day) Acceptable values are from 60s to 864000s (10 days)

Fast Boot

When fast boot is enabled, at the power up the HMI device will start the HMI application as fast as possible. In this mode, there are not showed diagnostic information (e.g. the loading bar) but only the minimum necessary features are loaded before loading the User Interface (e.g. System Settings, VNC, SSH, etc. will be load after loading the HMI application).

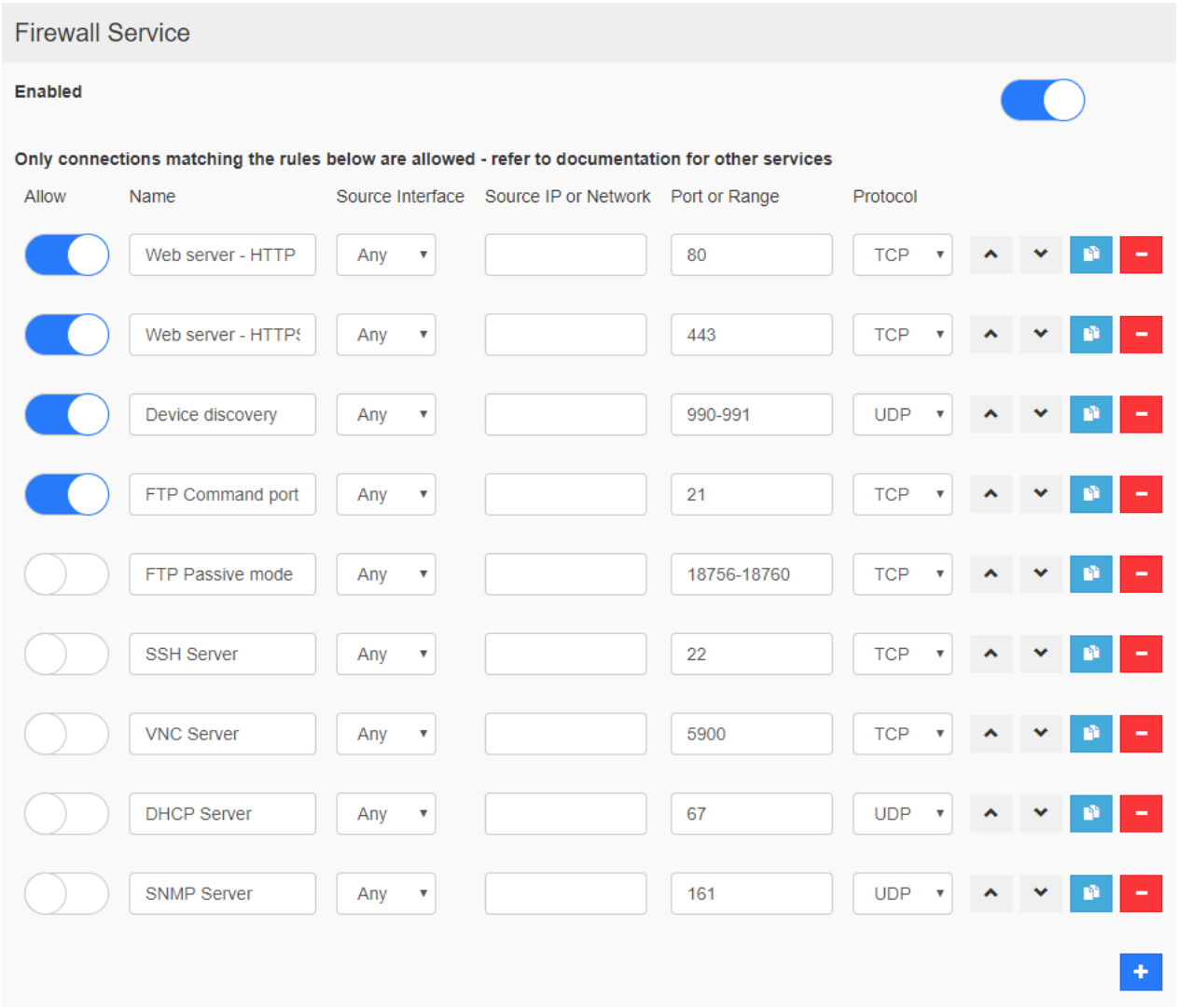
To obtain best performance, in addition of enabling the fast boot mode, it is recommended to:

- disable any service that is not necessary
- avoid keeping enabled the persistent log
- use static IP address instead of DHCP service

Required BSP v1.0.242 or greater

Firewall Service

When the firewall is enabled, only connections matching the defined rules are allowed. Note that some rules must be enabled for the HMI can to work properly.



Notes:

- The firewall is based on IP tables which operates only at layer 3 (layer 2 packets won't be filtered, e.g. ARP)
- Only INPUT and FORWARD packets are filtered, not OUTPUT
- PING/ICMP echo reply packets are always allowed
- Internet sharing scenarios (e.g. 3g or wifi connection to endpoints) are not supported
- Packets filtered by the firewall are dropped

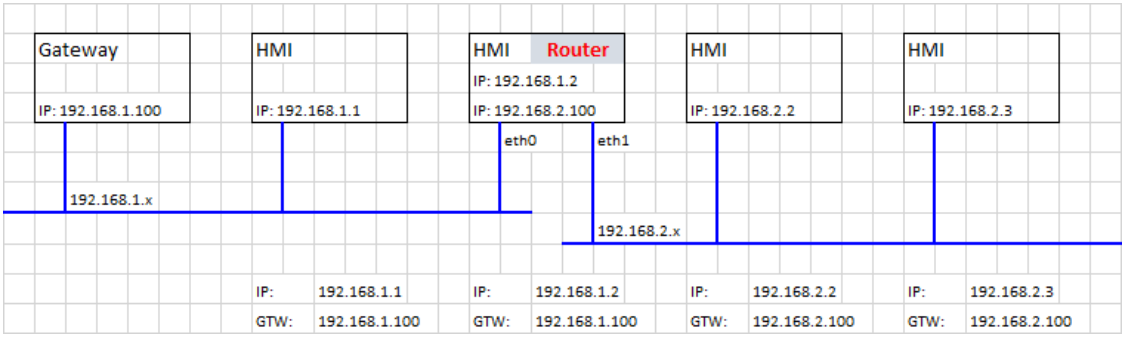
If you enable the Firewall and you need to use the FTP passive mode with LRH SW HMI Runtime older than version 2.10.0.280 then you need to open the ports 1024-2048/tcp and 16384-17407/tcp. From version 2.10.0.280 instead, LRH SW HMI Runtime uses the ports 18756-18760/tcp that are proposed into Firewall settings by default.

Firewall is available from BSP v1.0.532
If you are updating from an old BSP version and you don't see the default rules, you have to reset the system settings (see "Update System Components" on page 560).

Router Service

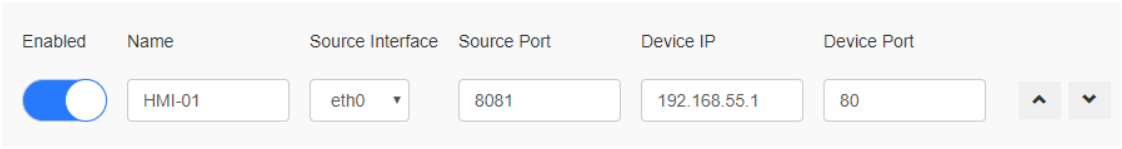
This service uses IP Forwarding and Network Address Translation to share the connection from WAN (eth0) towards LAN (eth1 or eth2): connected endpoints may reach the same networks reachable by the gateway (including Internet if

available).



Port Forwarding

Port forwarding redirects incoming TCP packets requests from WLAN interface from one address and port number combination to another combination of address and port number.

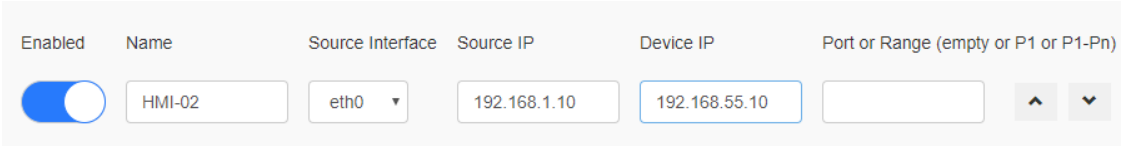


Available from BSP v1.0.507

1:1 NAT

1:1 NAT, create alias IP on WLAN and forward all packets (or given port range) with that destination IP to another device attached to a LAN

Available from BSP v1.0.507



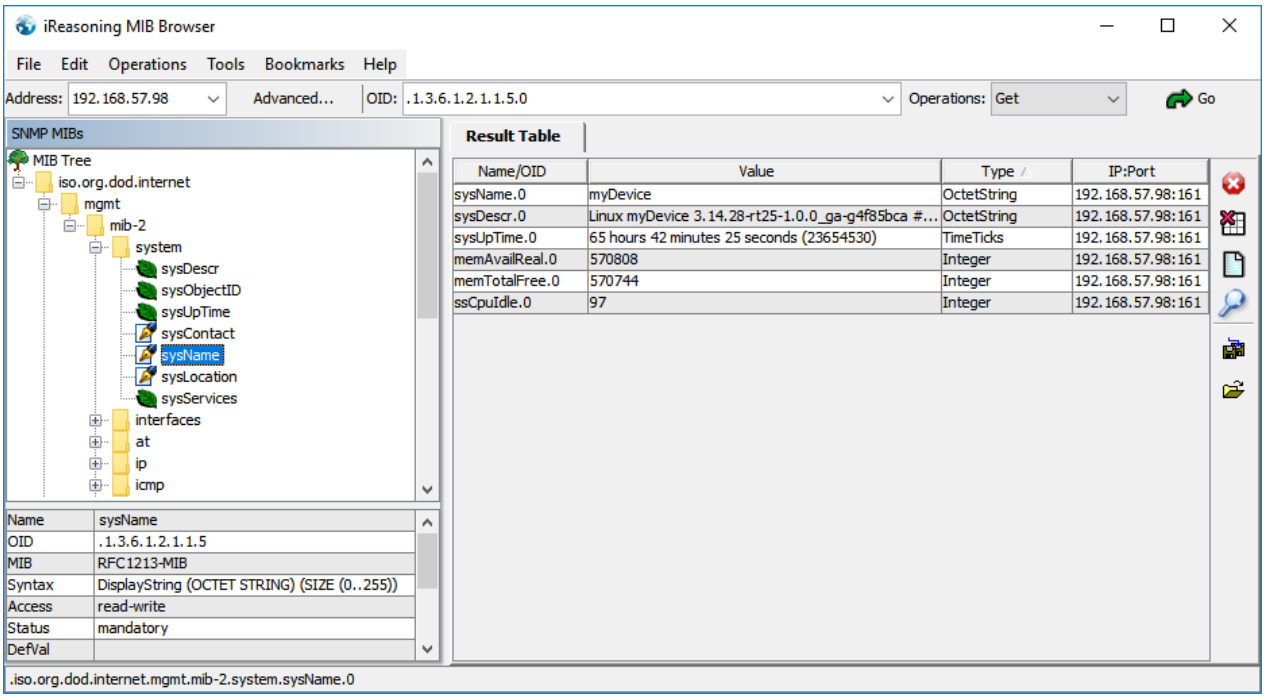
Show loading bar during boot

Enable/Disable the display of the loading bar during the boot phase.

SNMP Server

SNMP is a network protocol that allow to manage network infrastructures. It is commonly used to monitor network devices as switches, routers, etc. connected to a LAN network.

When the SNMP service is enabled, an SNMP Manager can retrieve information from the HMI device using the SNMP protocol. Currently, there are not proprietary MIBs available. Only the standard public community MIBs are available in read only mode.



Example:

System Name:	.1.3.6.1.2.1.1.5.0
System Description:	.1.3.6.1.2.1.1.1.0
System UpTime:	.1.3.6.1.2.1.1.3.0
Total RAM used:	.1.3.6.1.4.1.2021.4.6.0
Total RAM Free:	.1.3.6.1.4.1.2021.4.11.0
Idle CPU time (%):	.1.3.6.1.4.1.2021.11.11.0

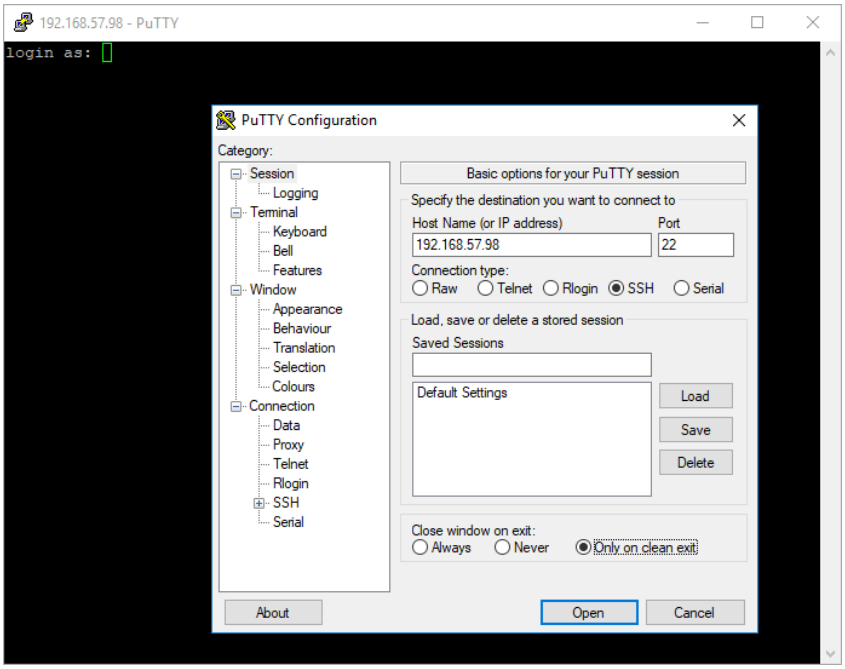
SNMP Server runs on UDP port 161

This service is designed to be used during the development phase. For security reasons, remember to disable the service before switch to production.

SSH Server

SSH service has been designed only for advanced users. It provides remote login to HMI device using the secure shell protocol. On PC you can run a SSH Client as, for example, PuTTY that is an open source software distributed under the MIT license.

The default password for the username the admin is “admin”. See the "Authentication" on page 559 chapter to additional information.



SSH Server runs on TCP port 22

This service is designed to be used during the development phase. For security reasons, remember to disable the service before switch to production.

VNC Service

VNC is a service that allows remote access to the display of the HMI device. VNC clients can be used to get the remote control of the HMI device.

VNC should be disabled after use and autostart is not recommended.

- Port: VNC Server listens for connections on TCP port 5900 (default)
- Multiple clients: allow multiple sessions on the same port (if disabled, previously logged clients are disconnected upon a new incoming connection)
- View only: do not allow active user interactions (clients can only watch)
- Encryption: activate SSL encryption of connections (not widely supported - check client compatibility)
- Authentication: whether users are authenticated upon session creation. A custom VNC specific password can be set or system passwords can be used (this option is only available if also Encryption is enabled)

Plugins

This page will show the parameters available to configure the optional plugins modules attached to the HMI device. See the description of the each plug-in module to additional information.

Management

Management is available only when logged as admin.

From the management area is possible "Update System Components" on the next page of the HMI device.

CAUTION: Working in the Management area is a critical operation and, when not performed correctly, may cause product damages requiring service of the product. Contact technical support for assistance.

Use the “Clear” command inside the “Data” section to remove HMI Runtime from the device (Factory Restore)

Display

Parameter	Description
Brightness	Brightness level of the display
Back light timeout	Backlight inactivity timeout
Orientation	Display orientation

Restart

HMI device restart command

Authentication

Enter in edit mode to change the authentication passwords.

Administrator username with full access right is "admin" with default password "admin". Generic username is “user” with default password “user”

WARNING: For security reasons, change the default passwords for both usernames

x.509 Certificate

HMI Device use a self-certificate to encrypt the Internet communication trough the HTTPS protocol. You can personalize the certificate with the data of your Company and ask to a Certificate Authority to firm it.

The procedure to personalize and firm your certificate is:

1. Enter in edit mode and fill the necessary parameters, then push GENERATE button to generate a self-signed certificate with your data.
2. Export the “Certificate Signed Request”
3. Sent the “Certificate Signed Request” to a Certificate Authority to firm it (general this is a paid service)
4. Import the signed certificate into the HMI device

Certificate's parameters

Parameter	Description
Device Name	The name of your device
Organization	The legal name of your organization
Unit	The division of your organization handling the certificate

Parameter	Description
State	The state/region where your organization is located
Location	The city where your organization is located
Country	The two-letter ISO code for the country where your organization is location
Valid (days)	Validity of the certificate
Key Length	Number of bits of the key used from the cryptographic algorithm

Managed certificates are base64 encoding

Required BSP v1.0.239 or greater

EXIT

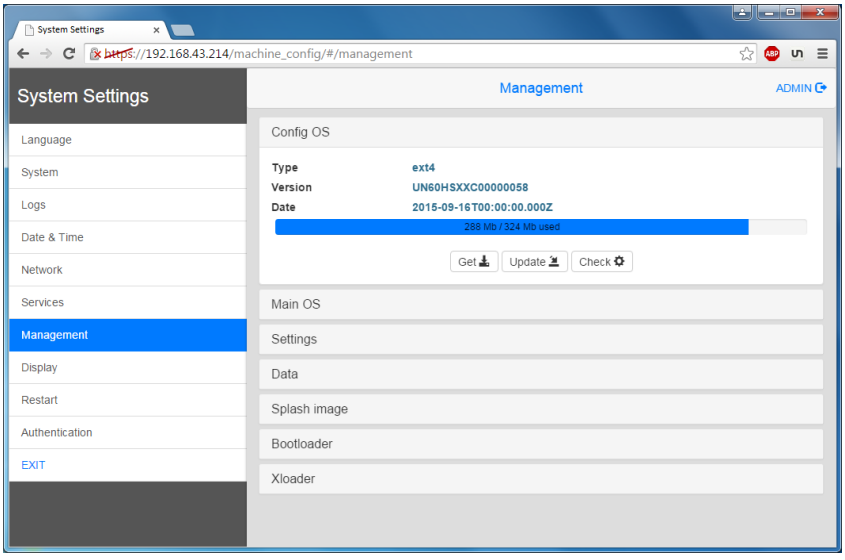
Exit from the System Setting tool.

Update System Components

CAUTION: Working in the Management area is a critical operation and, when not performed correctly, may cause product damages requiring service of the product. Contact technical support for assistance (the latest BSP files will provided from tech support).

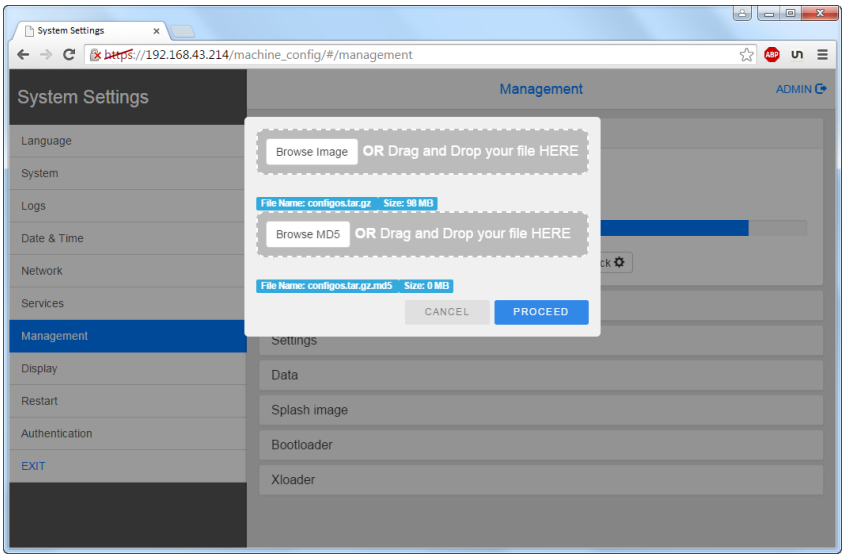
The system components of the Linux device can update locally using an USB memory key or remotely via web browser.

To update system components enter System Settings in Config OS mode via tap-tap procedure on HMI or open web browser to https://<HMI-IP-address>/machine_config and select the “Management” section.



Expand the component to update and select [Update]

On the opened dialog, click [Browse Image], then select the “xxx-mainos-xxx.tar.gz” file. Click then on [Browse MD5] and select the “xxx-mainos-xxx.tar.gz.md5” file.



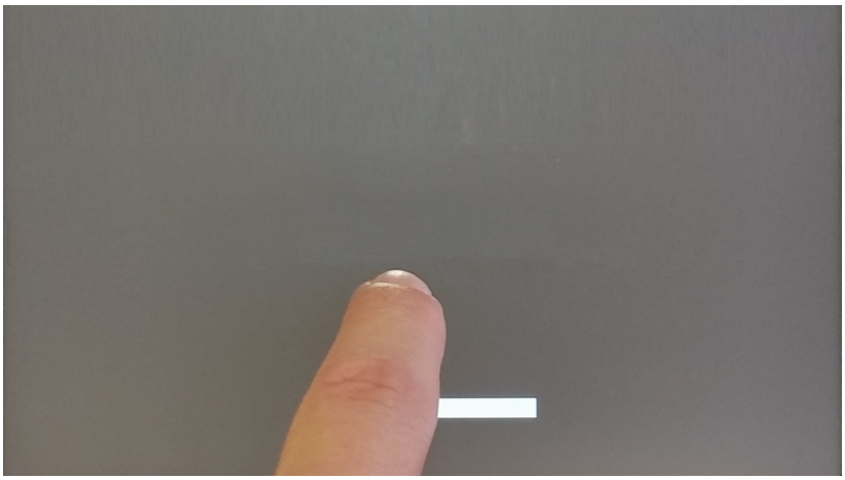
Important: Do not turn off the device while a system component is being upgraded.

At the end of the component update, restart HMI and leave it starting normally.

Enter System Settings in Config OS mode via tap-tap procedure

System Setting in Config OS mode is available via tap-tap sequence, this mode can be accessed also when HMI is facing a software failure.

Tap-tap consist in a sequence of several touch activations by simple means of the finger tapping the touch screen performed during the power-up phase and started immediately after the HMI is powered on.



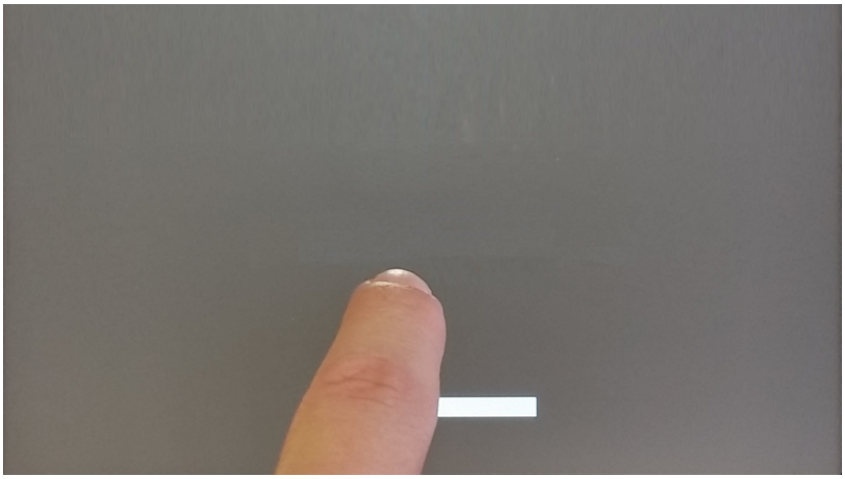
When “tap-tap detected” message appears on the top of the screen, press and hold the finger on touchscreen, to select “Restart: Config OS”

HMI will restart into System Settings in Config OS mode:

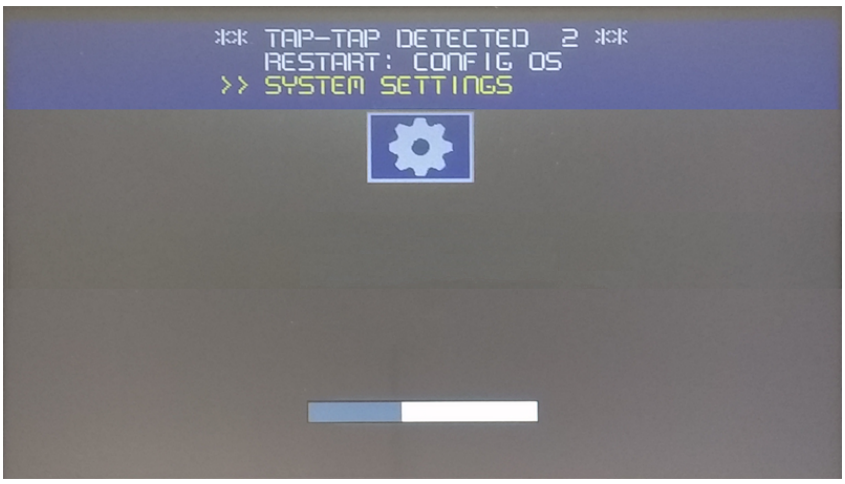
Touchscreen calibration

System Setting Calibration allows to calibrate Touchscreen device, can be accessed by tap-tap procedure.

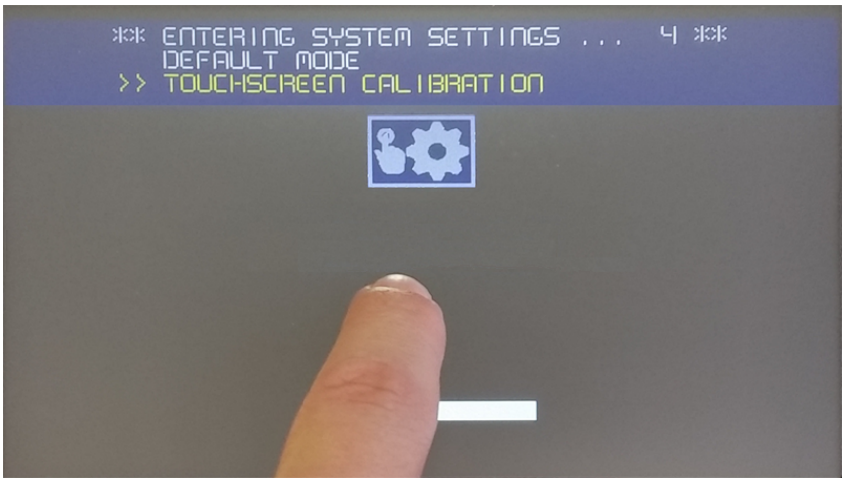
Tap-tap consists in a sequence of several touch activations by simple means of the finger tapping the touch screen performed during the power-up phase and started immediately after the HMI is powered on.



When “tap-tap detected” message appears on the top of the screen, wait for 5 seconds (without touching the screen) to enter System Settings sub menu



Press on touch screen, “Touchscreen calibration” voice will be highlighted in yellow, hold pressed for few seconds until touchscreen calibration procedure starts

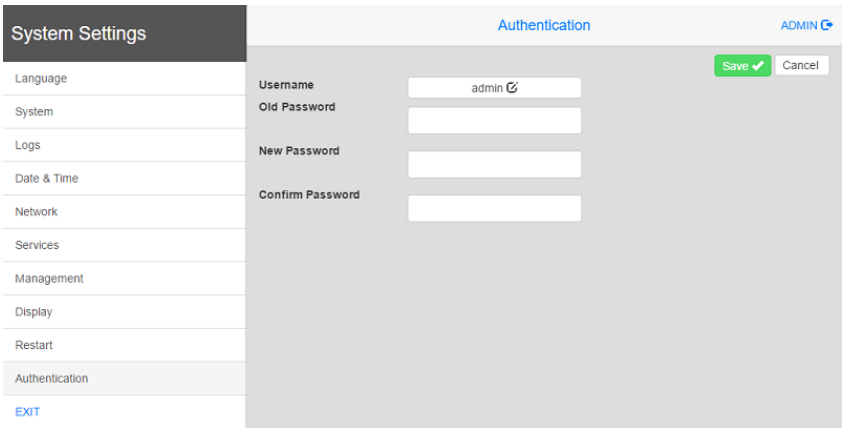


Follow the instructions on screen to complete the calibration procedure, system will prompt to touch specific points to calibrate the touchscreen device.

Password protection

Internal password of the HMI device.

From the Authentication tab, inside the "System Settings" on page 547, activate the edit mode and select the username to change the associated password.



Password for admin user can modified even from the context menu of theLRH SW HMI Runtime (see "Context menu options" on page 8 for details) and from the update package (see "Update package" on page 95for details).

Accessing at the system settings from the HMI device do not require to enter a password until the default "admin" password is not changed.

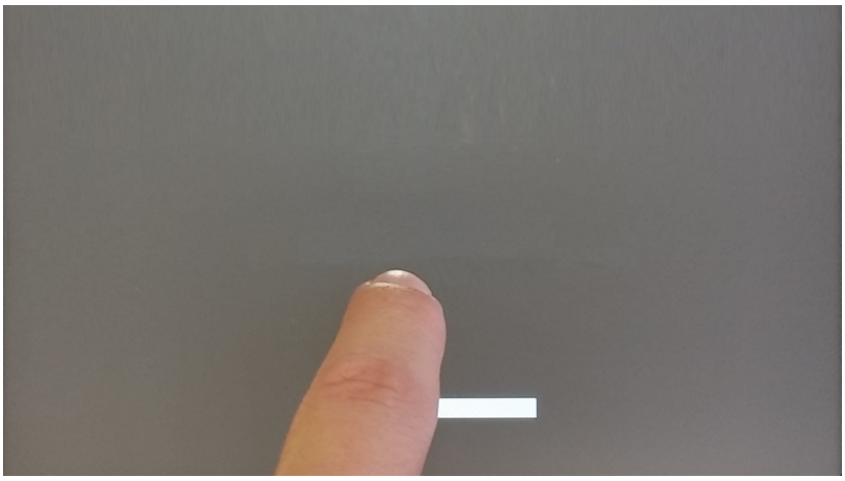
There are two usernames:

- Administrator username with full access right is "**admin**" with default password "admin".
- Generic username is “**user**” with default password “user”

Factory restore

System Settings in Default mode allows to uninstall HMI Runtime or change Startup sequence, this mode is available via tap-tap sequence and can be accessed also when HMI is facing a software failure.

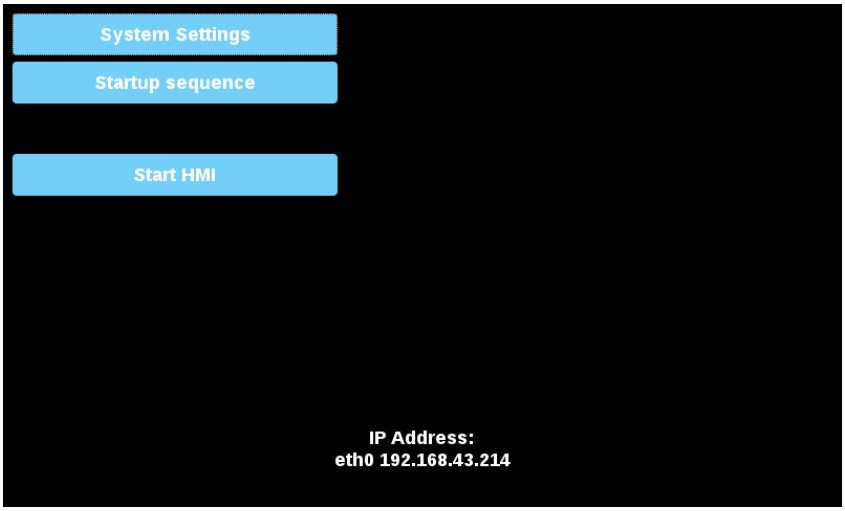
Tap-tap consists in a sequence of several touch activations by simple means of the finger tapping the touch screen performed during the power-up phase and started immediately after the HMI is powered on.



When “tap-tap detected” message appears on the top of the screen. Wait for 5 seconds (without touching the screen) to enter System Settings sub menu

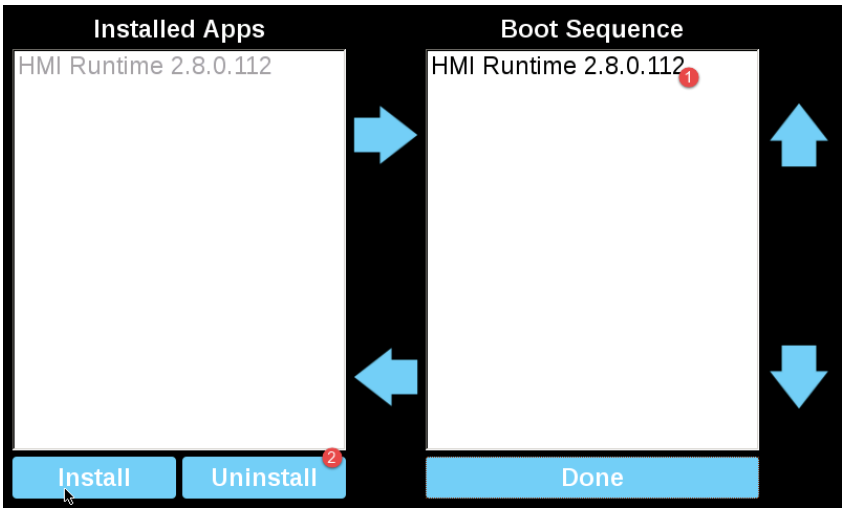
Wait for 5 more seconds (without touching the screen) to enter Default Mode

To uninstall the Runtime from HMI in Default Mode screen select [Startup Sequence]:

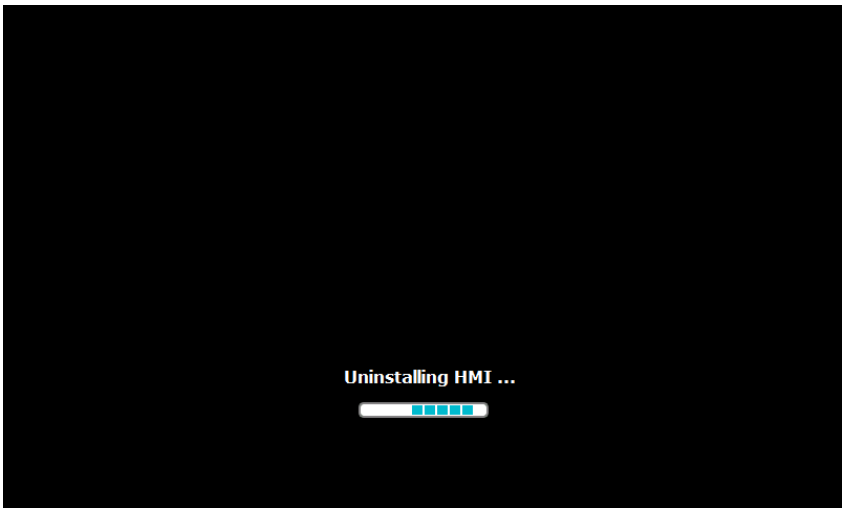


From the installed applications view:

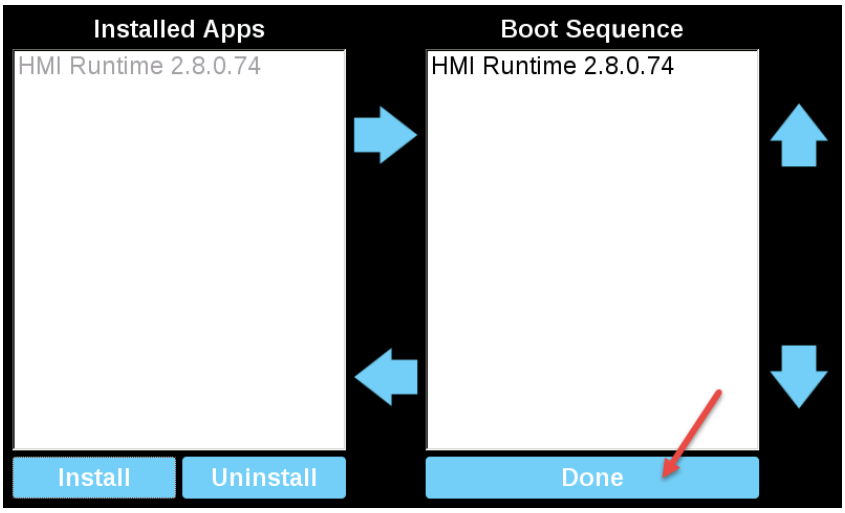
1. Select the Runtime you want to remove
2. Uninstall the selected Runtime



Runtime uninstall process will be performed:



At the end press "Done" button



Configuration mode

In the case that it is not even possible to boot the device, there is a special procedure to recovery the device by booting it in a special mode called configuration mode. From this mode you can open the device management dialog from where you can delete user data, restore system setting or update the firmware of the device.

To boot the device in configuration mode choice one of the below procedures

- Power on the device and immediately power off when splash screen appear on the screen (if you cannot see the splash screen, power off the device when you heart the beep-beep). Repeat this procedure for three time then power on again the device and wait the configuration mode appears.
- Create a special file named “$0030D8$.bin” and put it inside an empty SD card. Insert the SD card into the device and power on the device. Device will start in configuration mode.

47 Updating system components in HMI devices

Most of the system software components can be easily upgraded ensuring a high degree of flexibility in providing updates and fixes to existing and running systems.

New software modules can be updated

- Directly on HMI device using an USB flash drives (see "System Settings" on page 543 for details)
- From LRH SW application (see "Update of system components from the application" on the next page for details)

Each HMI device is labeled with a product code including all factory settings (hardware, software and firmware components). Refer to this label for information on your HMI device. The HMI device update tool also provides detail on the components actually running on the device.

CAUTION: Make sure you use the correct upgrade files, since loading upgrade files unsuitable for your device will cause serious system malfunction. Always check your device product code.

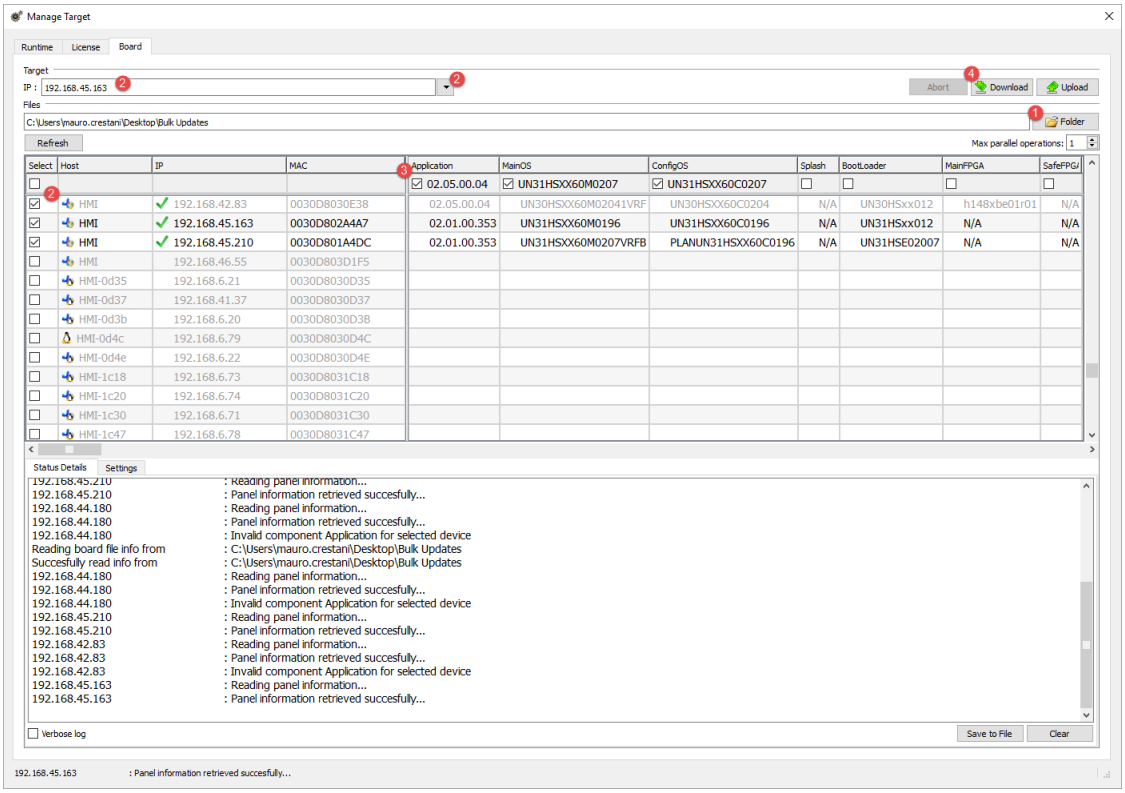
Note: Upgrade files are distributed upon request as a part of technical support activity.

Service call: Downgrade operations are complex tasks which might cause serious damage to your equipment if not performed correctly. These operations are reserved to technical support.

Update of system components from the application

You can download system components to a single HMI device or to a bulk of HMI devices of the same type using the Ethernet communication interface.

Path: ***Run> Manage Target> Board***



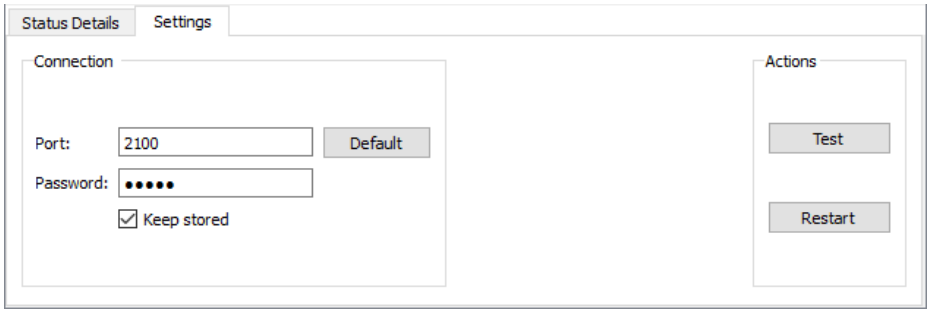
1. Select the folder that contains the files to download to the HMI device or where to upload files from the HMI device
2. Select one or more HMI device.
3. Select the components that you will download (or upload) to/from the devices
4. Start the Download to HMI or the Upload from the HMI operation

Note:

- The tool is designed to update multiple HMI devices of the same type. Please avoid putting files for different device type into the same folder
- If the desired target IP is not listed, type it directly into the box. The discovery service is a broadcast service. When a remote connection is done via VPN or from external networks, it will not work and you will have to enter the address manually.
- Download of the selected components will be performed only to the compatible devices
- Based on your network and hardware capabilities you can increase the number of devices to update in parallel
- You need to restart the HMI device to finalize the update.

Settings

From the **Settings** tab you can specify the Port and the Password parameters to use to communicate with the HMI devices. Leave Password empty if no password is set on the HMI device side.



WARNING: Bulk mode is working only with the HMI devices that have the same connection parameters

Uploading a splash screen picture

You can replace the default splash screen image shown by the devices during the power up phase.

The image used as splash screen must comply with the following requirements:

Filename	splash.bmp
Format	Bitmap, RGB 565 format
Size	< 500 KB
Bitmap width	Even number (for example 430x239)

To upload the splash screen image:

1. Rename the new image splash.bmp and copy it in the source folder.
2. Select HMI devices
3. Click **Download**.

To ensure the best visual results, splash screen images must have a black background.