PAGE LAB

User Manual

Revision 4.0 - April 2013

AXEL

SOFTWARE SOLUTIONS FOR THE INDUSTRIAL AUTOMATION

PageLab User Manual
Revision 4.0 - 2013-04-16
Published by Axel S.r.l.
Via del Cannino, 3
21020 Crosio della Valle

Contents

1. Overview 1

1.1 Main elements 1

1.2 Run-time functionalities 4

1.3 Communicating with the target 4

2. Creating a simple PageLab project 5

2.1 Purpose of this chapter 5

2.2 Creating a new project 5

2.3 Inserting the first page in the project 6

2.3.1 Creating a new page 6

2.3.2 Editing the colors of the page 7

2.4 Inserting a secondary page 8

2.4.1 Creating a secondary page 8

2.4.2 Dimensioning and setting the secondary page 8

2.4.3 Viewing the title bar and the system button 9

2.4.4 Assigning a style to the window 10

2.4.5 Choosing the start window 11

2.5 Inserting static controls 11

2.5.1 Inserting a line 12

2.5.2 Inserting a rectangle in the page 12

2.6 Inserting static images 14

2.6.1 Importing a bitmap in the project 14

2.6.2 Associating an imported bitmap with an image control 16

2.7 Text strings 17

2.7.1 Inserting a text string 17

2.8 Data management in PageLab 17

2.8.1 Declaring a local variable 17

2.8.2 Declaring a global variable 19

2.8.3 Importing the PLC variables in the PageLab project 19

2.8.4 Inserting field parameters 20

2.9 Inserting edit box 21

2.9.1 Inserting an edit box in the page 21

2.9.2 Edit box and PageLab local variable association 24

2.9.3 Edit box and PageLab global variable association 25

2.9.4 Linking an edit box with a target (or system) variable 25

2.9.5 Linking an edit box with a PLC LogicLab variable 26

2.9.6 Linking an edit box to a parameter 26

2.9.7 Linking an edit box to a variable by dragging and dropping 27

2.10 Inserting buttons 27
2.10.1 Inserting a led-button 27
2.10.2 Inserting a boolean variable command button 28
2.10.3 Inserting a button to open a child page 29
2.10.4 Inserting a button aimed at launching a procedure of the user 31
2.11 Visibility and updating of controls 32
2.11.1 The visibility property 33
2.11.2 The refresh property 34
2.12 Compiling and downloading the project on the target 35
2.12.1 Connecting to the target 35
2.12.2 Compiling pages for the target 36
2.12.3 Downloading and executing the compiled pages on the target 36
2.12.4 Simulation 36
3. PageLab layout 37
3.1 Project window 37
3.2 Embedded editors 37
3.3 Properties window 38
3.4 Toolbars 38
3.5 The output window 38
3.6 Target variables and parameters 38
3.7 Table of keys-actions associations 39
4. HMI project in PageLab 41
4.1 Project properties 41
4.1.1 General 41
4.1.2 System options 42
4.1.3 Language selection 42
4.1.4 Global periodic procedure 44
4.2 Frame set 44
4.3 Pages 45
4.3.1 Navigating between pages 45
4.3.2 Child Pages 45
4.3.3 Pop-up pages 46
4.3.4 Asynchronous messages 46
4.4 Controls 47
4.4.1 Static 47
4.4.2 Graphic element 47
4.4.3 Edit box 47
4.4.4 Text box 48

4.4.5 Image 48
4.4.6 Animation 48
4.4.7 Button 48
4.4.8 Chart 49
4.4.9 Trend 49
4.4.10 Progress bar 49
4.4.11 Combo box 49
4.4.12 Checkbox 49
4.4.13 Custom control 50

4.5 Variables 50
4.5.1 Local variables 50
4.5.2 Global variables 51
4.5.3 Variables imported from PLC 51
4.5.4 System variables 51

4.6 Multiple pages management 52
4.6.1 Association of elements of a set 52
4.6.2 Navigation of the elements of a set 53
4.6.3 Pages numbering 53

4.7 Advanced operations on pages 53
4.7.1 Export/import of pages to/from files 54
4.7.2 Export/import procedures and variables 54
4.7.3 Copy/paste of pages in the project 54
4.7.4 Rename pages 55
4.7.5 Templates of page management 55

4.8 Events 59
4.8.1 Page or control events 59
4.8.2 Key pressure events 59
4.8.3 Events raised by software 59
4.8.4 Procedures that can be associated to events 60
4.8.5 Actions that can be associated to key pressure 61

4.9 Resources 61
4.9.1 Fonts 62
4.9.2 Bitmaps 62
4.9.3 Strings table 63
4.9.4 Enumeratives 63
4.9.5 Images lists 64
4.9.6 Sets 64

4.10 Automatic documentation 65

4.11 Managing projects 66
4.11.1 Selecting the target device 66

5. Appendix I: page properties and object properties 69

5.1 Frame set 69

5.1.1 Properties 69

5.2 Child page 69

5.2.1 Properties 69

5.2.2 Events 70

5.3 Pop-up page 71

5.3.1 Properties 71

5.3.2 Events 72

5.4 Static 72

5.4.1 Properties 72

5.4.2 Events 73

5.5 Line 73

5.5.1 Properties 73

5.6 Rectangle 74

5.6.1 Properties 74

5.7 Edit box 74

5.7.1 Properties 74

5.7.2 Format specification - printf 76

5.7.3 Events 77

5.8 Text box 77

5.8.1 Properties 77

5.8.2 Events 78

5.9 Image 78

5.9.1 Properties 78

5.10 Animation 79

5.10.1 Properties 79

5.10.2 Events 80

5.11 Button 80

5.11.1 Properties 80

5.11.2 Events 82

5.12 Progress bar 83

5.12.1 Properties 83

5.12.2 Events 84

5.13 Custom control 84

5.13.1 Properties 84

5.13.2 Events 85

5.14 Chart 85

5.14.1 Properties 85

5.14.2 Events 87

5.15 Trend 87

5.15.1 Properties 87

5.15.2 Events 90

5.16 Combo box 90

5.16.1 Properties 90

5.16.2 Events 91

5.17 Check box 92

5.17.1 Properties 92

5.17.2 Events 93

1. OVERVIEW

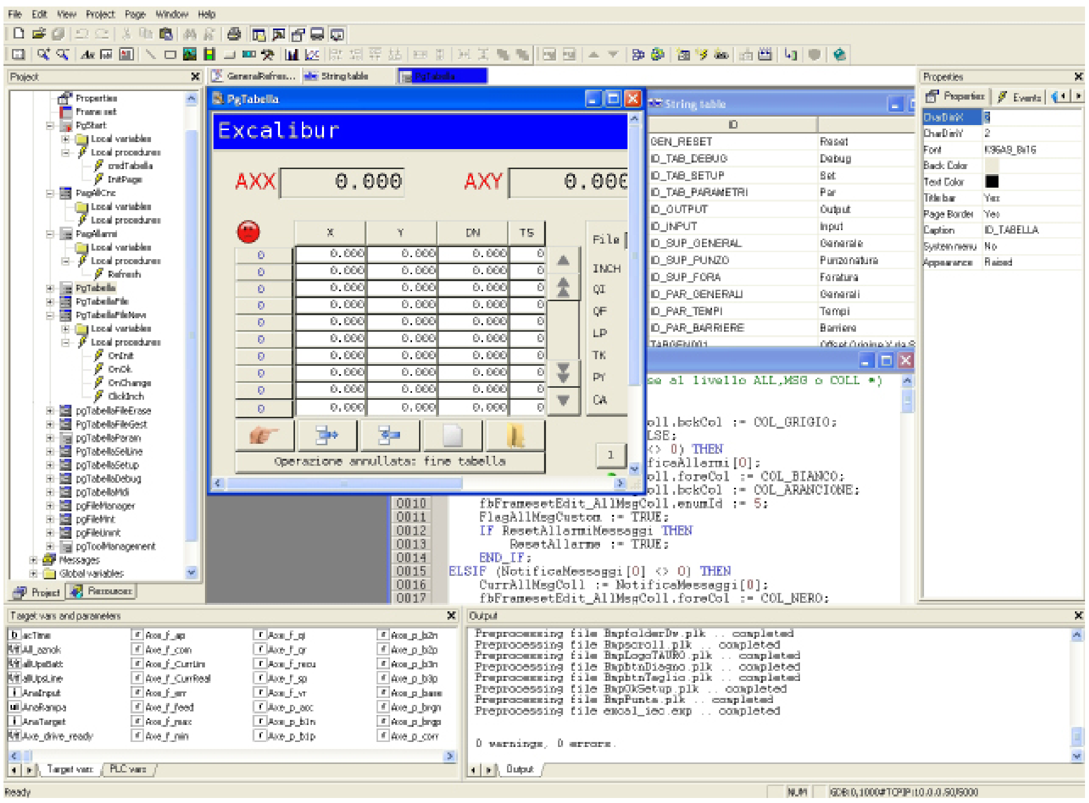
PageLab is a software application that allows the developer to create user interfaces for embedded systems based on HMI runtime.

PageLab is an easy to learn and use software, which allows the user to implement graphical interfaces in a visual way. The realized pages are viewed in PageLab as they will appear on the final target.

Thanks to its multi-pages structure, PageLab can support HMI (Human Machine Interface) applications with an arbitrary number of pages.

It is equipped with a considerable number of tools to realize even complex applications and it interfaces directly to the PLC IEC1131 LogicLab compiler for managing the variables which are defined in the target PLC application.

The following paragraphs show you the main features of this product.



File Edit View Project Page Window Help
Project
PgTabella
Excalibur
AXX 0.000
AXY 0.000
Operazione annullata: fine tabella
String table
Properties
Target vars and parameters
Output
Ready

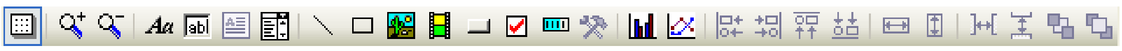
ID	
GEN_RESET	Reset
ID_TAB_DEBUG	Debug
ID_TAB_SETUP	Set
ID_TAB_PARAMETRI	Par
ID_OUTPUT	Output
ID_INPUT	Input
ID_SUP_GENERAL	Generale
ID_SUP_PUNZO	Punzonatura
ID_SUP_FORA	Foratura
ID_PAR_GENERALI	Generali
ID_PAR_TEMPI	Tempi
ID_PAR_BARRIERE	Barriere

```
se al livello ALL,MSG o COLL *)

oll.bckCol := COL_GRIGIO;
LSE;
<> 0) THEN
ficaAllarmi[0];
oll.foreCol := COL_BIANCO;
oll.bckCol := COL_ARANCIONE;
    fbFramesetEdit_AllMsgColl.enumId := 5;
    FlagAllMsgCustom := TRUE;
    IF ResetAllarmiMessaggi THEN
        ResetAllarme := TRUE;
    END_IF;
ELSIF (NotificaMessaggi[0] <> 0) THEN
    CurrAllMsgColl := NotificaMessaggi[0];
    fbFramesetEdit_AllMsgColl.foreCol := COL_NERO;
```

```
Preprocessing file BmpfolderDw.plk .. completed
Preprocessing file Bmpscroll.plk .. completed
Preprocessing file BmpLogoTAURO.plk .. completed
Preprocessing file BmpbtnDisegno.plk .. completed
Preprocessing file BmpbtnTaglio.plk .. completed
Preprocessing file BmpOkSetup.plk .. completed
Preprocessing file BmpPunta.plk .. completed
Preprocessing file emcol_iec.exp .. completed


0 warnings, 0 errors.
```

1.1 MAIN ELEMENTS

Set of controls

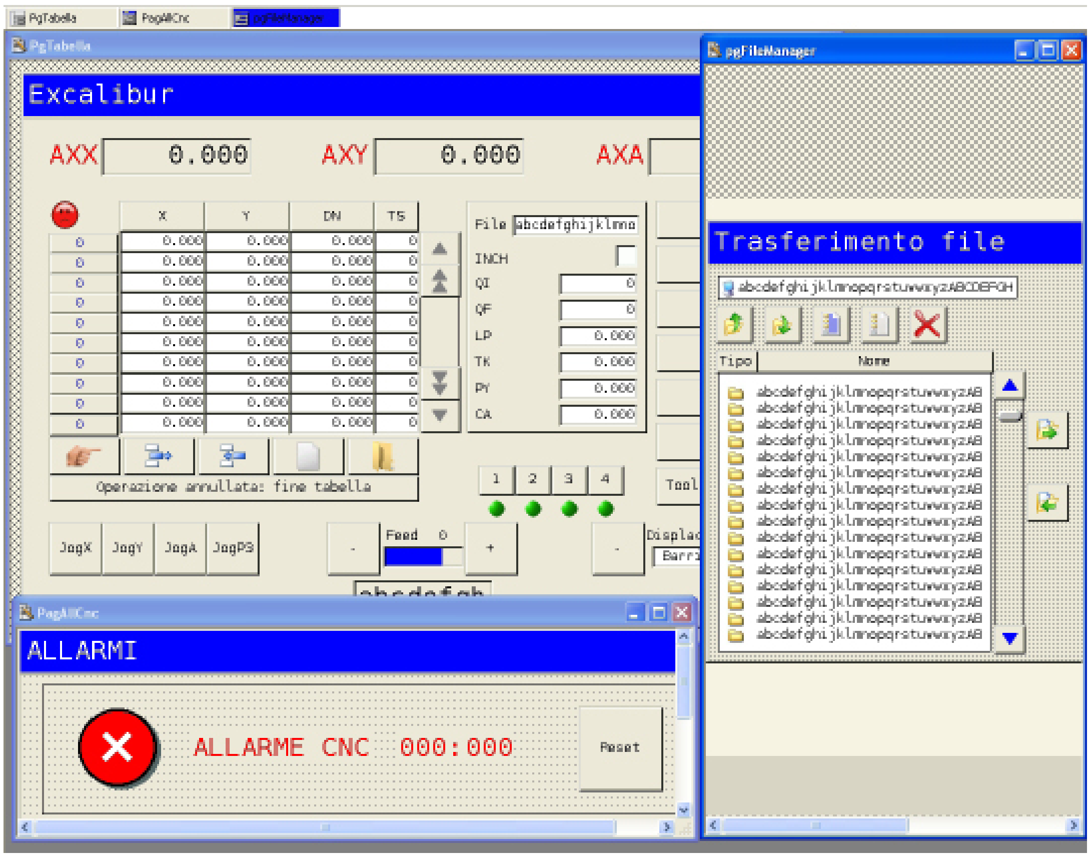


Each page may contain an arbitrary number of defined graphic controls. There are two classes of graphic controls:

- Static controls: drawing tools such as lines, rectangles, and figures.
- Dynamic controls: multilayered objects, which enable data and images display and user interaction (strings, editboxes, textboxes, buttons, progress, charts and trends, custom controls).

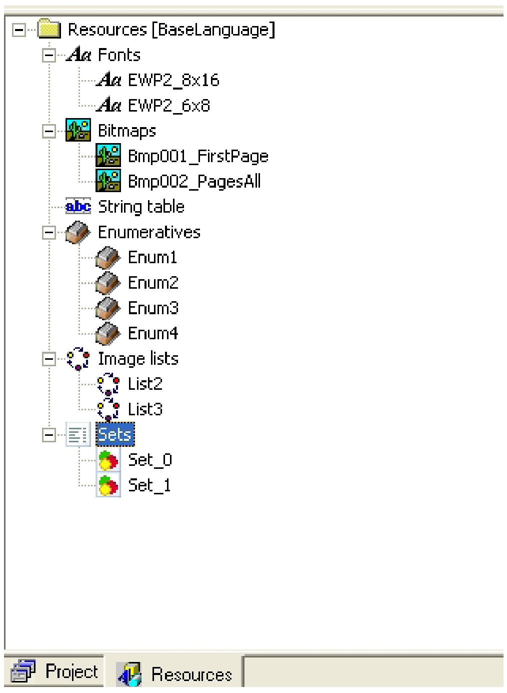
PageLab is an open system, allowing the implementation of custom controls which may be included in the target system.

Multi-pages structure



PageLab supports the definition of an arbitrary number of pages (full-screen or pop-up). Each page may contain links to other pages, so that the whole project takes a tree structure.

Resources management

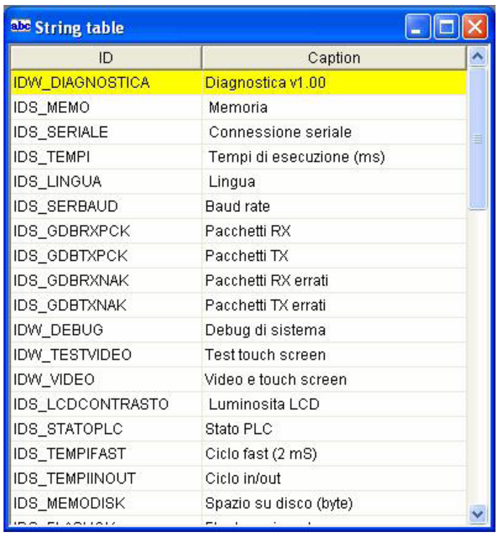


The controls' properties in the page are not statically defined in the project code, but they can be managed separately as resources.

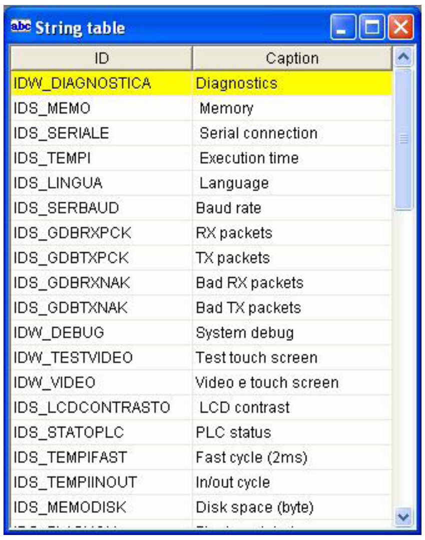
Resources include fonts for characters display, images, string table, enumerated data types, and elements sets.

Specifically regarding the images, PageLab allows to import bitmap files directly from the Windows-formatted file (`.bmp`, `.gif`, `.emf`, `.jpg`, `.ico` etc.).

Languages management



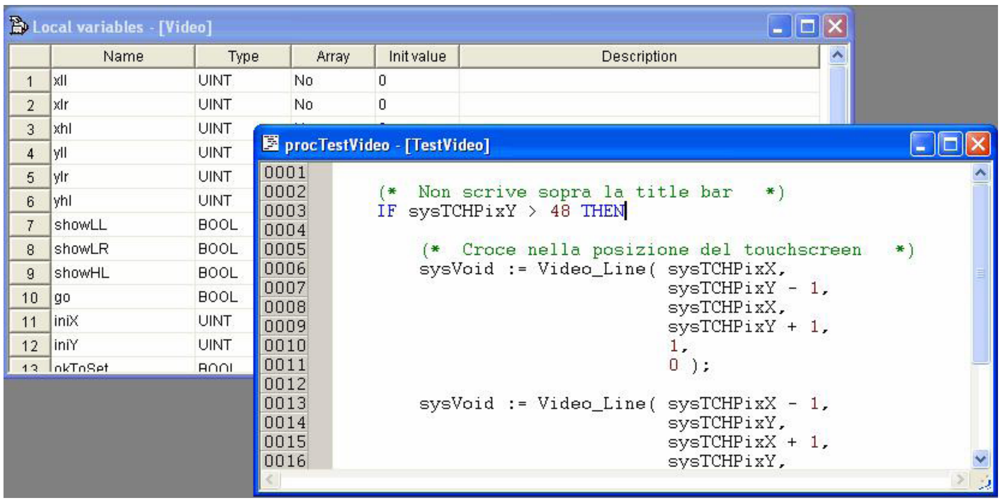
ID	Caption
IDW_DIAGNOSTICA	Diagnostica v1.00
IDS_MEMO	Memoria
IDS_SERIALE	Connessione seriale
IDS_TEMPI	Tempi di esecuzione (ms)
IDS_LINGUA	Lingua
IDS_SERBAUD	Baud rate
IDS_GDBRXPCK	Pacchetti RX
IDS_GDBTXPCK	Pacchetti TX
IDS_GDBRXNAK	Pacchetti RX errati
IDS_GDBTXNAK	Pacchetti TX errati
IDW_DEBUG	Debug di sistema
IDW_TESTVIDEO	Test touch screen
IDW_VIDEO	Video e touch screen
IDS_LCDCONTRASTO	Luminosita LCD
IDS_STATOPLC	Stato PLC
IDS_TEMPIFAST	Ciclo fast (2 mS)
IDS_TEMPIINOUT	Ciclo in/out
IDS_MEMODISK	Spazio su disco (byte)



ID	Caption
IDW_DIAGNOSTICA	Diagnostics
IDS_MEMO	Memory
IDS_SERIALE	Serial connection
IDS_TEMPI	Execution time
IDS_LINGUA	Language
IDS_SERBAUD	Baud rate
IDS_GDBRXPCK	RX packets
IDS_GDBTXPCK	TX packets
IDS_GDBRXNAK	Bad RX packets
IDS_GDBTXNAK	Bad TX packets
IDW_DEBUG	System debug
IDW_TESTVIDEO	Test touch screen
IDW_VIDEO	Video e touch screen
IDS_LCDCONTRASTO	LCD contrast
IDS_STATOPLC	PLC status
IDS_TEMPIFAST	Fast cycle (2ms)
IDS_TEMPIINOUT	In/out cycle
IDS_MEMODISK	Disk space (byte)

Strings and enumerated data types are structured as to ease the multilingual device; moreover PageLab provides a function to export/import the above mentioned elements to/from a text file, in order to simplify the translation from a language to another.

Variables and procedures



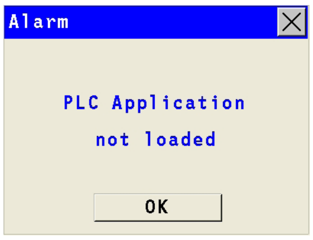
	Name	Type	Array	Init value	Description
1	xll	UINT	No	0	
2	xlr	UINT	No	0	
3	xhl	UINT			
4	yll	UINT			
5	ylr	UINT			
6	yhl	UINT			
7	showLL	BOOL			
8	showLR	BOOL			
9	showHL	BOOL			
10	go	BOOL			
11	iniX	UINT			
12	iniY	UINT			
13	okToSet	BOOL			

```
    (*  Non scrive sopra la title bar   *)
    IF sysTCHPixY > 48 THEN

        (*  Croce nella posizione del touchscreen   *)
        sysVoid := Video_Line( sysTCHPixX,
                               sysTCHPixY - 1,
                               sysTCHPixX,
                               sysTCHPixY + 1,
                               1,
                               0 );

        sysVoid := Video_Line( sysTCHPixX - 1,
                               sysTCHPixY,
                               sysTCHPixX + 1,
                               sysTCHPixY,
```

PageLab enables the implementation of procedures which may be as complex as you want in the ST language. Through these procedures, the user can interact with the PageLab application, the PLC application or the target system variables to customize the interface's behaviour or the whole CNC.

1.2 RUN-TIME FUNCTIONALITIES

Asynchronous messages management


Alarm

PLC Application not loaded

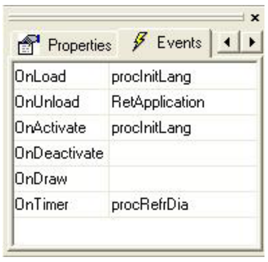
OK

PageLab supports the issue of asynchronous messages whatever their complexity. You can entirely customize the issue messages management by typing a simple ST procedure.

Multilingual support

PageLab allows you to change strings, resources, and enumerations language without recompiling nor reloading the application.

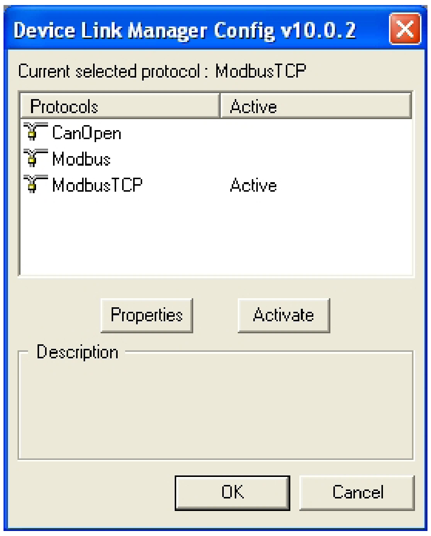
Events management



OnLoad	procInitLang
OnUnload	RetApplication
OnActivate	procInitLang
OnDeactivate	
OnDraw	
OnTimer	procRefrDia

PageLab applications are structured in events; the user may seize the available events and manage them through ST-coded procedures.

1.3 COMMUNICATING WITH THE TARGET


Device Link Manager Config v10.0.2

Current selected protocol : ModbusTCP

Protocols	Active
CanOpen	
Modbus	
ModbusTCP	Active

You can establish the communication with the target device through the PC communication drivers, thus using one of the available custom protocols (which can be easily implemented thanks to the modular structure of the communication system).

2. CREATING A SIMPLE PAGELAB PROJECT

2.1 PURPOSE OF THIS CHAPTER

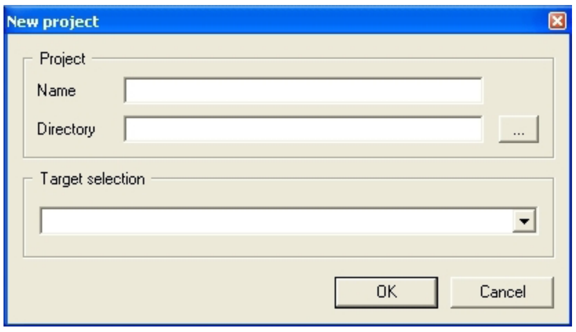
This chapter aims to lead the user to realize a simple HMI project with PageLab, through a sequence of easy steps.

Here below you can find the list of this chapter's topics.

- Creating a new project: starting at zero the realization of a HMI project.
- Inserting the first page in the project.
- Inserting a secondary page.
- Inserting static controls: how to insert simple objects (lines, rectangles, etc.) in a page.
- Inserting static images: how to insert an image in a page, starting at a *.bmp* file.
- Inserting strings: how to insert a text label.
- Inserting edit boxes: how to access the data of the system and the control PLC, how to declare new variables, how to insert text frames to view/edit these data.
- Inserting buttons: learning to use an essential control for the interaction between the user and the system.
- Compiling and downloading the project.

2.2 CREATING A NEW PROJECT

Launch PageLab, then select the *New Project* command from the *File* menu. The following dialog box appears.



Type the name you want to assign to the project in the *Name* field, and in the *Directory* field specify the directory where you want to create the project folder.

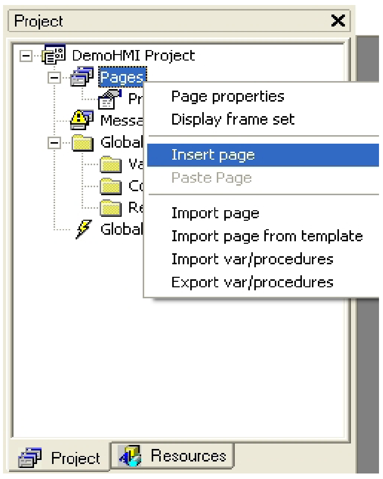
Select the target which will execute the HMI from the *Target selection* menu. The contents of this menu can be customized: if the desired target does not appear in the list, refer to your hardware provider.

Confirm your choice by pressing *OK*. PageLab automatically creates the folder *I:\Demo manuale\Demo HMI* as specified in *Directory*.

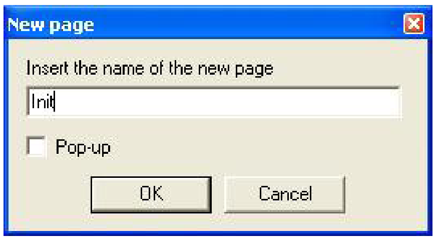
2.3 INSERTING THE FIRST PAGE IN THE PROJECT

2.3.1 CREATING A NEW PAGE

To insert a new page in the project, right-click on the `Pages` item of the project tree.

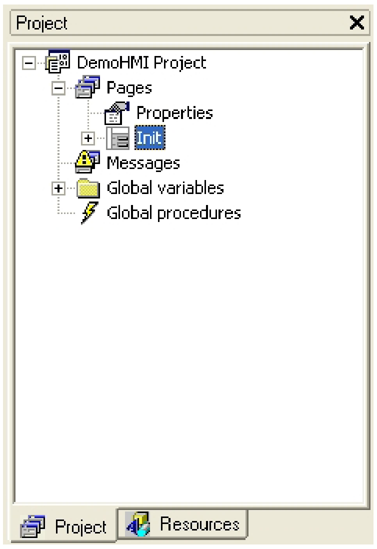


Select the *Insert page* option from the menu which has just shown up. This causes a dialog box to appear where you have to specify the page name and whether the page is a pop-up one or not.

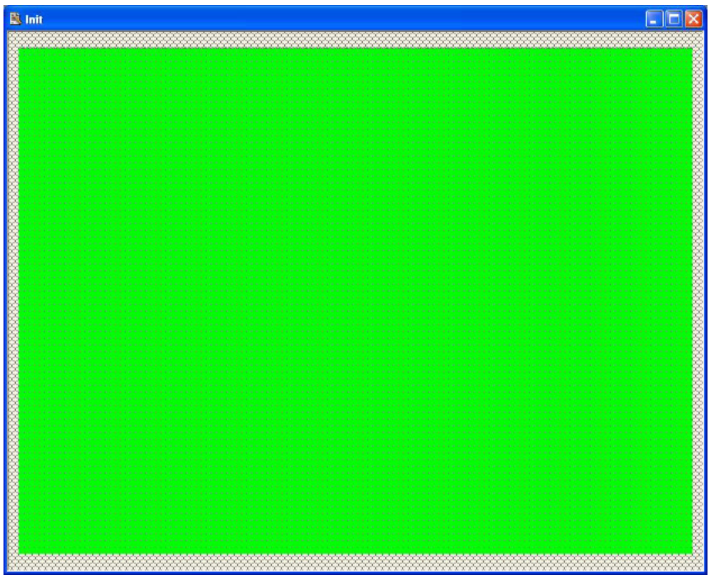


If you do not select the *Pop-up* property when creating it, the page is called *Child Page*. Its main feature is that it fits the whole video area. Consequently the user cannot define position and size of a child page because they are automatically set depending on the video area and on an eventual frame set (see 4.2).

Choose to create a child page and call it *Init*: type the name *Init* in the apposite field and press *OK* to confirm your choice. A new node appears in the pages folder of the project tree.



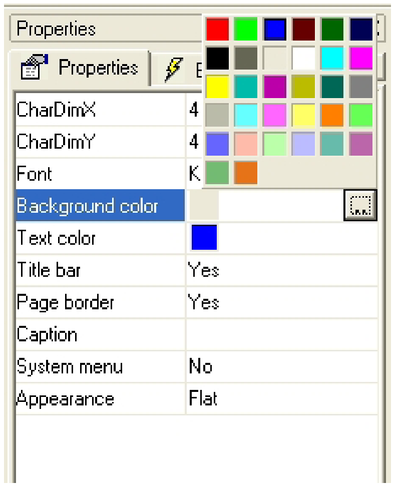
Double-click on the `Init` item to open the document with this page preview, which is blank at the moment.



2.3.2 EDITING THE COLORS OF THE PAGE

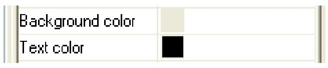
You can edit the background color of the page and the foreground default text color through the page properties: double-click in the `Background Color` field. A little button appears.

Pressing it, the colors palette appears. Then you can select the desired color.



CharDimX	4
CharDimY	4
Font	K
Background color	
Text color	🟦
Title bar	Yes
Page border	Yes
Caption	
System menu	No
Appearance	Flat

Choose grey as background color and black as default text color.

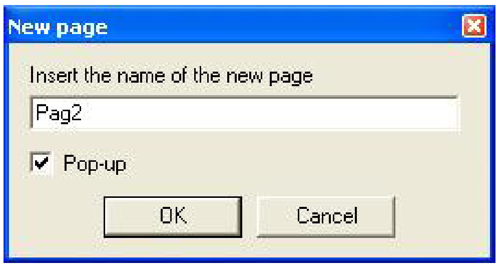


Background color	⬜
Text color	⬛

2.4 INSERTING A SECONDARY PAGE

2.4.1 CREATING A SECONDARY PAGE

Let us assume that you want to create a secondary page: right-click on the *Pages* item of the project tree and choose the *Insert page* option from the contextual menu. Type the name *Pag2* in the dialog box which appears and select the pop-up property.

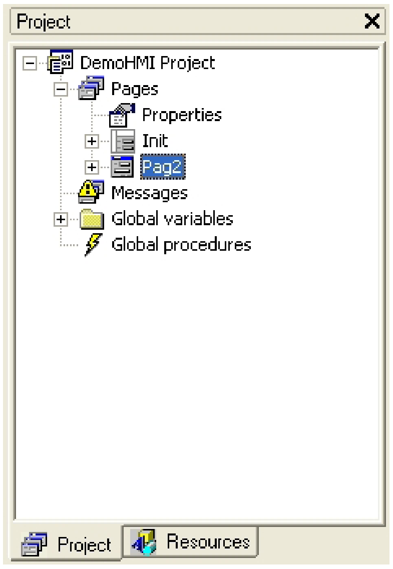

New page

Insert the name of the new page

Pag2

☑ Pop-up

Consequently a new item appears in the *Pages* folder of the project tree.

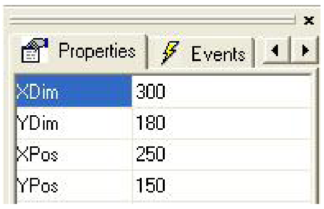


2.4.2 DIMENSIONING AND SETTING THE SECONDARY PAGE

Note that the icon of the *Init* page different from the new *Pag2* one. In fact, the last one has been created as pop-up page, whereas the first one has been created as child page.

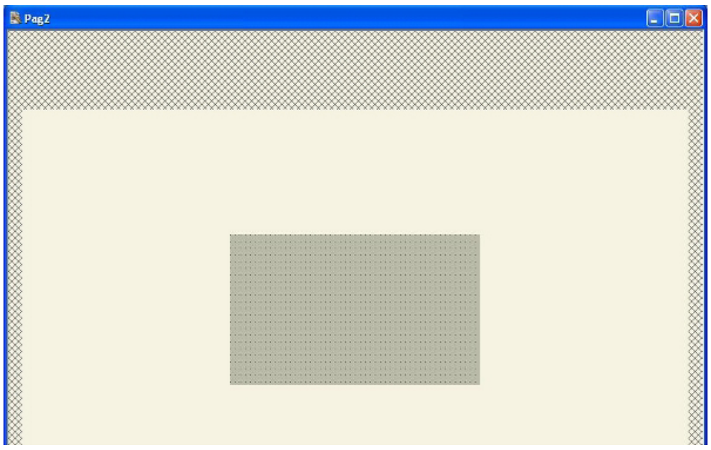
Pop-up pages are not subjected to any restriction from the frame set (see 4.2): their dimensions and positions can be chosen by the user.

Assign to the secondary window the dimensions 300x180 pixel and set it (x, y) = (250, 150) because these are the top left-hand corner's coordinates of the window. Double-click on the *Pag2* item of the project tree. In this way you open the corresponding document. Assign dimensions and position.



XDim	300
YDim	180
XPos	250
YPos	150

After editing the colors, too, the new window will look like the picture below.

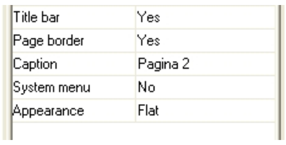


The grey area in the centre is the active area of the `Pag2` page, whereas the clearer area which surrounds it represents the video area of the target system. In this way you obtain a clear vision of the new page placement.

2.4.3 VIEWING THE TITLE BAR AND THE SYSTEM BUTTON

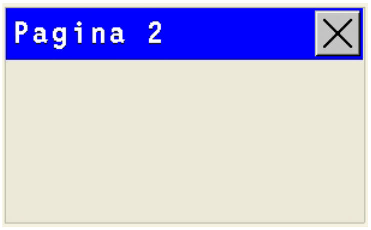
PageLab enables the automatic creation of a title bar (`Title bar` properties = `Yes`) and of a button to close the page (`System menu` properties = `Yes`), besides the print of a text string as title (`Caption` properties).

Let us assume that you want to activate the title bar and the close button, and to print the `Pagina 2` string as title.

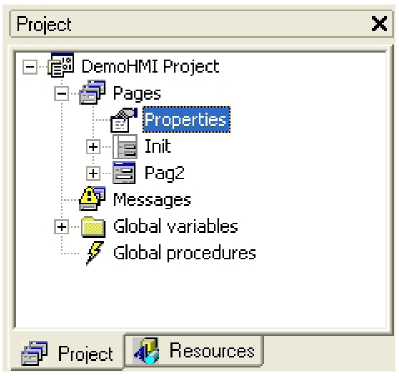


Title bar	Yes
Page border	Yes
Caption	Pagina 2
System menu	No
Appearance	Flat

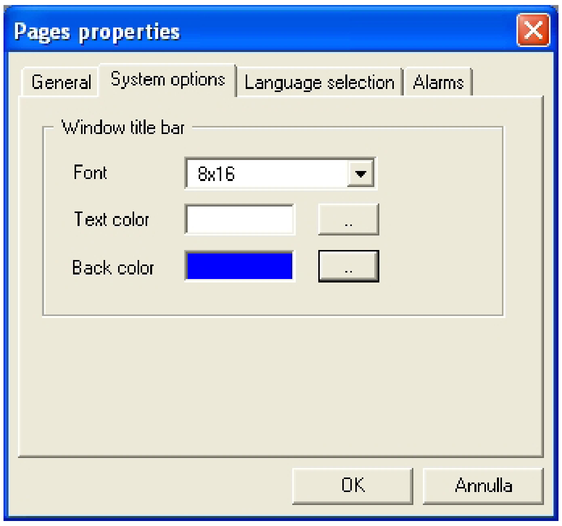
Then the secondary page looks like the following picture.



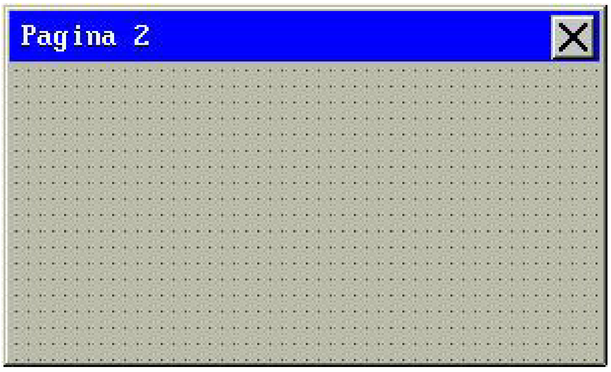
The text and the background color and the used font are the same for all the pages of the project, so you will not find them in this specific page properties. In order to customize these features, double-click on the `Properties` item of the project tree.



A multi-tabs window opens. In `System options` assign the font (in this case 8x16), the text color and the background color (in this case respectively white and blue).

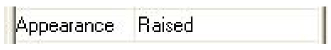


Then the secondary page looks like the following figure.

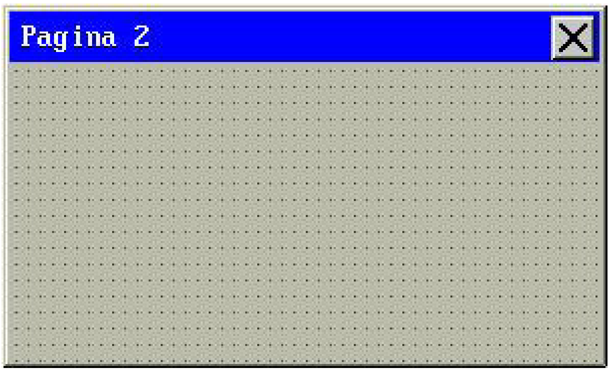


2.4.4 ASSIGNING A STYLE TO THE WINDOW

PageLab supports three styles for the windows, which you can select through the `Appear-ance` property: `Flat` (the default style when you create a window), `Sunken` and `Raised`. Choose the last one.

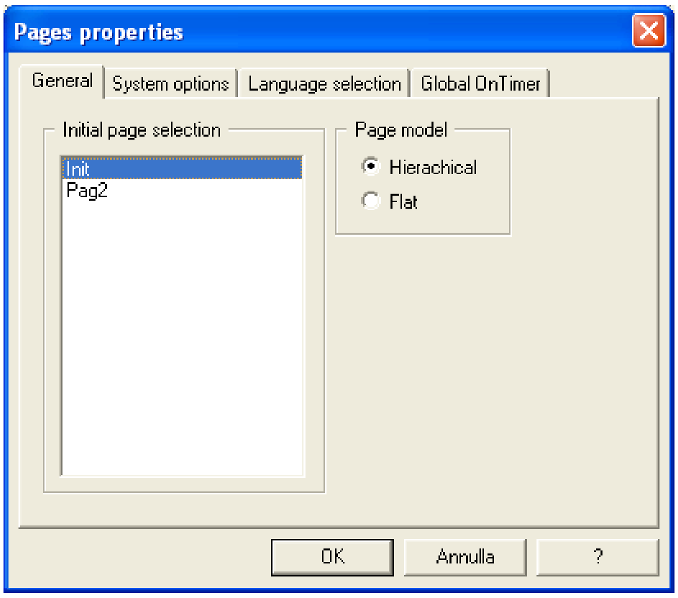

Appearance Raised

The window looks like the picture below.



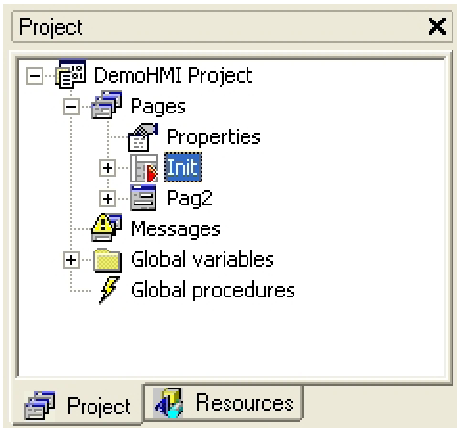
2.4.5 CHOOSING THE START WINDOW

The user has to indicate the start window of the whole HMI project. The start window will open at the HMI application start. If the project consists in one single page, the system will take this one as start page. You can indicate the start page in the project properties window, which you can open by double-clicking on the `Properties` item of the project tree. The `General` window is used for this purpose.



In order to indicate the start page, select the desired one from the list. Then confirm your choice by clicking *OK*.

The start page is marked in the project tree by a red triangle.



2.5 INSERTING STATIC CONTROLS

The two pages which you have just created are blank yet. Go back to the first page (`Init`) and start inserting some controls.

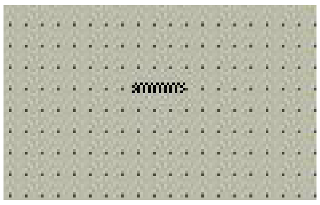
Static controls are objects which are drawn once, when opening the page, and they do not change until the page is active.

2.5.1 INSERTING A LINE

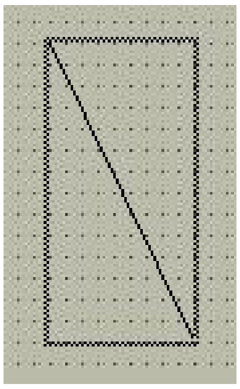
Insert a line by clicking the corresponding button in the *Page toolbar*.

Move the mouse to the active area of the page. A cross + appears. The object will be inserted in the grid near to the mouse cursor.

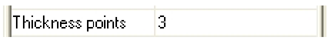
Confirm the insertion point by left-clicking. A new *Line* control appears. It has a default size and horizontal alignment.



You can resize it by dragging one of the two ends of the line.

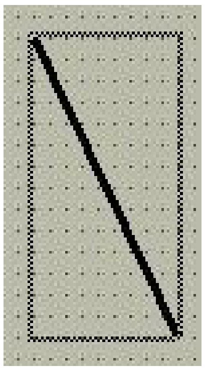


You can edit the line thickness through the *Thickness points* property of the control. For example, assign a 3 pixel thickness.



Thickness points 3

In the page preview you can see how the line looks like.

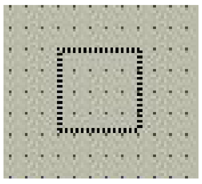


2.5.2 INSERTING A RECTANGLE IN THE PAGE

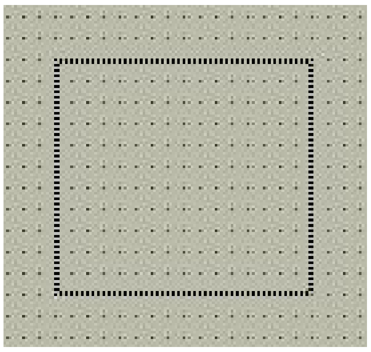
Press the corresponding button in the *Page toolbar*.

Move the mouse to the active area of the page. A cross + appears. The object will be inserted in the grid near to the mouse cursor.

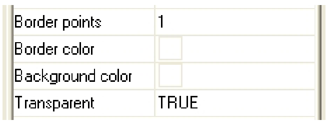
Confirm the insertion point by left-clicking. A new `Rectangle` control appears. It has a default size.



You can edit both the dimensions dragging one of the rectangle vertexes, or one dimension at a time dragging one of the rectangle's sides.

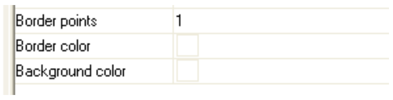


You can customize the border and the background color and the transparency through the control properties. For example, make the rectangle white and opaque with white border and thickness set to 1.



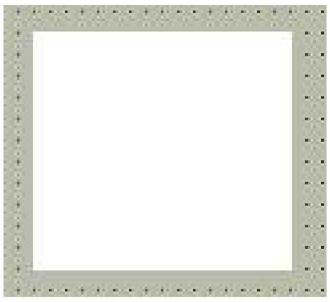
Border points	1
Border color	⬜
Background color	⬜
Transparent	TRUE

If the target has new feature of trasparency the properties are now like this.

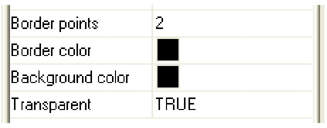


Border points	1
Border color	⬜
Background color	⬜

In the page preview you can see how the rectangle looks like.

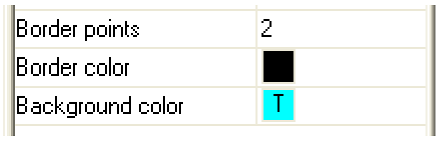


Now superimpose another rectangle to the first one. Let us assume that you want the new rectangle to be transparent with black borders, and thickness set to 2.



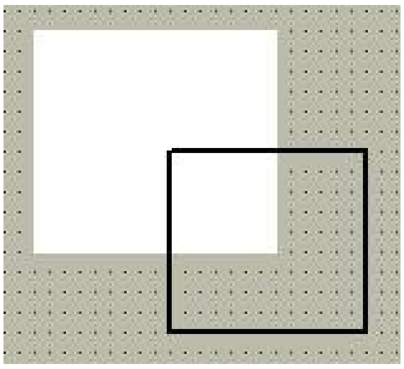
Border points	2
Border color	⬛
Background color	⬛
Transparent	TRUE

If the target has new feature of trasparency the properties are now like this.



Border points	2
Border color	■
Background color	T

In the page preview you will see the following image.



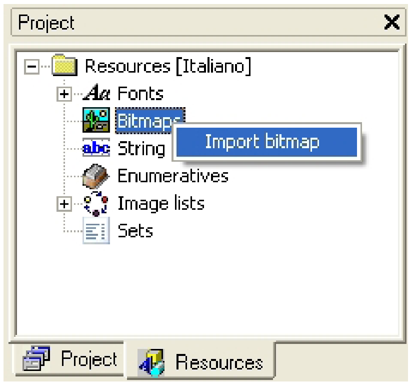
2.6 INSERTING STATIC IMAGES

The following paragraph shows you how to insert static images in the page. Static images are different from animations (images which may change dynamically, even though they have fixed position and dimensions) and from floating images (images which move in the page).

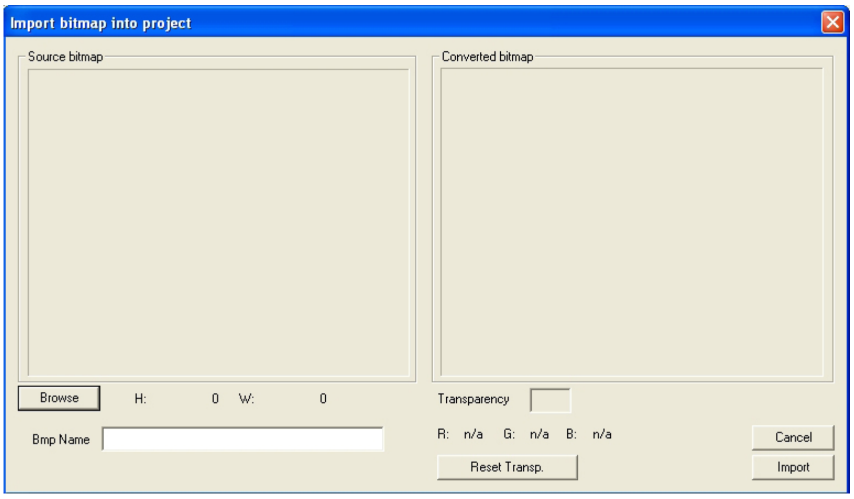
2.6.1 IMPORTING A BITMAP IN THE PROJECT

Image that has to be visualized must be available on PC as a basic Windows image file (*.bmp*, *.dib*, *.emf*, *.gif*, *.ico*, *.jpg*, *.wmf* ...). If this pre-condition holds, you can start the importing procedure.

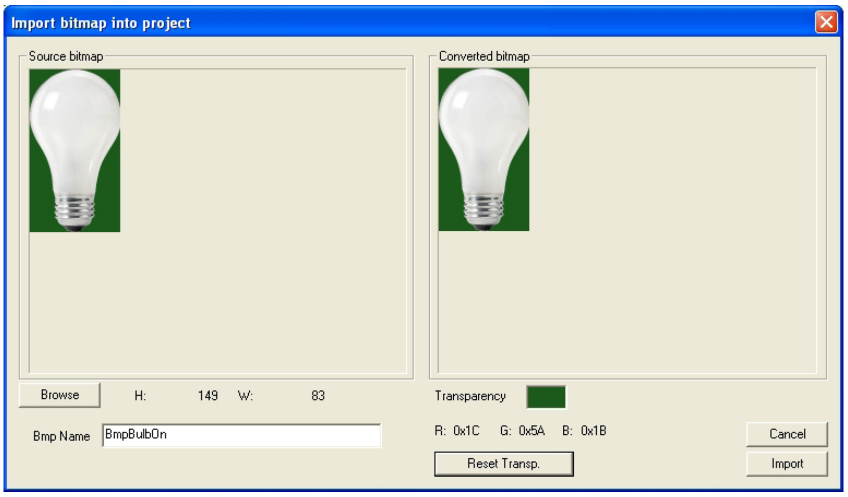
Right-click the *Bitmaps* item in the resources tree and select the *Import bitmap* command in the contextual menu which appears.



A dialog window opens.



Pressing the *Browse* button, you can navigate in the computer resources and select the source file. In this case, the source file is *BulbOn.jpg*, which represents a lighted bulb.



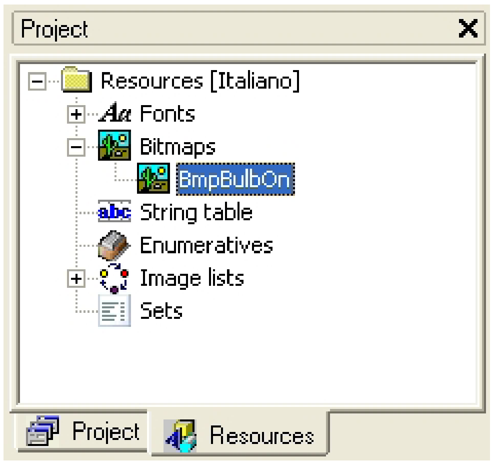
In the *Bitmap Name* field, you can assign the bitmap name which will appear in the resources tree; the default name is the file name without extension and preceded by the *Bmp* prefix.

The *Transparency color* field lets you specify the transparency color, that is a color which will not be really drawn but will let the elements appear through the bitmap background.

You can customize the transparency color by taking the desired one with the mouse from the *Converted bitmap* window.

RGB indicate the transparency color components. If the values are *n/a* it means that no transparency color has been selected. The *Reset Transp.* button lets to cancel the last selected transparency color.

At last you can confirm the operation by clicking the *Import* button. The imported bitmap appears as a new item in the resources tree.

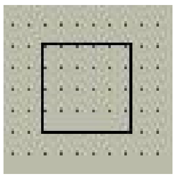


2.6.2 ASSOCIATING AN IMPORTED BITMAP WITH AN IMAGE CONTROL

The control which is aimed to display the static images is called `Image`: press the corresponding button in the `Page toolbar`.

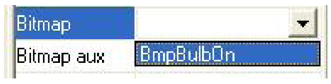
Move the mouse to the active area of the page. A cross + appears. The object will be inserted in the grid near to the mouse cursor.

Confirm the insertion point by left-clicking. A new blank frame appears.

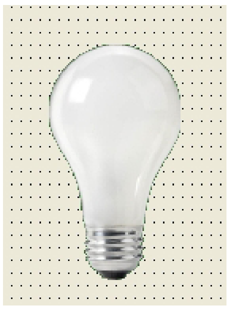


Trough the `Bitmap` property specify the image which this `Image` control must display.

Choose the desired bitmap from the list; in this case, you can see and select the only bitmap which you have imported: `BmpBulbOn`.



The control changes its size to be compatible with the assigned bitmap measures. The image in the page preview looks like the following picture.



2.7 TEXT STRINGS

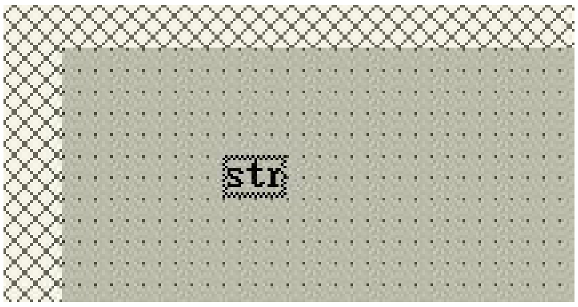
Text strings are not part of static controls because they have some properties which let them change in a page through time. Visibility, selection, and refresh may be assigned to variables, which may change their value at any time.

2.7.1 INSERTING A TEXT STRING

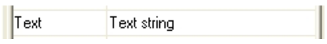
Click the corresponding button in the *Page toolbar*.

Move the mouse to the active area of the page. A cross + appears. The object will be inserted in the grid near to the mouse cursor.

Confirm the insertion point by left clicking. A new *Static* (that is string) control with the default text *str* appears.

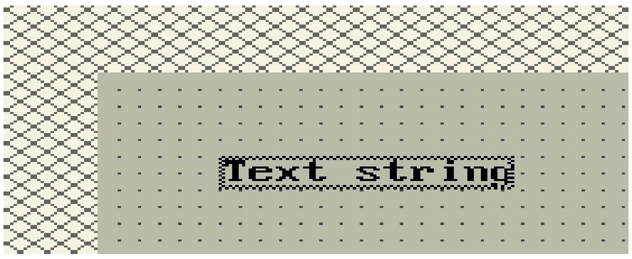


You can edit the contents of the string through the *Text* property of the control. For example, *Text string*.



Text	Text string

The page preview looks like the image below.



This is the basic use of the string. Alternatively you can assign strings by taking them from the resources (see 4.9.3).

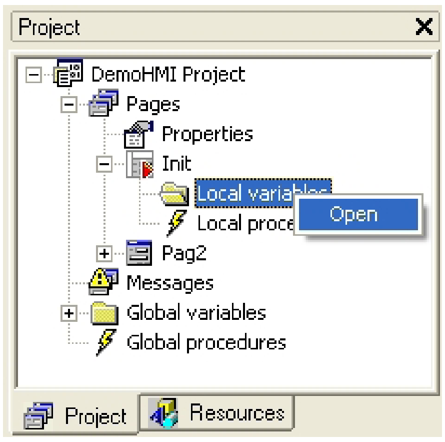
2.8 DATA MANAGEMENT IN PAGELAB

This paragraph shows you the variables management in PageLab. It is possible to distinguish the data in local variables (visible in the page scope only) and global variables (visible from every page). For some controls it is possible to use parameters and sets.

2.8.1 DECLARING A LOCAL VARIABLE

First of all declare a local variable, which you can use just in the specific page where the declaration takes place.

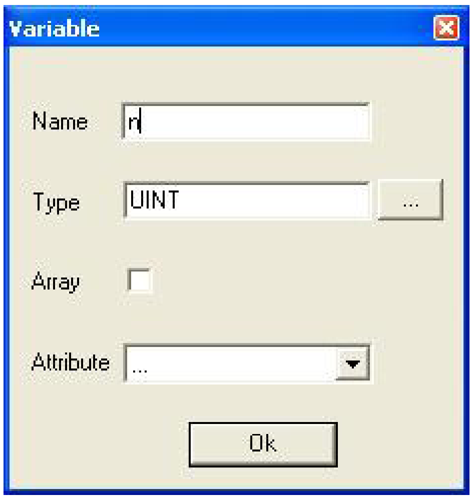
In the pages tree, under the *Init* page item, right-click on the *Local variables* item and select `Open` in the contextual menu which appears.



The local variables editor window opens. It is blank at present.

Click the `New record` button in the `Project toolbar`.

A dialog window opens requesting to specify the new variable's basic features. We can declare *n* as a new 16 bit unsigned integer variable.

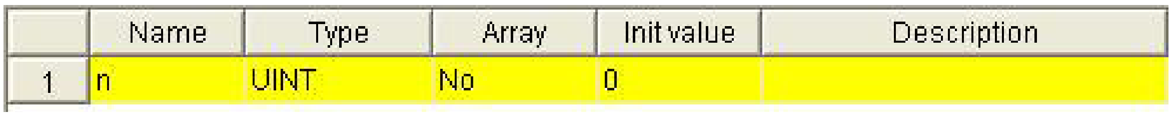


Variable

Name	n
Type	UINT
Array	☐
Attribute	...

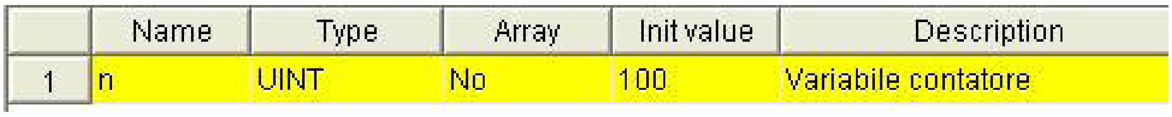
Ok

Confirm the operation by clicking `Ok`. The new corresponding record is added to the variables editor.



	Name	Type	Array	Init value	Description
1	n	UINT	No	0	

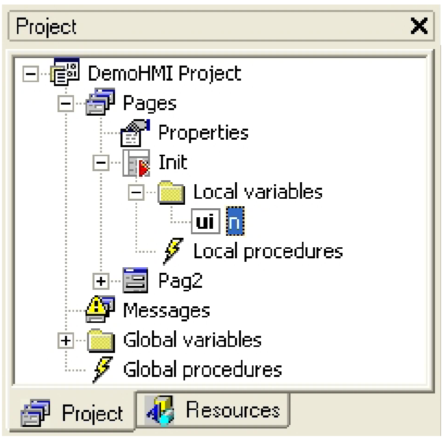
You can change this new variable's features editing the fields of the record which you have just created. For example, you may assign an initial value different from null and a comment.



	Name	Type	Array	Init value	Description
1	n	UINT	No	100	Variabile contatore

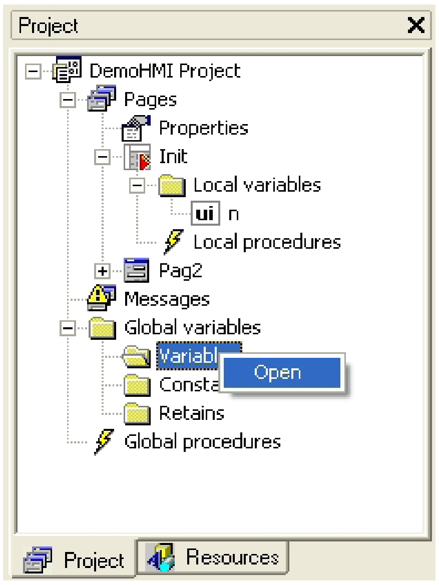
When you save the project by clicking the apposite button

or when you close the variables editor, PageLab adds a new item in the pages tree. It corresponds to the local variable which you have just declared.



2.8.2 DECLARING A GLOBAL VARIABLE

Let us assume that you want to declare a floating point global variable *t*: right-click on the `Variables` item under the `Global variables` node of the resources tree and select the `Open` command in the contextual menu which appears.



Follow the steps as shown in paragraph 2.8.1, until the new global variable appears as a new item in the pages tree.

2.8.3 IMPORTING THE PLC VARIABLES IN THE PAGELAB PROJECT

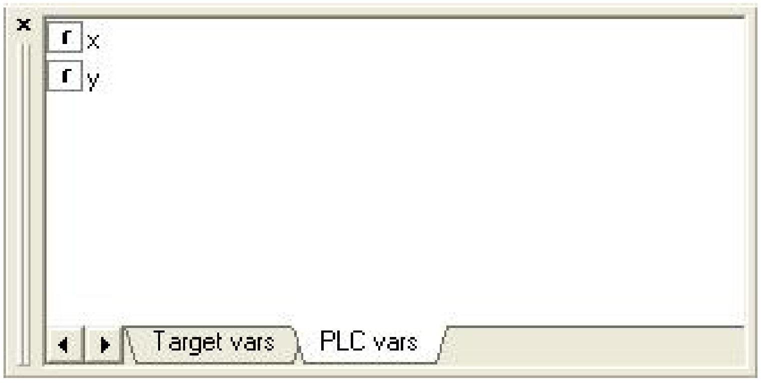
Usually an HMI project is not a stand-alone one, but is an interface for a PLC. More precisely, if the PLC project has been carried out with LogicLab, you can easily publish some variables to PageLab.

A variable of the LogicLab project can be exported to PageLab if it has been allocated on a datablock (it is not an automatic variable). If this pre-condition holds, when compiling the PLC, the program automatically creates an *.exp* file, which contains a list of the exported variables with their location in the datablocks, which the PageLab program can work out.

In order to import in PageLab the variables which have been exported from the PLC LogicLab project, you have to select the *Link PLC variables file...* from the *Project* menu.

A window opens and lets you select the file which contains the exported variables.

If you confirm to include the *.exp* file in the PageLab project, a new table called *PLC vars* appears in the libraries window. It contains the list of the exported variables.



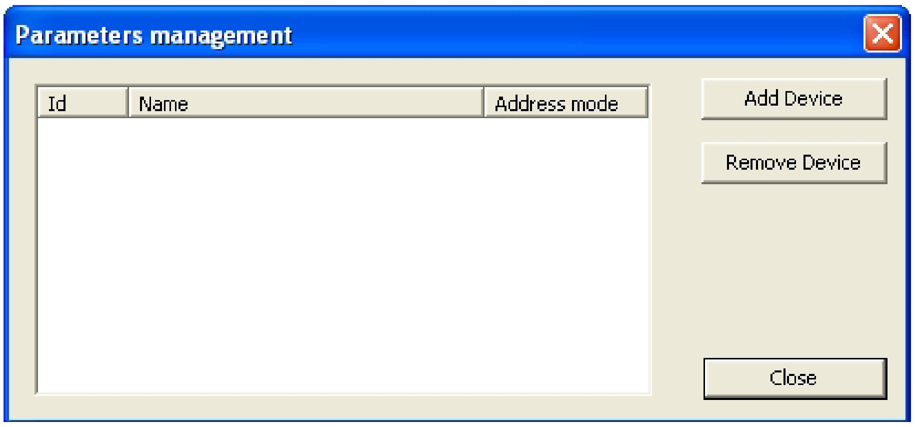
When you need to update the list of the exported variables, if the *.exp* file has not been moved to another directory, it is not necessary to repeat the above mentioned procedure. It is enough to launch the *Refresh PLC variables* command from the *Project* menu.

2.8.4 INSERTING FIELD PARAMETERS

Target system usually has internal variables and is connected on a fieldbus, so it needs to show some variables of the different devices which are connected on the net.

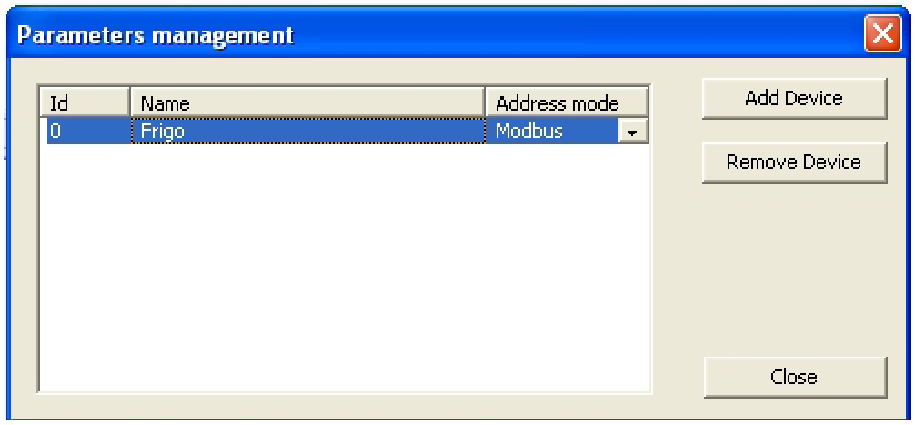
For this reason, PageLab lets you link a specific file which contains the variables definition on the bus. Click the apposite button in the toolbar.

The parameters management window appears.



Through the *Add Device* button you can add a new object linked to the target on the fieldbus.

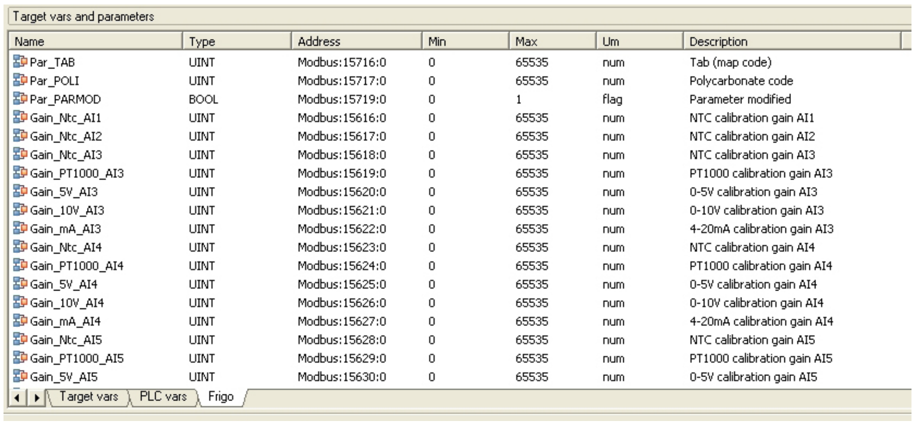
The selection window appears. Then you have to take from your PC a *.parx* file (see chapter 7). After inserting this file, the parameters management window will look like the image below.



Id	Name	Address mode
0	Frigo	Modbus

A device called *Frigo* has been inserted. In order to see the relevant parameters, click the *Close* button.

In the *Window target vars and parameters* you will see the device and its parameters.



Name	Type	Address	Min	Max	Um	Description
Par_TAB	UINT	Modbus:15716:0	0	65535	num	Tab (map code)
Par_POLI	UINT	Modbus:15717:0	0	65535	num	Polycarbonate code
Par_PARMOD	BOOL	Modbus:15719:0	0	1	flag	Parameter modified
Gain_Ntc_AI1	UINT	Modbus:15616:0	0	65535	num	NTC calibration gain AI1
Gain_Ntc_AI2	UINT	Modbus:15617:0	0	65535	num	NTC calibration gain AI2
Gain_Ntc_AI3	UINT	Modbus:15618:0	0	65535	num	NTC calibration gain AI3
Gain_PT1000_AI3	UINT	Modbus:15619:0	0	65535	num	PT1000 calibration gain AI3
Gain_5V_AI3	UINT	Modbus:15620:0	0	65535	num	0-5V calibration gain AI3
Gain_10V_AI3	UINT	Modbus:15621:0	0	65535	num	0-10V calibration gain AI3
Gain_mA_AI3	UINT	Modbus:15622:0	0	65535	num	4-20mA calibration gain AI3
Gain_Ntc_AI4	UINT	Modbus:15623:0	0	65535	num	NTC calibration gain AI4
Gain_PT1000_AI4	UINT	Modbus:15624:0	0	65535	num	PT1000 calibration gain AI4
Gain_5V_AI4	UINT	Modbus:15625:0	0	65535	num	0-5V calibration gain AI4
Gain_10V_AI4	UINT	Modbus:15626:0	0	65535	num	0-10V calibration gain AI4
Gain_mA_AI4	UINT	Modbus:15627:0	0	65535	num	4-20mA calibration gain AI4
Gain_Ntc_AI5	UINT	Modbus:15628:0	0	65535	num	NTC calibration gain AI5
Gain_PT1000_AI5	UINT	Modbus:15629:0	0	65535	num	PT1000 calibration gain AI5
Gain_5V_AI5	UINT	Modbus:15630:0	0	65535	num	0-5V calibration gain AI5

When you need to update the list of parameters, if the *.parx* file has not been moved to another directory, it is not necessary to repeat the above mentioned procedure, but it is enough to press the button

2.9 INSERTING EDIT BOX

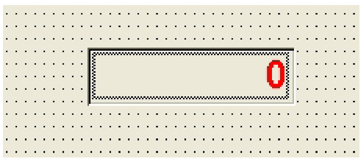
An edit box is a text frame which lets you display and eventually edit an associated variable or parameter.

2.9.1 INSERTING AN EDIT BOX IN THE PAGE

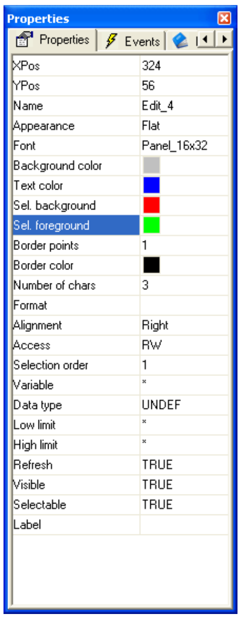
Insert an *Edit box* control in the page by pressing the corresponding button in the *Page toolbar*.

Move the mouse to the active area of the page. A cross + appears. The object will be inserted in the grid near to the mouse cursor.

Confirm the insertion point by left-clicking. A new text frame appears. It consists by default in a certain number of characters and its font is specified in the *Font* property of the page.



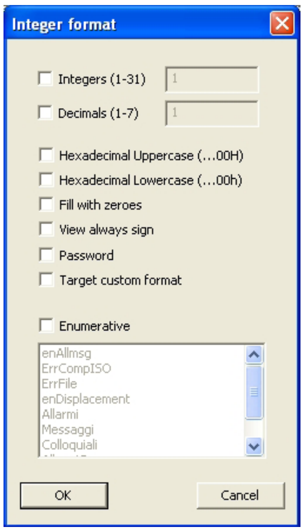
Edit this control’s properties as you can see below.



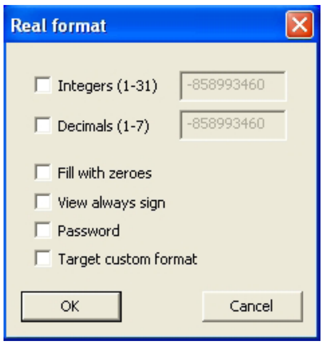
XPos	324
YPos	56
Name	Edit_4
Appearance	Flat
Font	Panel_16x32
Background color	⬜
Text color	🟦
Sel. background	🟥
Sel. foreground	🟩
Border points	1
Border color	⬛
Number of chars	3
Format	
Alignment	Right
Access	RW
Selection order	1
Variable	*
Data type	UNDEF
Low limit	*
High limit	*
Refresh	TRUE
Visible	TRUE
Selectable	TRUE
Label	

In the following list you can find all the changes which may be carried out:

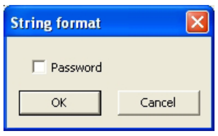
- *Appearance*: you can make the edit box appearance “sunken” by assigning the *Sunken* property.
- *Font*: you can customize font by choosing, for example, a 16x32 font instead of the default 8x16 font.
- *Select background* and *Select Foreground*: respectively text and background colors when the edit box is selected.
- *Number of Chars*: maximum number of characters which can be displayed.
- *Access*: in order to set the read-only mode, replace *RW* (read-write) with *RO* (read-only).
- *Refresh*: in order to constantly update the contents of the edit box, select the *TRUE* option. Otherwise, the contents are refreshed just when drawing the page for the first time.
- *Label*: if the target has a touchscreen display, shows keyboard and has this feature enabled, it is possible to add this text/’string resource’ as header of keyboard.
- *Format*: it represents the display format of the associated variable’s value. The format value can be inserted only if a variable is just available. It opens a dialog window with these settings according to the type of variable (integer, real, string).



- `Integers`: number of digit before comma
- `Decimals`: number of digit after comma
- `Hexadecimal Uppercase`: the number is shown as 0...0H representation with uppercase H letter
- `Hexadecimal Lowercase`: the number is shown as 0...0h representation with lowercase h letter
- `Fill with zeros`: fill the entire editbox controls with 0 where there are not numbers
- `View always sign`: show the +/- symbol in editbox
- `Password`: show only * symbols
- `Target custom format`: the target can define custom format to show the data in a particular way. In that case there is a variable on the target with the value of the corresponding user mode.
- `Enumerative`: this representation allows to select a string value corresponding to numeric value defined in `Resources`, under `Enumeratives`.



- `Integers`: number of digit before comma
- `Decimals`: number of digit after comma
- `Fill with zeros`: fill the entire editbox controls with 0 whrere there are not numbers
- `View always sign`: show the +/- symbol in editbox
- `Password`: show only * symbols
- `Target custom format`: the target can define custom format to show the data in a particular way. In that case there is a variable on the target with the value of the corresponding user mode.



- *Password*: show only * symbols.

The *Target custom format* is a special feature which enables a particular custom format implemented on the target.

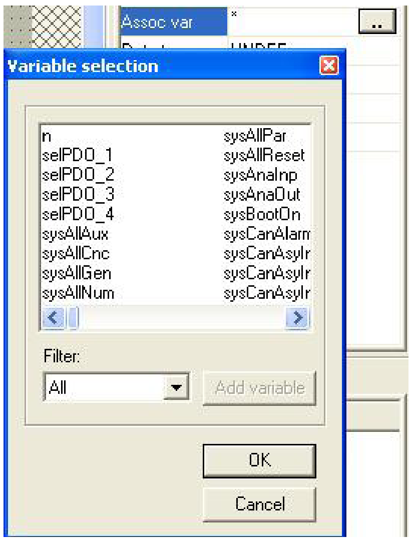
The format is specified according with language *printf* syntax (see 5.7.2).

2.9.2 EDIT BOX AND PAGELAB LOCAL VARIABLE ASSOCIATION

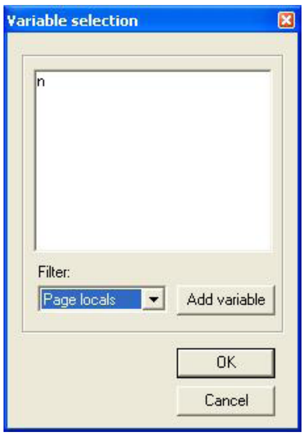
The edit box which you have just inserted lacks an essential element: the associated variable to take the values to display from. Let us assume that you want to link the edit box to a local variable (in order to get information on how to declare a local variable, see 2.8.1).

Select the edit box by clicking it once and select the *Variable* property.

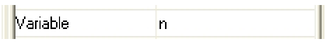
You can either type the name of the variable or click on the field and open the dialog window by clicking on the apposite button.



You can restrict the research just to the local variables of the *Init* page (consequently only the *n* variable) by using the *Filter* tool.



Select the local variable. The `Variable` field in the table properties refreshes accordantly.



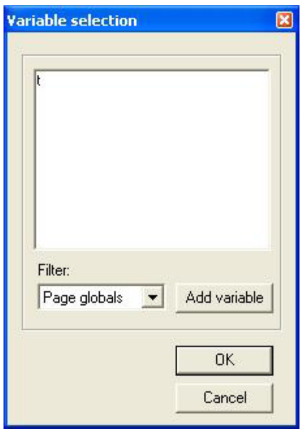
Variable	n

Then the `Edit box` control shows the *n* local variable's value constantly refreshed.

2.9.3 EDIT BOX AND PAGELAB GLOBAL VARIABLE ASSOCIATION

The principle to associate the `Edit box` control with a global variable is similar to the one to associate the `Edit box` control with a local variable. The difference consists in the variable declaration (in order to get information on how to declare a global variable, see § 2.8.2).

You can associate the Edit box with the global variable through the dialog window which was introduced in the preceding paragraph, but in this case it is necessary to use a different filter in the `Filter` field.

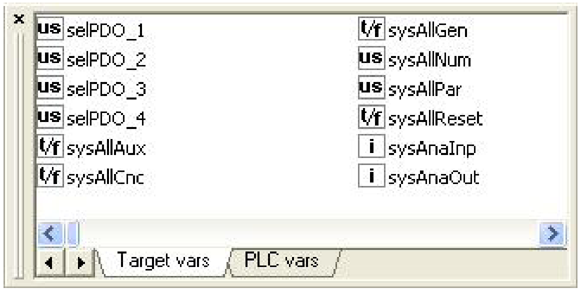


Variable selection

Filter: Page globals

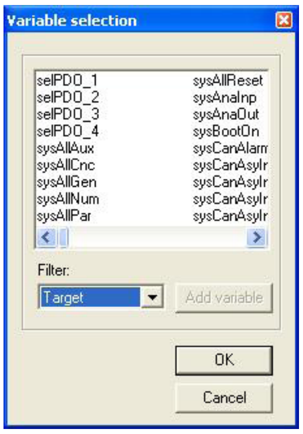
2.9.4 LINKING AN EDIT BOX WITH A TARGET (OR SYSTEM) VARIABLE

The target system executing PLC and HMI often publishes some variables which allow the interaction between user interface and system. In PageLab, such variables are called target variables. You can view them in the `Target vars` table of the `Target Vars and Parameters` window.



us selPDO_1	t/f sysAllGen
us selPDO_2	us sysAllNum
us selPDO_3	us sysAllPar
us selPDO_4	t/f sysAllReset
t/f sysAllAux	i sysAnaInp
t/f sysAllCnc	i sysAnaOut

You can associate an Edit box with a target variable through the dialog window which opens from the `Variable` field, but in this case it is necessary to use a different filter in the `Filter` field.



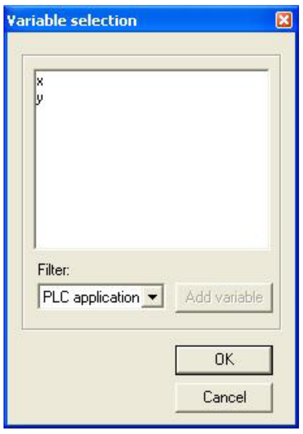
Variable selection

```
selPDO_1        sysAllReset
selPDO_2        sysAnaInp
selPDO_3        sysAnaOut
selPDO_4        sysBootOn
sysAllAux       sysCanAlarm
sysAllCnc       sysCanAsylr
sysAllGen       sysCanAsylr
sysAllNum       sysCanAsylr
sysAllPar       sysCanAsylr
```

Filter: Target

2.9.5 LINKING AN EDIT BOX WITH A PLC LOGICLAB VARIABLE

You can associate an Edit box with a PLC LogicLab variable through the dialog window which opens from the `Assoc var` field (see 2.9.2), but in this case it is necessary to use a different filter in the `Filter` field.



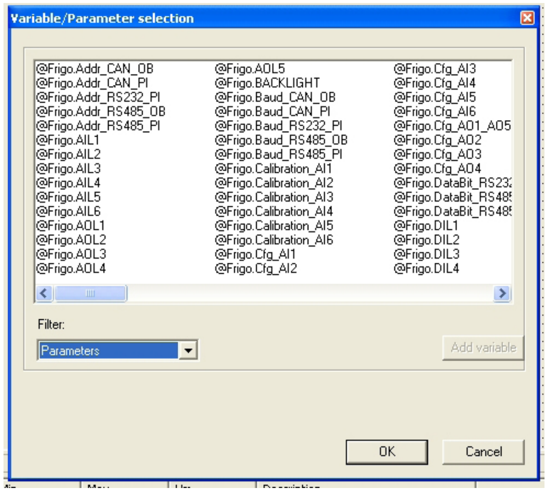
Variable selection

```
x
y
```

Filter: PLC application

2.9.6 LINKING AN EDIT BOX TO A PARAMETER

You can associate an Edit box with a parameter through the dialog window which opens from the `Variable` field (see 2.9.2), but in this case it is necessary to use a different filter in the `Filter` field.



Variable/Parameter selection

```
@Frigo.Addr_CAN_OB       @Frigo.AOL5                 @Frigo.Cfg_AI3
@Frigo.Addr_CAN_PI       @Frigo.BACKLIGHT            @Frigo.Cfg_AI4
@Frigo.Addr_RS232_PI     @Frigo.Baud_CAN_OB          @Frigo.Cfg_AI5
@Frigo.Addr_RS485_OB     @Frigo.Baud_CAN_PI          @Frigo.Cfg_AI6
@Frigo.Addr_RS485_PI     @Frigo.Baud_RS232_PI        @Frigo.Cfg_AO1_AO5
@Frigo.AIL1              @Frigo.Baud_RS485_OB        @Frigo.Cfg_AO2
@Frigo.AIL2              @Frigo.Baud_RS485_PI        @Frigo.Cfg_AO3
@Frigo.AIL3              @Frigo.Calibration_AI1      @Frigo.Cfg_AO4
@Frigo.AIL4              @Frigo.Calibration_AI2      @Frigo.DataBit_RS232
@Frigo.AIL5              @Frigo.Calibration_AI3      @Frigo.DataBit_RS485
@Frigo.AIL6              @Frigo.Calibration_AI4      @Frigo.DataBit_RS485
@Frigo.AOL1              @Frigo.Calibration_AI5      @Frigo.DIL1
@Frigo.AOL2              @Frigo.Calibration_AI6      @Frigo.DIL2
@Frigo.AOL3              @Frigo.Cfg_AI1              @Frigo.DIL3
@Frigo.AOL4              @Frigo.Cfg_AI2              @Frigo.DIL4
```

Filter: Parameters

The name of parameters is composed of `@device.variable name`, differently from variables which show just their name.

The parameter may be inserted in the apposite controls property in the following forms:

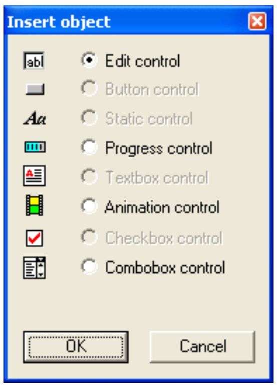
- explicit form = *@d.oi.os:type: d* = numerical ID of the device, *oi = object index*, *os = object subindex type= PLC type* (e. g. *@1.2010.0:UINT*);
- implicit form = *@dev.name: dev* = symbolic identifier of the device, *name* = symbolic name of the parameter (e. g. *Frigo.AIL1*).

The *d* (ID) field of the device is a numerical or symbolic identifier (to be defined at project creation). It refers to a specific device which may be local (the device which executes the pages itself) or on the fieldbus.

The *dev* field is a symbolic identifier of a device whose numerical ID can be retrieved by PageLab.

2.9.7 LINKING AN EDIT BOX TO A VARIABLE BY DRAGGING AND DROPPING

You may add variables and parameters to the *Target vars and parameters* window by dragging and dropping them in the page. PageLab will request to define the type of control to insert, to associate it with the variable.



2.10 INSERTING BUTTONS

Buttons are very versatile controls which play an essential role in the interaction between user and system, particularly in case of touchscreen systems without keyboard.

This chapter's aim is to show four kinds of use of the button control:

- LED-button to view the state of a boolean variable;
- command button of a boolean variable's state;
- opening button of a secondary page;
- activation button to start the execution of a customized procedure.

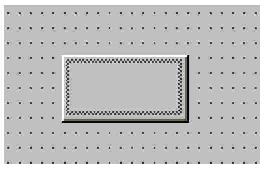
2.10.1 INSERTING A LED-BUTTON

The following paragraph teaches you how to use a button which shows an associated boolean variable's state.

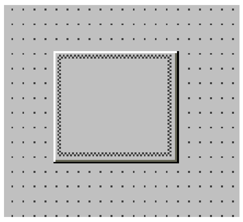
Insert a new button in the page by pressing the corresponding button in the *Page toolbar*.

Move the mouse in the active area of the page; a cross + appears. The object will be inserted in the grid near to the mouse cursor.

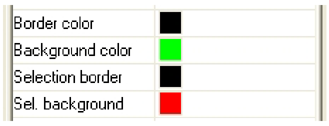
Confirm the insertion point by left-clicking. A new `Button` control appears. It has a default size.



You may edit both the dimensions by dragging one of the button's vertexes or one dimension at a time by dragging one of the button's sides.



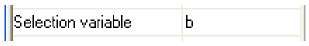
The `Border color` and the `Background Color` properties determine the border and the background color when the button is inactive, whereas the `Selection Border` and `Selection Background` properties define the border and the background color when the button is selected.



Border color	⬛
Background color	🟩
Selection border	⬛
Sel. background	🟥

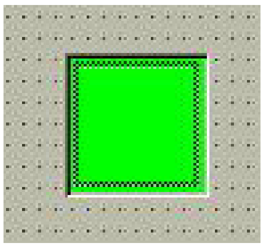
The `Selection variable` property determines the state of the button and, consequently, the couple of colors related to the control. This property may be associated either with a constant value (*FALSE* = the control is always inactive, *TRUE* = the control is always selected, on *PRESS* = the control is automatically selected when the user presses the button) or with a boolean variable whose value determines dynamically the selection state.

Declare a boolean global variable *b* and associate it with the control button as selection variable.



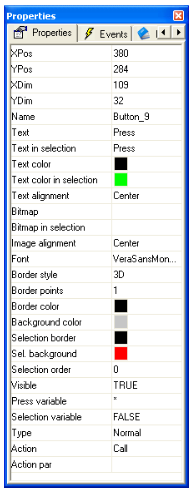
Selection variable	b

You may customize the button appearance through the `Appearance` property. For example, choose the `Sunken` option.



2.10.2 INSERTING A BOOLEAN VARIABLE COMMAND BUTTON

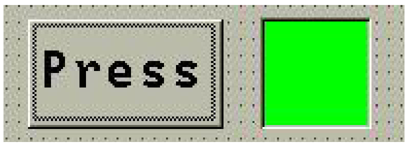
Insert a new button in the page by following the aforesaid instructions (see 2.10.1). Set it beside the LED button and let a text string show on it by means of the `Text property`.


Properties

Properties | Events

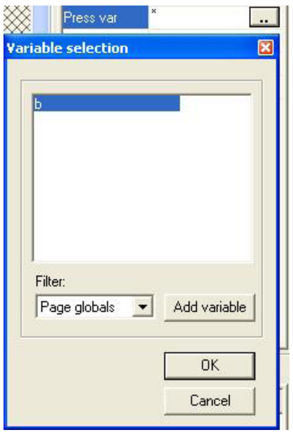
XPos	380
YPos	284
XDim	109
YDim	32
Name	Button_9
Text	Press
Text in selection	Press
Text color	⬛
Text color in selection	🟩
Text alignment	Center
Bitmap	
Bitmap in selection	
Image alignment	Center
Font	VeraSansMon...
Border style	3D
Border points	1
Border color	⬛
Background color	▒
Selection border	⬛
Sel. background	🟥
Selection order	0
Visible	TRUE
Press variable	*
Selection variable	FALSE
Type	Normal
Action	Call
Action par	

The preview looks like as follows.



The *Press variable* property allows the user to associate a boolean variable with a button control. The boolean variable's value corresponds to the pressure state of the button.

For example, associate the button which you have just created with the global variable *b* which has been created paragraph 2.10.1.


Press var *

Variable selection

b

Filter:

Page globals

At runtime, the LED-button (see 2.10.1) will be red when pressing the *Press* button. Otherwise it will be green.

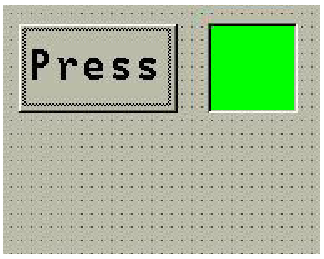
2.10.3 INSERTING A BUTTON TO OPEN A CHILD PAGE

Paragraph 2.4.1 showed you how to create a pop-up page.

The following paragraph explains how to invoke the *Pag2* page from the *Init* page by pressing a button.

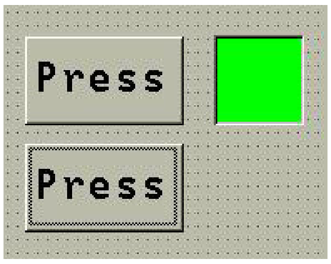
First of all insert a new button in *Init* and set it under the previously created *Press* button (see 2.10.2). As it should be exactly alike the previous one except the text string and the function, you can copy and paste the *Press* button and afterwards customize its properties.

Select the *Press* button by clicking once: the selection rectangle appears inside the control.

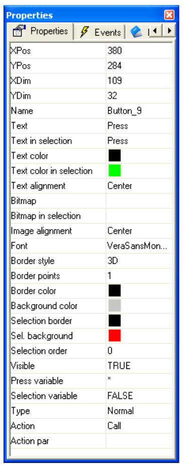


Press successively *Ctrl+C* and *Ctrl+V*. A cross + appears. The object will be inserted in the grid near the mouse pointer.

Confirm the insertion point by clicking under *Press*. A copy of the control appears; it is the same as the source button except its position and name.

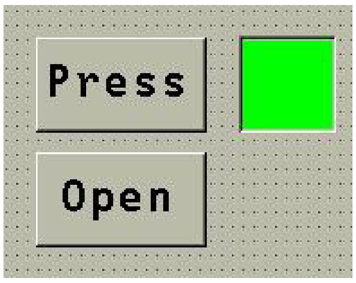


You can access this new control’s properties and customize them according to the relative purpose.



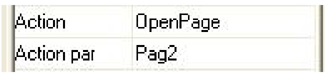
XPos	380
YPos	284
XDim	109
YDim	32
Name	Button_9
Text	Press
Text in selection	Press
Text color	■
Text color in selection	🟩
Text alignment	Center
Bitmap	
Bitmap in selection	
Image alignment	Center
Font	VeraSansMon...
Border style	3D
Border points	1
Border color	■
Background color	▨
Selection border	■
Sel. background	🟥
Selection order	0
Visible	TRUE
Press variable	*
Selection variable	FALSE
Type	Normal
Action	Call
Action par	

The preview looks like this.



The button control has got a very important attribute, which has not been represented in the properties grid above: the *Action* attribute allows the user to associate an action with the button pressure. Some actions require an additional parameter which you may specify in the *Action par* field.

In this case let us assume that you want that the pressure of the *Open* button opens the *Pag2* page. To obtain this select the *OpenPage* action in the *Action* field; then type the name of the child page *Pag2* in the *Action par* field.



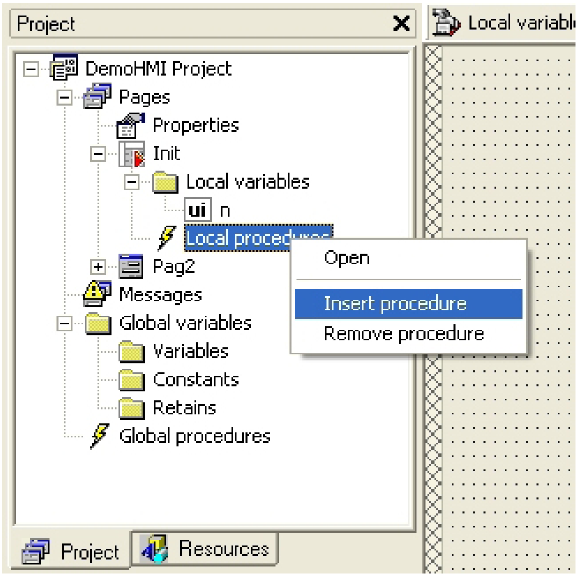
Action	OpenPage
Action par	Pag2

2.10.4 INSERTING A BUTTON AIMED AT LAUNCHING A PROCEDURE OF THE USER

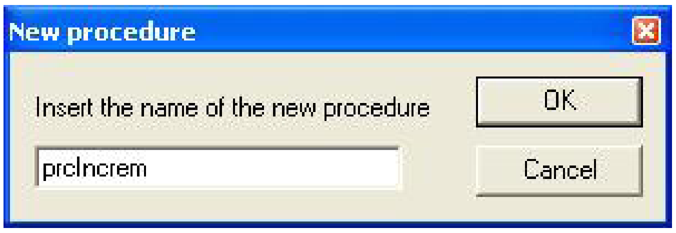
PageLab enables the user to implement some procedures (see 4.8.4) through which it is possible to customize the HMI behaviour: this feature makes PageLab projects very versatile.

Let us suppose that you want to create a procedure to increment the local variable *n* of the *Init* page. As this procedure applies on a local variable, it will be local in the *Init* page, too.

First of all create the procedure: expand the *Init* page tree, right-click on the *Local prcedures* item and select the *Insert procedure* command in the contextual menu which appears.

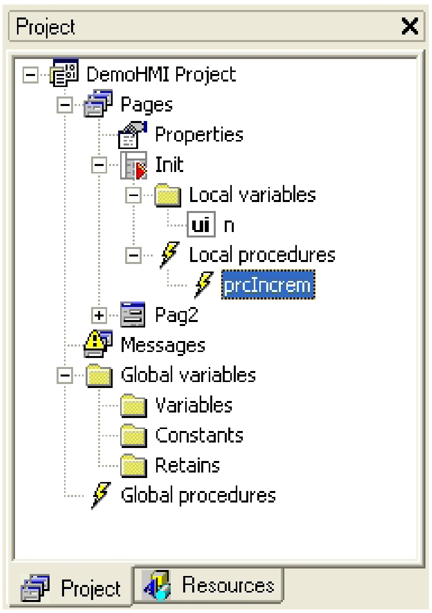


A little dialog window opens. The user is then required to type the new procedure's name. In this case, it may be *prcIncrem*.



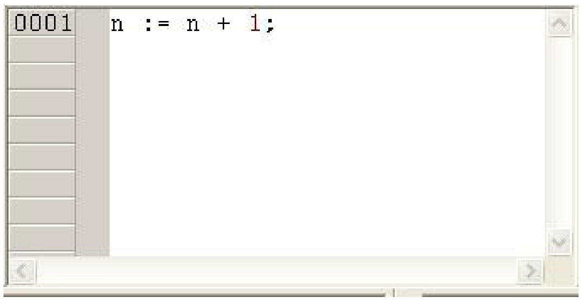
prcIncrem

Press the *OK* button. Then PageLab adds a new item in the page tree: it corresponds to the local variable which has been just declared.



Double-click on the above mentioned new item: the ST language editor opens and lets you either implement or edit the selected procedure's code.

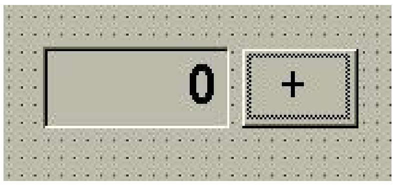
Write a procedure that applies a unit increment to the *n* variable.



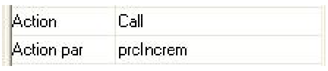
```
n := n + 1;
```

Then close the document.

Insert a new button beside the edit box associated with the *n* variable and type the character + in the `Text` property.



Let us suppose that you want to execute the `prcIncrem` procedure by clicking the + button: select the `Call` action in the `Action` field and type the procedure's name in the `Action par` field.



Action	Call
Action par	prcIncrem

Every time the user will press the + button when executing the HMI, *n* will increase by one and the edit box will show the up-to-date value.

2.11 VISIBILITY AND UPDATING OF CONTROLS

As stated in the previous paragraphs, each control has its own properties which the user may customize through the properties table fields.

Some of these features are specifically related to a single type of control. Others may be included in the properties set of different objects. The following paragraphs concern two important properties which are common to some kinds of control.

2.11.1 THE VISIBILITY PROPERTY

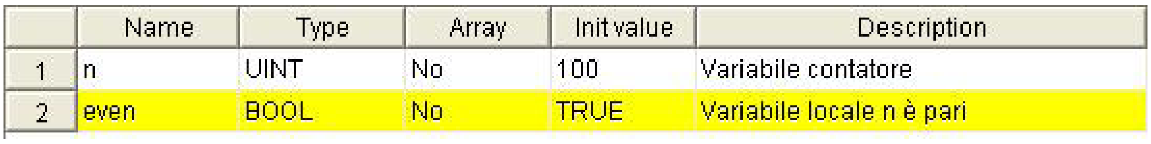
Almost all of controls are endowed with the `Visibility` property, which determines whether the object is visible or not. This property can be associated either with a constant value (*FALSE* = the control is always hidden, *TRUE* = the control is always shown) or a boolean variable, whose value dynamically establishes the visibility state.

By following the instructions in paragraph 2.7.1 you have inserted the string: `Stringa di testo` in the `Init` page. At present this string is always visible, as you can deduce from the assigned value to its `Visibility` property.

Visible TRUE

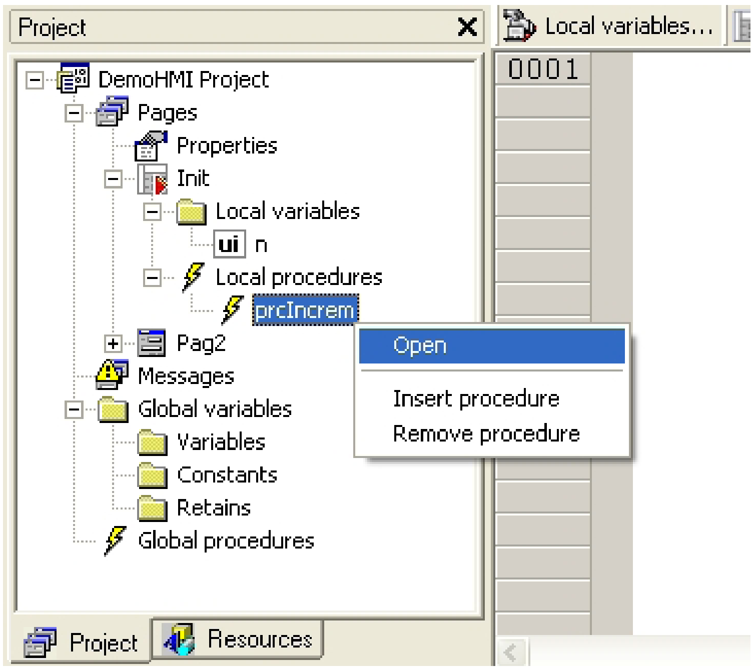
Let us assume that you want to assign this control's visibility to the local variable *n*, which is displayed in the edit box created in paragraph 2.9.2 and managed by the `prcIncrem` procedure, which was implemented in paragraph 2.10.4 and started up by the + button. More precisely, let us suppose that you want the text string visible when *n* is even, whereas hidden when *n* is odd.

To this purpose, it is necessary to declare a new boolean local variable which indicates whether at present *n* is even.

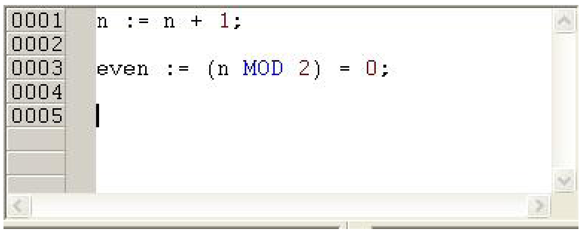


	Name	Type	Array	Init value	Description
1	n	UINT	No	100	Variabile contatore
2	even	BOOL	No	TRUE	Variabile locale n è pari

Then it is necessary to edit the `prcIncrem` procedure so that, when it refreshes the *n* value, it evaluates again whether it is even or odd. In order to access the `prcIncrem` source code, select the corresponding item in the project tree by right-clicking. Afterwards, choose `Open` from the contextual menu which appears.

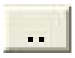


The ST language editor opens and the procedure's code may be extended as follows.

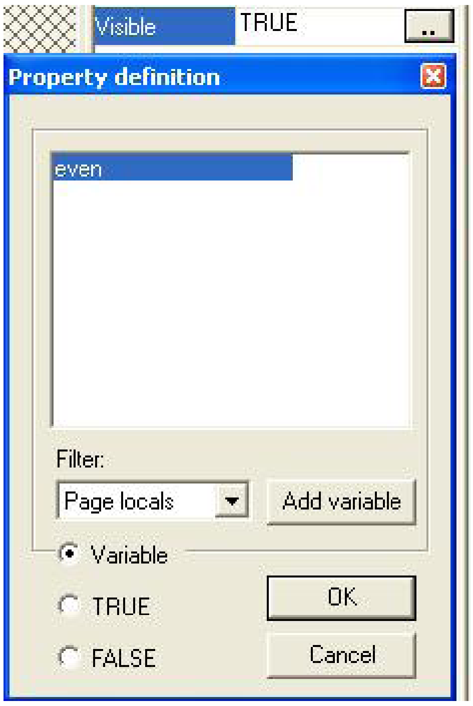


```
n := n + 1;

even := (n MOD 2) = 0;

```

In order to associate the string's visibility state with the *even* boolean variable, select the text string and click the *Visibility* field: a button appears.



After clicking it, a dialog box opens. Select the *radio button Variable*, which enables the overhead variables list; change the filter *Filter* into *Page locals* and select the only local boolean variable that is *even*.



Visible TRUE

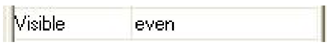
Property definition

even

Filter: Page locals

[✓ Variable] TRUE FALSE

Confirm your choice by clicking *OK.* The result is the following.



Visible even

2.11.2 THE REFRESH PROPERTY

When available, the *Refresh* property determines it the associated object has to be drawn once (when opening the page or coming back from a child page) or it needs to be constantly refreshed.

This property distinguishes, for example, the edit box and the text box.

With regard to the edit box, the refresh property has to be set when compiling and it cannot be edited at runtime. If you assign *Refresh = TRUE*, the associated variable's value is constantly read and refreshed, otherwise (*Refresh = FALSE*) the value is read and refreshed only when you open the page or when you come back from a child page.

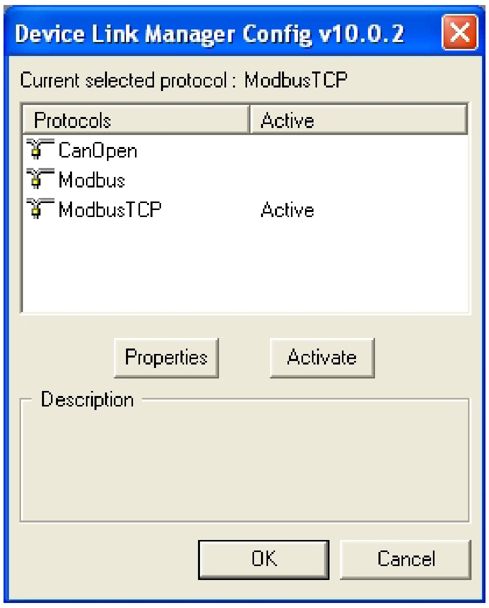
There is another option about text boxes: you can associate a boolean variable that is used as trigger for refresh: when the trigger variable becomes *TRUE*, the control's contents are refreshed then it is automatically reset by PageLab to *FALSE*.

2.12 COMPILING AND DOWNLOADING THE PROJECT ON THE TARGET

The following paragraph shows you how to compile and download a HMI project on the target board that runs PageLab.

2.12.1 CONNECTING TO THE TARGET

Launch the *Communication settings* command from the *Project* menu. This causes the following dialog window to open.



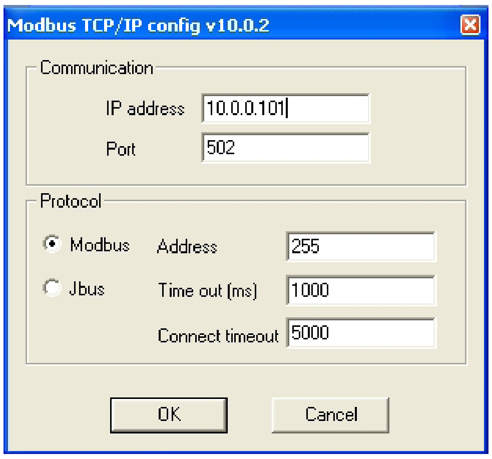
Device Link Manager Config v10.0.2

Current selected protocol : ModbusTCP

Protocols	Active
CanOpen	
Modbus	
ModbusTCP	Active

The user is required to select a suitable communication protocol from the left column and to activate it by pressing the *Activate* button.

Then the *Properties* button becomes active: by clicking it the user accesses another dialog window which is different in accordance with the specific selected control and lets set the protocol’s parameters. Let us consider the following example.



Modbus TCP/IP config v10.0.2

Communication

IP address: 10.0.0.101

Port: 502

Protocol

(● Modbus) ○ Jbus

Address: 255

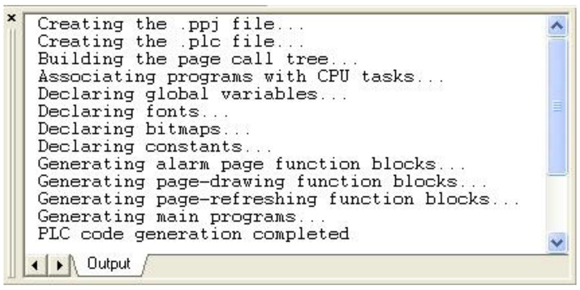
Time out (ms): 1000

Connect timeout: 5000

2.12.2 COMPILING PAGES FOR THE TARGET

You can start compiling the HMI project by clicking the corresponding button in the PageLab's `Project toolbar`.

Compilation is composed of two phases: the first one consists in the PLC code generation which realizes the pages as they have been planned in PageLab. The program shows in the `Output window` the progress level of the compilation and displays eventual errors.



```
Creating the .ppj file...
Creating the .plc file...
Building the page call tree...
Associating programs with CPU tasks...
Declaring global variables...
Declaring fonts...
Declaring bitmaps...
Declaring constants...
Generating alarm page function blocks...
Generating page-drawing function blocks...
Generating page-refreshing function blocks...
Generating main programs...
PLC code generation completed
```

The second one consists in the compilation of the PLC code which has been generated during the first phase. It can be started only if the first phase has been accomplished without any error.

This process is carried out by an external tool: the PLC command-line compiler `llc`, which PageLab automatically invokes with the suitable parameters.

2.12.3 DOWNLOADING AND EXECUTING THE COMPILED PAGES ON THE TARGET

At the end of the compilation, if all the phases have been successfully accomplished, you will see the downloading button become active in the `Project toolbar`

Clicking it, you activate again the PLC command-line compiler `llc`, which this time just downloads the compiled code in the target.

The downloading permission management depends on the implementation of the on board firmware. Consequently it changes according to the destination target of the download.

2.12.4 SIMULATION

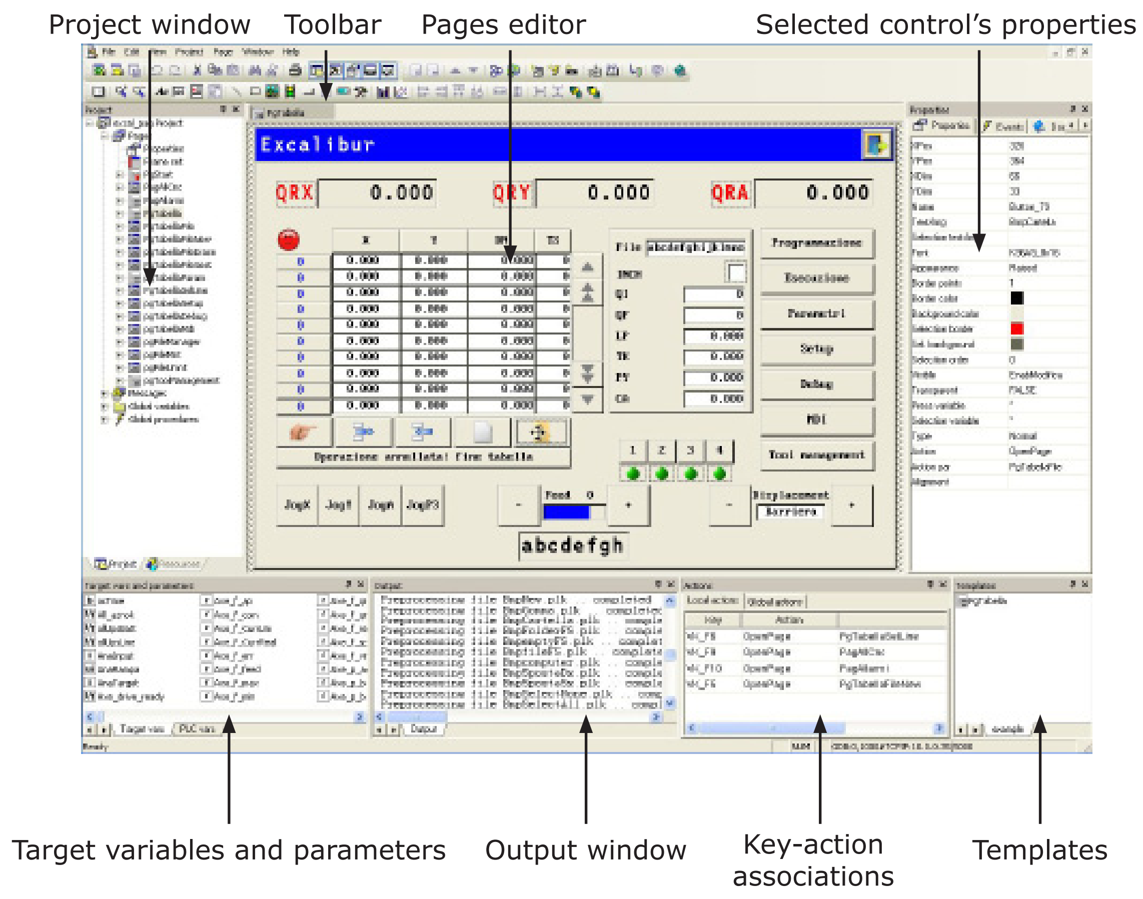
Depending on the target device you are interfacing with, you may be able to simulate the execution of the HMI application with PageLab's integrated simulation environment: SimuLab.

In order to start the simulation, just click on the appropriate item on the Project toolbar.

Refer to SimuLab's manual to gain information on how to control the simulation.

3. PAGELAB LAYOUT

The following picture shows you the layout and the essential elements of PageLab.



3.1 PROJECT WINDOW

This window includes two pages which are alternatively selectable by pressing the corresponding tab:

- Project: it shows the project tree and all the objects the project is composed of, hierarchically arranged. The pages node contains the project properties and the single pages. Each page contains the list of the local variables (visible and usable only in the page where they are declared) and the local procedures, which can be invoked only from the page where they are implemented. Moreover there is the node of the asynchronous messages, the node of the global variables (visible and usable from whatever page) and the node of the global procedures which you can invoke from whatever page.
- Resources: it shows the project resources, that is fonts, bitmaps, strings table, enumerated data types, images lists, and sets.

3.2 EMBEDDED EDITORS

PageLab is endowed with three types of editor:

- Pages editor: in order to open this editor, double-click the name of the desired page in the project tree (see 3.1). This tool shows you a page preview and lets you edit it: you may either add or remove controls (see 4.4), customize properties, manage the events and the documentation.

- Variables editor: by double-clicking on a local or global variable in the project tree, you can see respectively the declaration table of the local variables or the global variable table.
- Procedures editor: it allows the user in implementing procedures to be associated to the events which are defined for the various project's objects (pages and controls) or generated from the user himself (see 4.8).

3.3 PROPERTIES WINDOW

Each time you select an object in the pages editor, the properties window automatically refreshes and shows the selected object's properties and events.

This window is composed of many pages which you may select alternatively by pressing the corresponding tag above.

- Properties: it shows a table including the selected object's properties either it is a whole page or it is a page's control. The user is enabled to customize this values through the right-hand column of the table.
- Events: it shows a table including the typical events of the currently selected object. The user may associate either a local or variable procedure with each event by typing its name on the corresponding row of the event in the right-hand column.
- Doc: it displays a table which shows the *Description* field of the currently selected object. The user may describe the object and this description will be included in the automatic documentation management (see 4.10).

3.4 TOOLBARS

The user can give commands to PageLab through some useful toolbars. A toolbar can be defined as a collection of buttons which you may enable by left-clicking them and whose functions are intuitively represented by their icons.

The toolbars support tooltips, too. A tooltip is a small text frame containing a short description of the object which PageLab automatically displays when you hover with the mouse over a button.

PageLab is endowed with three essential toolbars:

- Main toolbar: it contains the commands to open and save the project, to cancel/restore the last changes, to print, to display or close other toolbars.
- Project toolbar: it allows you to add new elements to the project as variables, pages, events, actions, as well as to enable or prevent the simulation mode and to compile and download the whole project.
- Page toolbar: it allows you to choose a new type of control to be inserted in the active page, to align or equally space several controls, or to set the vertical order of the elements on the page.

3.5 THE OUTPUT WINDOW

PageLab prints in this window some messages which indicate the progress and the output of the requested processes: opening and compilation of a project, resources importing/exporting, etc..

3.6 TARGET VARIABLES AND PARAMETERS

This window shows the list of external variables, available for PageLab coding.

The window is composed of several pages which you may alternatively select by pressing the corresponding tab. One page contains the list of the available variables (file *.tgt*), another page contains the list of the variables which have been exported from the PLC LogicLab (file *.exp*). Other pages are optional, as many as the number of devices with external parameters linked to the project.

3.7 TABLE OF KEYS-ACTIONS ASSOCIATIONS

This table takes primary importance in case of traditional keyboards without touchscreen where the user interact with the system by pressing the relevant keys.

See in paragraph 4.8.5 the list of actions which may be associated to keys.

4. HMI PROJECT IN PAGELAB

PageLab manages the creation (development) of pages for a specific application as projects.

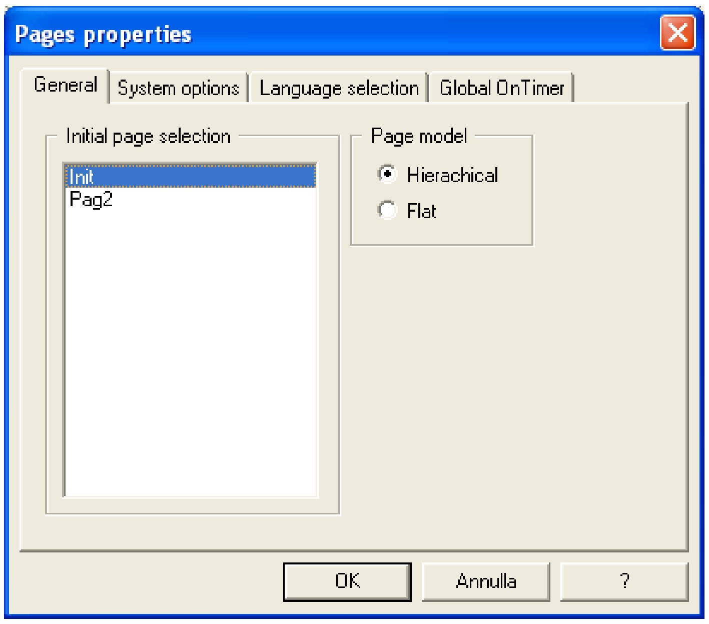
The PageLab project is composed of several pages where the user may arbitrarily arrange the controls.

In each PageLab project you have to specify the start page which will be displayed at the start of the system. Other pages will have at least a parent page from which they will be invoked and may have child page to invoke. The invoking/invoked relations implicitly give to the whole project a tree structure.

4.1 PROJECT PROPERTIES

In the project tree, click the `Pages` item and access the `Properties` item. By double-clicking the `Properties` item you open a dialog window which is composed of four pages. The following paragraphs show you the features of these pages.

4.1.1 GENERAL



Initial page selection: Init, Pag2 (Init selected)

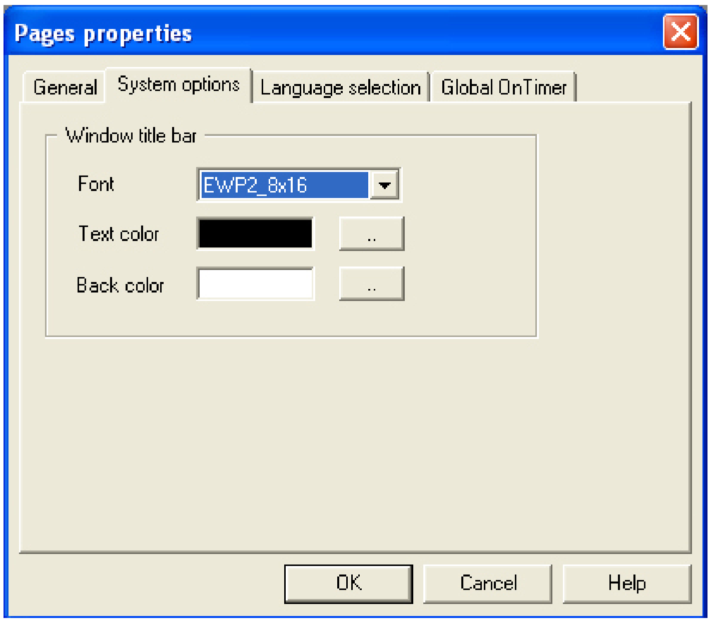
Page model: [✓ Hierachical] Flat

It allows to select the PageLab project's start page among the implemented pages.

The *Page Model* feature allows to select the type of page model in case of a page calls another page. If the model is hierarchical then a child page cannot recall a parent page. Instead if the model is flat all the pages can call the others without limitations.

4.1.2 SYSTEM OPTIONS

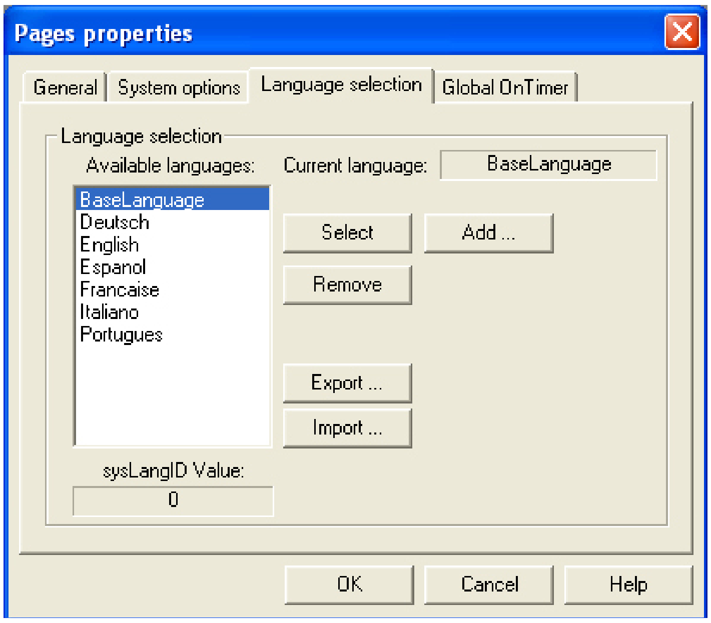
It allows the user to customize the window's title bar features: the font, the text color and the background color.



Window title bar

Font	EWP2_8x16
Text color	■
Back color	□

4.1.3 LANGUAGE SELECTION



Language selection

Available languages:

- BaseLanguage
- Deutsch
- English
- Espanol
- Francaise
- Italiano
- Portugues

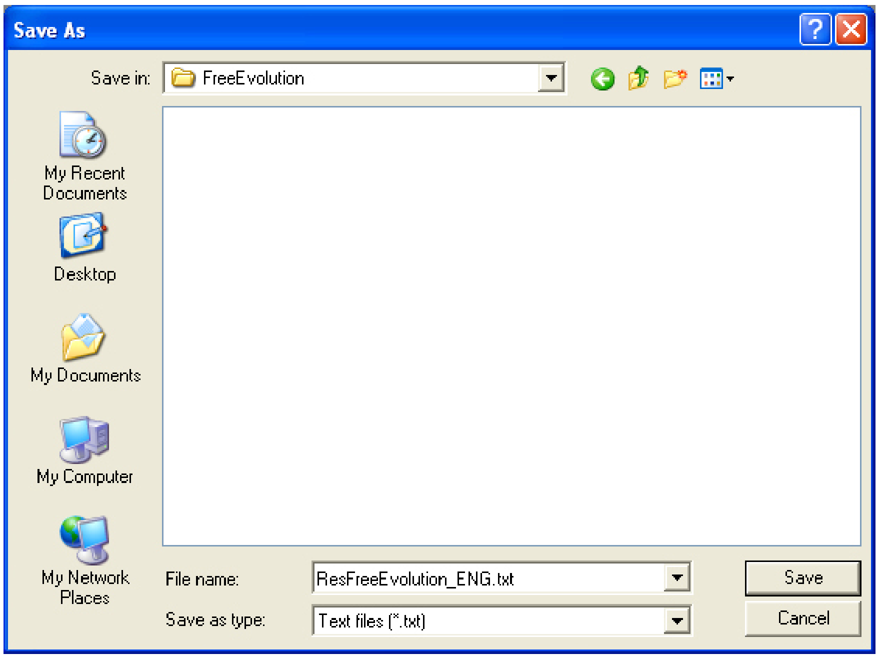
Current language: BaseLanguage

sysLangID Value: 0

It allows you to add, remove, export, import, and select the resources languages (see 4.9). The label: `sysLangID Value` indicates the value which the `sysLangID` target variables must take to display the pages in the selected language.

In order to add a language, apply the following procedure.

First of all export the language supported by the translator, choose Italian and press the *export...*button, which opens a window requiring the destination folder for the selected language file.

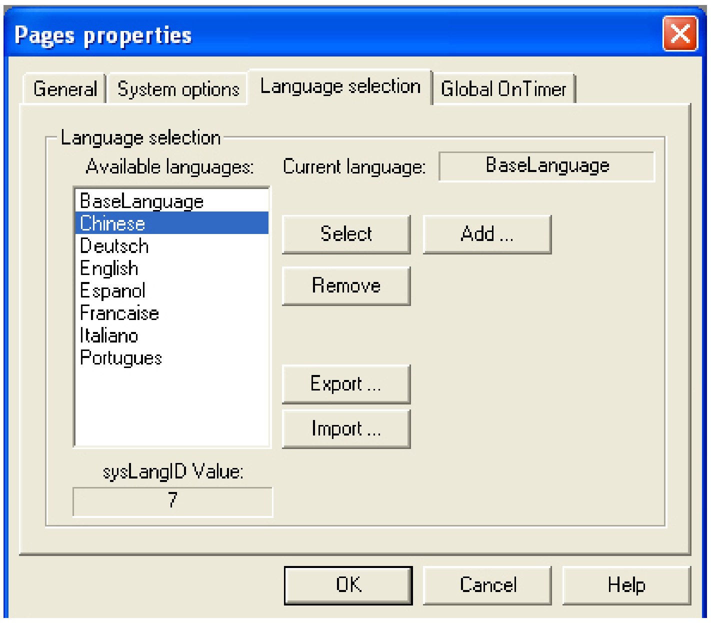


The program suggests a file name: `Res` + project title+'_' + first three characters of the language + extension`.txt`. At the end of the exportation the file is composed of all the project's resources which have to be translated:

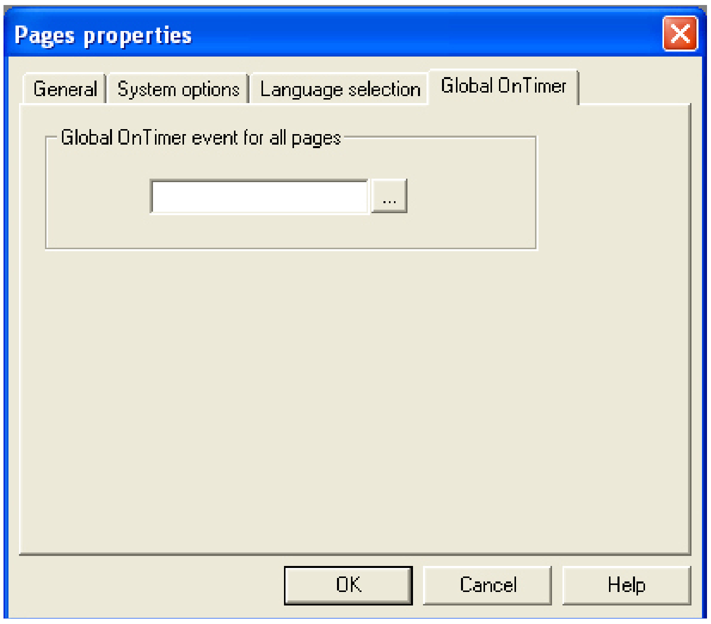
- strings
- enumeratives

Translate the file and replace the text under the `Language` tag with the one of the new language (for example, in this case change it into `Chinese`). In the `Language selection` panel choose the `Import...` button, then select the suitable file in the PC.

The new language appears in the list.

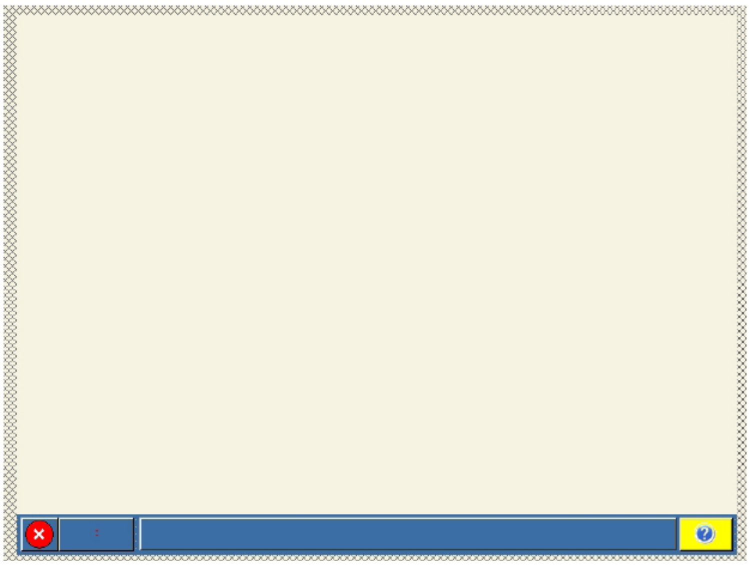


4.1.4 GLOBAL PERIODIC PROCEDURE



Global on timer allows you to specify the name of a global procedure to be periodically and independently executed on the active page. Such a procedure may be effectively used to constantly test one or more PLC variables and to emit alarm messages, for example through asynchronous messages (see 4.3.4).

4.2 FRAME SET



PageLab allows to define areas which are called frames and are placed on the sides of the screen and are always active.

The user may set these frames' dimensions and insert some controls which are active whatever the currently loaded page. Consequently frames are useful to host the objects which have to appear in the whole project. In this way the user does not need to duplicate them in each page.

As regards to the above, there are two exceptions: the pop-up pages (see 4.3.3) when the `Modal` property is set to *Yes* and all the asynchronous messages. When these pages are active, the controls of the frame set are automatically disabled.

4.3 PAGES

4.3.1 NAVIGATING BETWEEN PAGES

PageLab manages pages development for a specific application as projects.

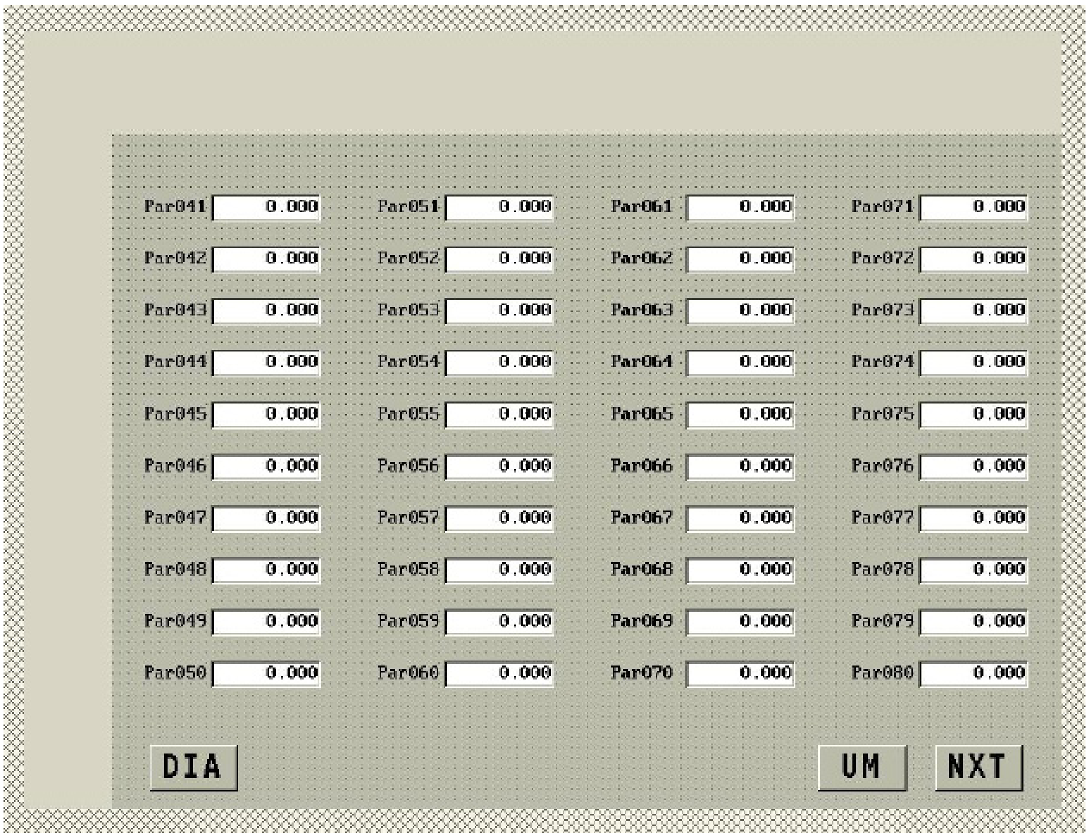
PageLab project is composed of pages where the user can arbitrarily arrange controls.

In each PageLab project it is necessary to define a start page which will be viewed at system startup. Other pages must have at least a parent page from which they are invoked and may have child page to invoke. The invoking-invoked relations of the pages give the whole project, even though in an implicit way, a multi-node tree structure.

A child page may be invoked in two ways:

- Through an action associated to a key: associate an *OpenPage* action with a physical key (if there is a keyboard) or with a virtual key (whose pressure is an event raised by software);
- Through an action associated with a button: insert in the parent page a *Button* control (see 4.4.7) and specify in the *Action* property that by pressing it the child page opens.

4.3.2 CHILD PAGES



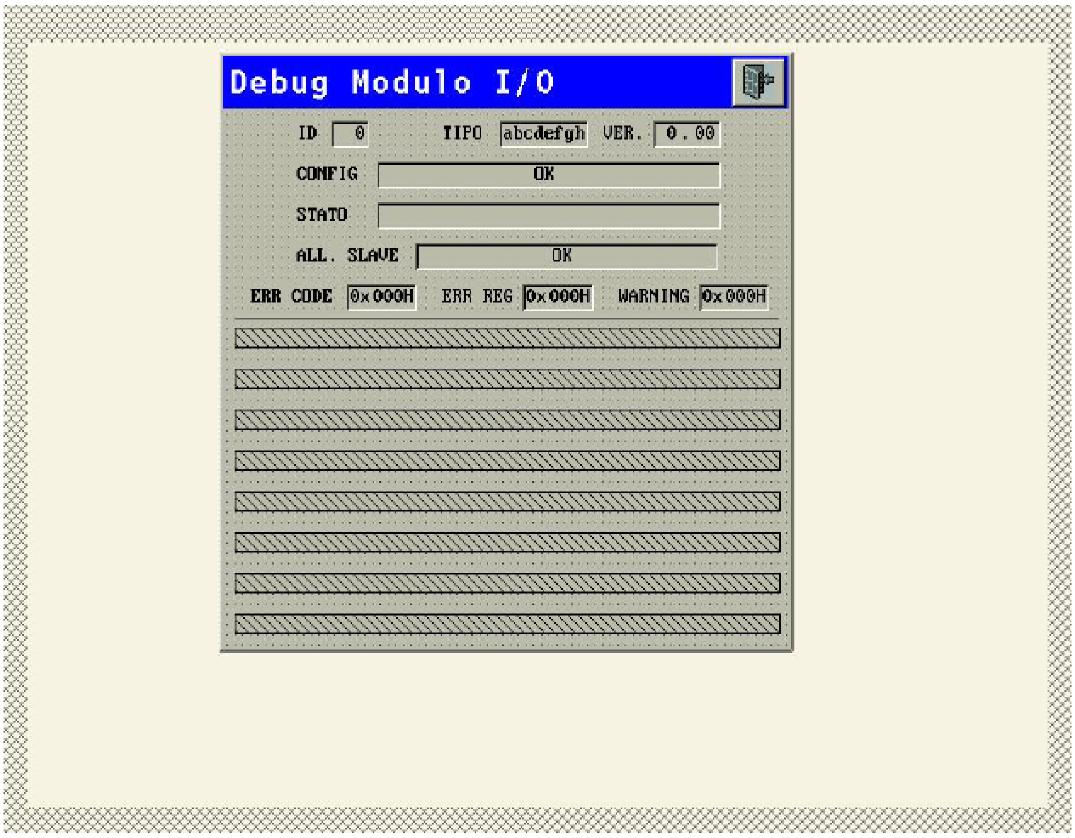
Par041	0.000	Par051	0.000	Par061	0.000	Par071	0.000
Par042	0.000	Par052	0.000	Par062	0.000	Par072	0.000
Par043	0.000	Par053	0.000	Par063	0.000	Par073	0.000
Par044	0.000	Par054	0.000	Par064	0.000	Par074	0.000
Par045	0.000	Par055	0.000	Par065	0.000	Par075	0.000
Par046	0.000	Par056	0.000	Par066	0.000	Par076	0.000
Par047	0.000	Par057	0.000	Par067	0.000	Par077	0.000
Par048	0.000	Par058	0.000	Par068	0.000	Par078	0.000
Par049	0.000	Par059	0.000	Par069	0.000	Par079	0.000
Par050	0.000	Par060	0.000	Par070	0.000	Par080	0.000

Let us assume that you want to add a page to a project.

PageLab displays a dialog window which requests to insert the name you intend to assign to the new page. This dialog window contains a checkbox with the label: *Pop-up*. If you do not select it, the new page will be a child one.

A child page fits the whole screen or, alternatively if there are defined frames (see 4.2), it fits the free remaining area. Consequently, the user cannot define position and dimensions of a child page as they are automatically set according to the screen and the frame set.

4.3.3 POP-UP PAGES



Debug Modulo I/O

ID 0 TIPO abcdefgh VER. 0.00

CONFIG OK

STATO

ALL. SLAVE OK

ERR CODE 0x000H ERR REG 0x000H WARNING 0x000H

When creating a new page, if the user selects the aforesaid checkbox with the `Pop-up` label, the new page will be a pop-up one.

There are no restrictions about position and dimension. In fact the user may superimpose a pop-up page on the frames: when activating this page, if it is not modal (property: *Modal*), the controls superimposed on the open page will be disabled; otherwise, all the controls will be inactive.

4.3.4 ASYNCHRONOUS MESSAGES

Asynchronous messages are similar to standard pages, except the following features:

- They have an additional property, that is the identifier of the associated message (*Msg ID*).
- They cannot contain invocations to child pages.
- They have no defined parent page nor a tree structure (see Introduction 4.), but they can be invoked from any other standard page.

An asynchronous message cannot be explicitly invoked; the system displays it whatever the active page when it intercepts a message containing the corresponding *Msg ID*. This message may be launched either by the firmware or by a procedure through the *Video_SendMessage* function (see 8.1.8) by using the following syntax:

```
Video_SendEvent( kWM_MSG, Msg ID );
```

4.4 CONTROLS

A control is a display element which is contained in a page. The following paragraphs shows you the controls which PageLab supports.

4.4.1 STATIC

It displays a fixed string, whose contents cannot be edited when executing. In fact, you should specify the text of the string directly or by the association of the ID of a string defined as resource to support multi language management. For project resources and multi language support see paragraph 4.9.

In order to insert a *Static* control, press the corresponding button in the *Page toolbar*.

Then click the point where you want to insert the control.

You can get information on properties and events of the *Static* control in paragraph 5.4.

4.4.2 GRAPHIC ELEMENT

It displays a static line or rectangle. This means that their properties cannot be edited when executing.

In order to insert a *Line* control, press the corresponding button in the *Page toolbar*.

Then click the point where you want to insert the control.

In order to insert a *Rectangle*, press the corresponding button in the *Page toolbar.*

Then click the point where you want to insert the control.

You can get information on properties and events of the *Line* and *Rectangle* controls in paragraphs 5.5-5.6.

4.4.3 EDIT BOX

It displays the contents of an associated variable.

In order to insert an *Edit box*, click the corresponding button in the *Page toolbar.*

Then either click the point where you want to insert the control or drag a variable from the project tree or from the library window.

You can get information on properties and events of the *Edit box* control in paragraph 5.7.

4.4.4 TEXT BOX

It displays the contents of an associated string variable. It supports the formatting on several lines of the text which is contained in the string.

To insert a `Text box` control in the page, press the corresponding button in the `Page toolbar`.

Then either click the point where you want to insert the control or drag a variable from the project tree or the library window.

You can get information on properties and events of the `Text box` control in paragraph 5.8.

4.4.5 IMAGE

It displays a bitmap image.

In order to insert an `Image` press the corresponding button in the `Page toolbar`

Then click the point where you want to insert the control.

You can get information on properties and events of the `Image` control in paragraph 5.9.

4.4.6 ANIMATION

It displays a bitmap image which you select from a list of images depending on the value of an associated selection variable.

In order to insert an `Animation` press the corresponding button in the `Page toolbar`.

Then click the point where you want to insert the control.

You can get information on properties and events of the `Animation` control in paragraph 5.10.

4.4.7 BUTTON

You may use the `Button` control either to check a boolean variable's state or (press= `TRUE`, release = `FALSE`) or to send a command to the system.

In order to insert a `Button` press the corresponding button in the `Page toolbar.`

Then either click the point where you want to insert the control or drag a boolean variable from the project tree or the library window.

You can get information on properties and events of the `Button` control in paragraph 5.11.

4.4.8 CHART

Chart control draws the static diagram of one or more arrays of values associated.

In order to insert a *Chart* control, click the corresponding button in the *Page toolbar.*

Then click the point where you want to place the control.

You can get information on properties and events of the *Chart* control in paragraph 5.14.

4.4.9 TREND

After assigning up to 8 numerical variables, the object will automatically and periodically (once every a defined time) acquire their values and will draw the corresponding graphic in a dynamic and automatic way.

In order to insert a *Trend* control press the corresponding button *Page toolbar*.

Then click the point where you want to insert the control.

You can get information on properties and events of the *Trend* control in paragraph 5.15.

4.4.10 PROGRESS BAR

It represents the progress of an operation by showing a stained bar in a horizontal or vertical rectangle. The length of the bar, related to the bar's lenght, shows the percentage of the completed operation.

In order to insert a *Progress bar* control press the corresponding button in the *Page toolbar*.

Then click the point where you want to place the control.

You can get information on properties and events of the *Progress bar* control in paragraph 5.12.

4.4.11 COMBO BOX

Shows a list of strings connected to a variable with an enumerator element. In order to insert a Combobox, click the corresponding button in the Page toolbar.

Then either click the point where you want to insert the control or drag a variable from the project tree or from the library window. You can get information on properties and events of the Combobox control in paragraph 5.16.

4.4.12 CHECKBOX

Displays a check box that allows the user to select a true or false condition indetified by a variable.

In order to insert an Checkbox, click the corresponding button in the Page toolbar.

4.4.13 CUSTOM CONTROL

This control is implemented in the firmware. You can have several types of custom controls which are marked by the `Control ID` property and each type of control may have several instances.

In order to insert a *Custom control*, press the corresponding button in the *Page toolbar.*

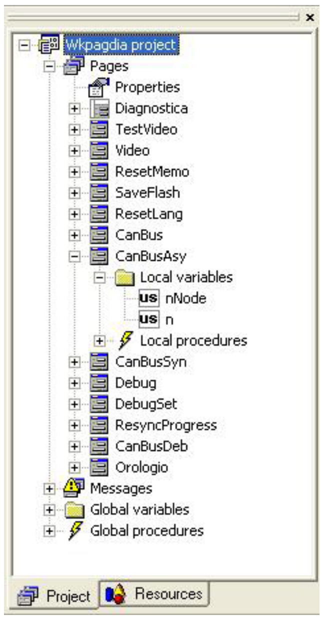
Then click the point where you want to insert the control.

You can get information on properties and events of the *Custom control* in paragraph 5.13.

4.5 VARIABLES

In a PageLab project there are different classes of variables. The following paragraphs show you their features.

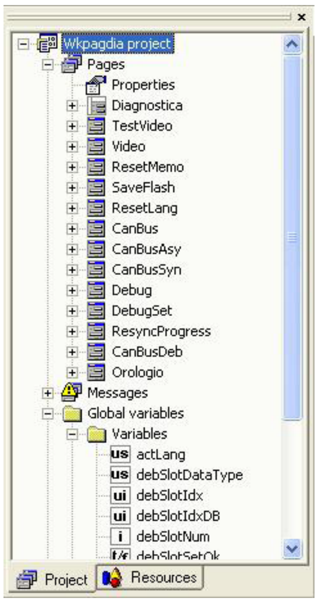
4.5.1 LOCAL VARIABLES



Local variables are variables of the PageLab project. You can access them only through the page they were declared from.

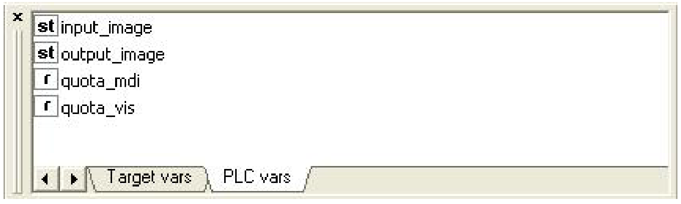
They are listed in the project tree, under the *Local variables* folder. Local variables can be used to carry out operations on PLC (for example to apply a different scale or to add an offset) or system variables or to implement local procedures.

4.5.2 GLOBAL VARIABLES



Global variables are declared in PageLab and they are accessible from every page of the project. Global variables are listed in the `Global variables` folder in the project tree. The function of the global variables is similar to the local variable's one but the different visibility scope makes them unusable for the implementation of global procedures or for the parameters passing between distinct pages.

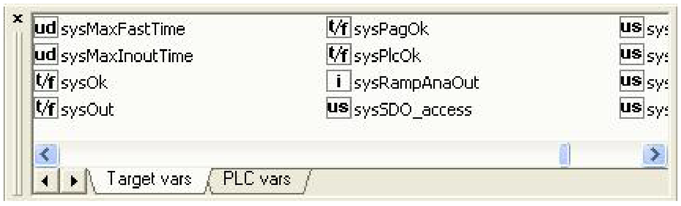
4.5.3 VARIABLES IMPORTED FROM PLC



A compiled PageLab project consists in a PLC that, once downloaded on the target board, is executed by the actual PLC., which is implemented with LogicLab. Variables exported from the LogicLab PLC contained in the *.exp* file enable the interaction between these two distinct components. PLC variables which are not automatic, thus associated to a data-block are exported in the *.exp* file.

In order to include a *.exp* file in the PageLab project, press the `Link PLC variables file` option from the `Project` menu, then search for the file in the PC resources. After linking a *.exp* file to the PageLab project, you can get a list update of the exported variables by selecting the `Refresh PLC variables` option from the same menu.

4.5.4 SYSTEM VARIABLES



The interaction between PageLab and target is enabled by system variables which the software publishes outside in a *.tgt* file.

You may access system variables in read/write or in read-only mode; if you try to access a read-only variable in write mode, an error will occur when compiling.

4.6 MULTIPLE PAGES MANAGEMENT

These functions allow to construct pages with data of different kind that must be represented on distinct pages for space reasons.

Sets (see 4.9.6) can be used with edit boxes or progress bars. Sets are ensemble of variables even of different type. Set definition can be done from the resource tree, they are implemented using a table with a series of variables that are dynamically associated to the control basing on the current index assigned to the page.

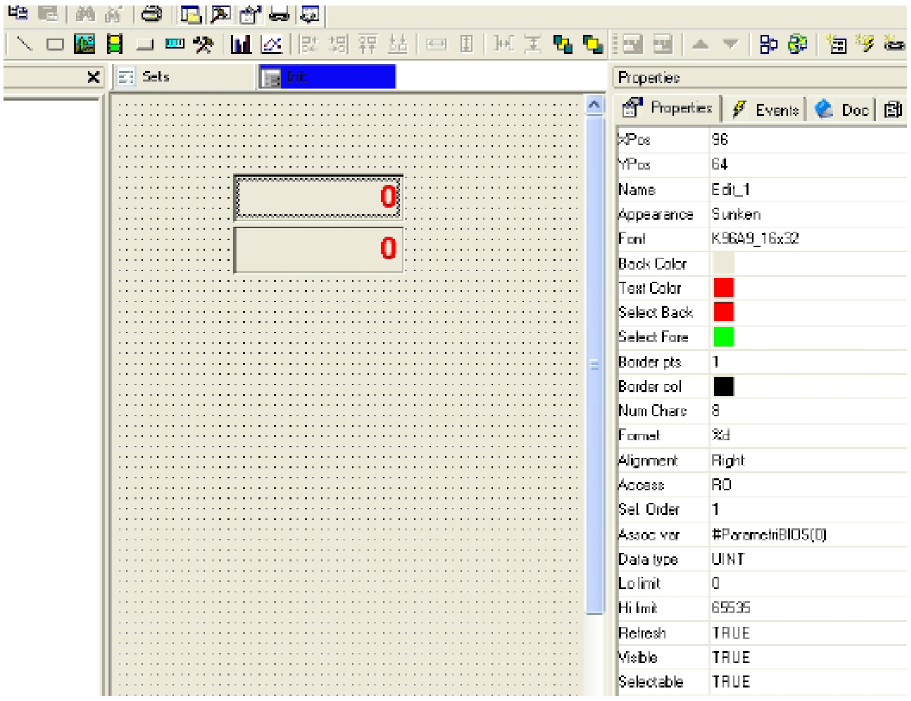
Let us see how to use a set.

4.6.1 ASSOCIATION OF ELEMENTS OF A SET

Elements of a set can be associated to a control using the following syntax: character # first of all, then the name of the set followed by the index of the position of the element in the page, between round brackets.

Position index is used to indicate the order in which elements are shown in case of more than one element in the same page.

A page contains one or more controls based on one (or more) set. At runtime, the page is replied in order to show all the elements contained in the set. In the last page, if any control cannot be filled with element value, that control is hidden.



XPos	96
YPos	64
Name	Edit_1
Appearance	Sunken
Font	K96A9_16x32
Back Color	⬜
Text Color	🟥
Select Back	🟥
Select Fore	🟩
Border pts	1
Border col	⬛
Num Chars	8
Format	%d
Alignment	Right
Access	RO
Sel. Order	1
Assoc var	#ParametriBIOS(0)
Data type	UINT
Lo limit	0
Hi limit	65535
Refresh	TRUE
Visible	TRUE
Selectable	TRUE

We created before a set of five elements named *BIOSParameters*, now we can associate *#BIOSParemeters(0)* to the first edit box and *#BIOSParemeters(1)* to the second. So there are three pages: first page with the first two elements of the set, second page with elements 2 and 3 and third page showing the last element of the set. In the last page the second edit box is not visible.

4.6.2 NAVIGATION OF THE ELEMENTS OF A SET

Navigation of pages that represent a set of elements is automatically done using the *NextEdit* event of the last selectable control of the page and using *PrevEdit* event of the first selectable control of the page.

It is also possible to send special events to force the change of the page in this way:

```
Video_SendEvent( kEV_WM_CHANGESETPAGE, numpage );
```

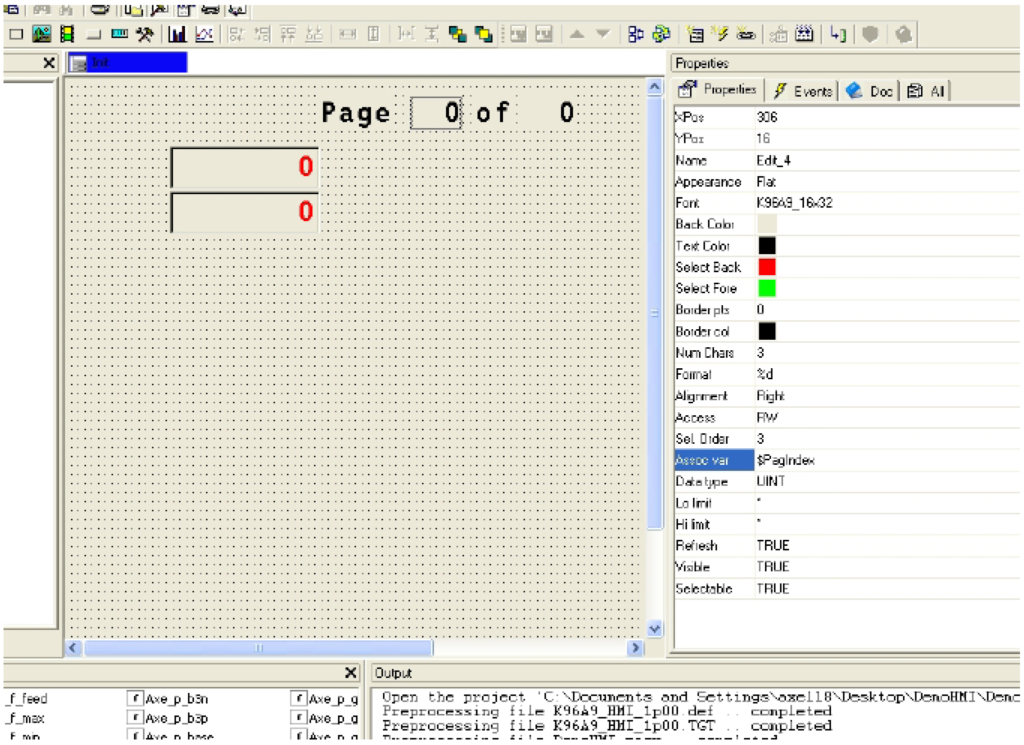
Where `numpage` is the number of the page of the set.

4.6.3 PAGES NUMBERING

PageLab defines two variables related to pages numbering:

- *$PagIndex* = current index of the page containing controls based on a set;
- *$PagNumber* = number of pages that complete the visualization of the whole elements of the set.

These variables can be used in the page to show the numeration of the pages. In fact they can be used as variables associated to edit box controls in this way:



XPos	306
YPos	16
Name	Edit_4
Appearance	Flat
Font	K96A9_16x32
Back Color	⬜
Text Color	⬛
Select Back	🟥
Select Fore	🟩
Border pts	0
Border col	⬛
Num Chars	3
Format	%d
Alignment	Right
Access	RW
Sel. Order	3
Assoc var	$PagIndex
Data type	UINT
Lo limit	*
Hi limit	*
Refresh	TRUE
Visible	TRUE
Selectable	TRUE

```
Open the project 'C:\Documents and Settings\axel18\Desktop\DemoHMI\Demo
Preprocessing file K96A9_HMI_1p00.def .. completed
Preprocessing file K96A9_HMI_1p00.TGT .. completed
```

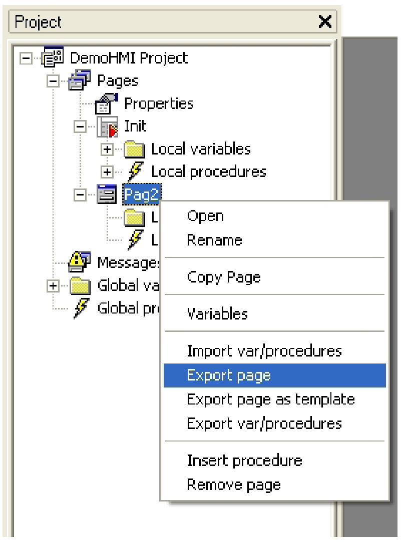
4.7 ADVANCED OPERATIONS ON PAGES

Advanced operations such as export/import, copy/paste and page based template management can be done with PageLab. Next paragraphs show these arguments in details.

4.7.1 EXPORT/IMPORT OF PAGES TO/FROM FILES

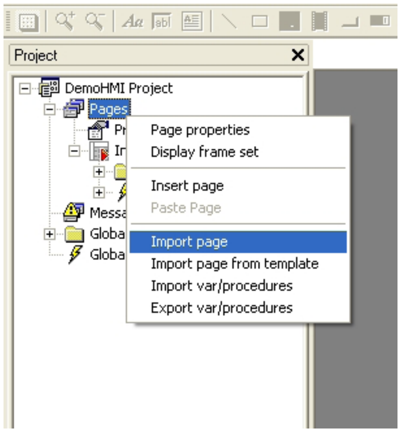
Each page, even if of a certain complexity, can be saved to be used later in other projects.

To do so click with the right button on the page node in the project window then select *Export page* from the menu:



Next, application asks user to insert the name of the file in which the page will be saved. This file assumes a *.pex* extension. Export file contains page info and local procedures.

Import operation is quite similar to the export operation. Select *Pages* node, click with right button and select *Import page* from the popup menu. User can then select the file of the page to import. Imported page takes the same name that it had when it was exported.

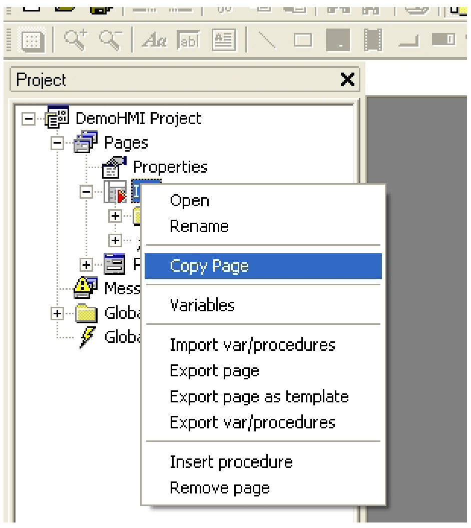


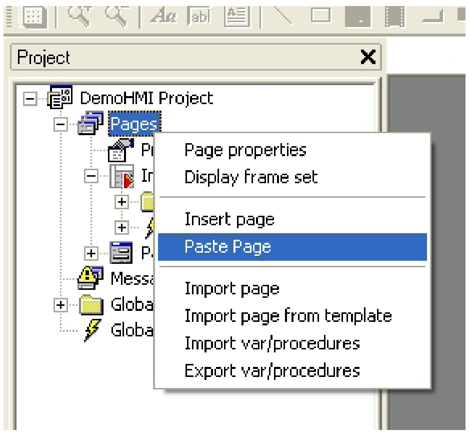
4.7.2 EXPORT/IMPORT PROCEDURES AND VARIABLES

It is also possible to export/import local or global variables and procedures using the menu commands *Export var/procedures* or *Import var/procedures*.

4.7.3 COPY/PASTE OF PAGES IN THE PROJECT

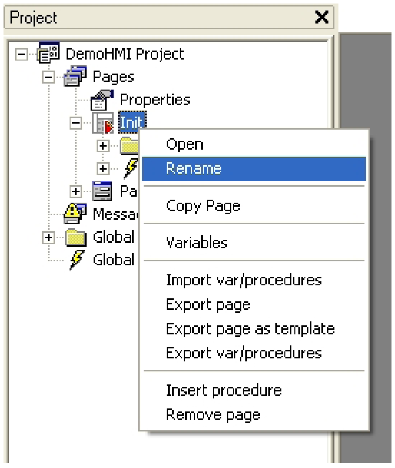
It is possible to copy and to paste a page inside the project. Select desired page, click with right button of the mouse and select *Copy Page* from the menu. Then, to paste page copied, select *Pages* node and select *Paste Page* from the menu.





4.7.4 RENAME PAGES

Select desired page from the project tree then click with the right mouse button and select *Rename* from the menu. This allows the user to change the name of the page.



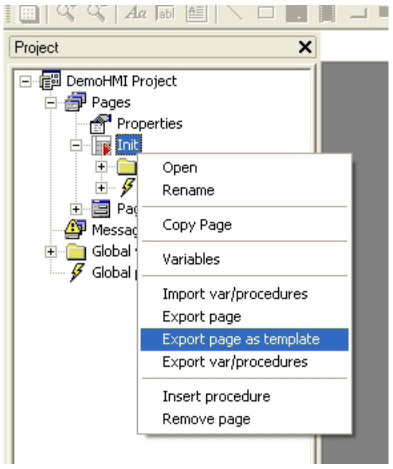
N.B.: this operation changes only the name of the page, project references to the renamed page are not automatically updated.

4.7.5 TEMPLATES OF PAGE MANAGEMENT

Templates allow the user to save only the skeleton of the page and not the whole page. Templates can be described as pages without references to external variables. Templates can be grouped in libraries files (*.petx*) and can be linked into the project.

4.7.5.1 EXPORT PAGES INTO A TEMPLATE FILE

To export a page into a library of templates follow the procedure paragraph 4.7.1 initial steps then select `Export page as template` from the menu.



A library file with `.petx` extension (new or already existing) should be indicated. Template is appended to the existing templates and a name for the library is requested. If the template is already available in the library a message asks the user if he desires to rewrite the existing template or not.

Page is exported as template into the specified library with all its element but without any referenced variable.

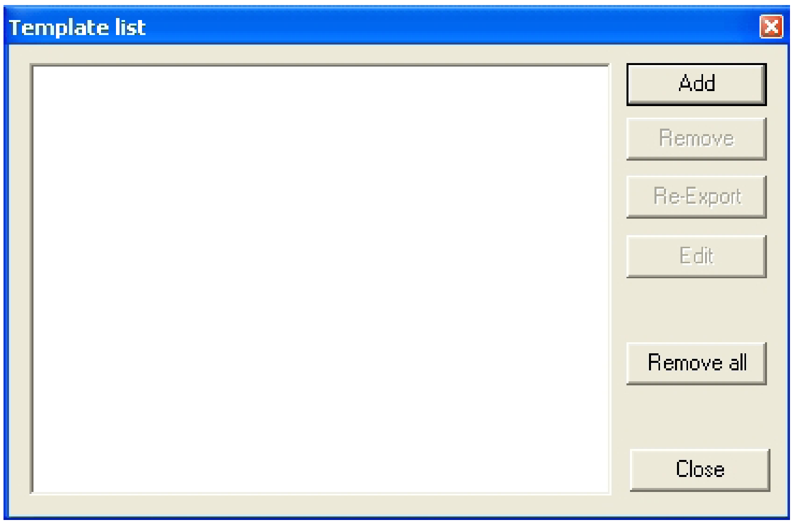
Scripts and local variables are exported without changes. References to variables contained in the scripts are not modified.

Child pages, popup and asynchronous messages can be treated as templates.

4.7.5.2 USAGE OF THE TEMPLATE LIBRARY IN A PROJECT

It is possible to include a template library in a project in order to use templates when desired.

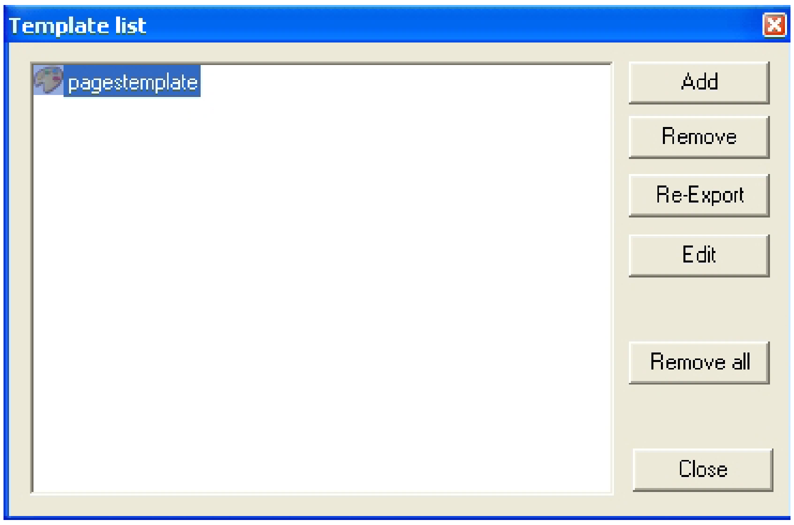
Select `Template Management` menu voice from `Project` menu. Following window will be shown:



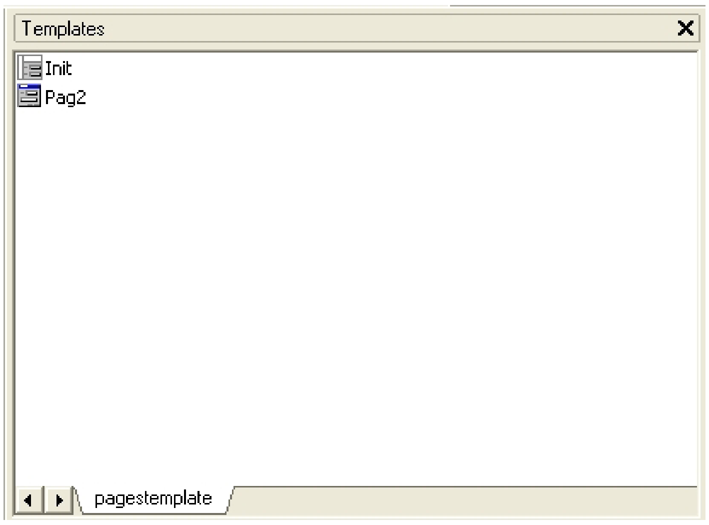
Available operations are listed here:

- Add: add a template library to the project. Including a library means that a reference to the library's *.petx* file is added to the current project, and that a local copy of the library is made.
- Remove: Remove template library from current project.
- Edit: To modify local copy of the template library removing no more used templates.
- Re-export: Export local copy of the template library into a new *.petx* library file.
- Remove All: Remove all template libraries from current project.

Now press the *Add* button to add a template library to the project. Once chosen one of the available libraries, *Template list* window appears as shown here.



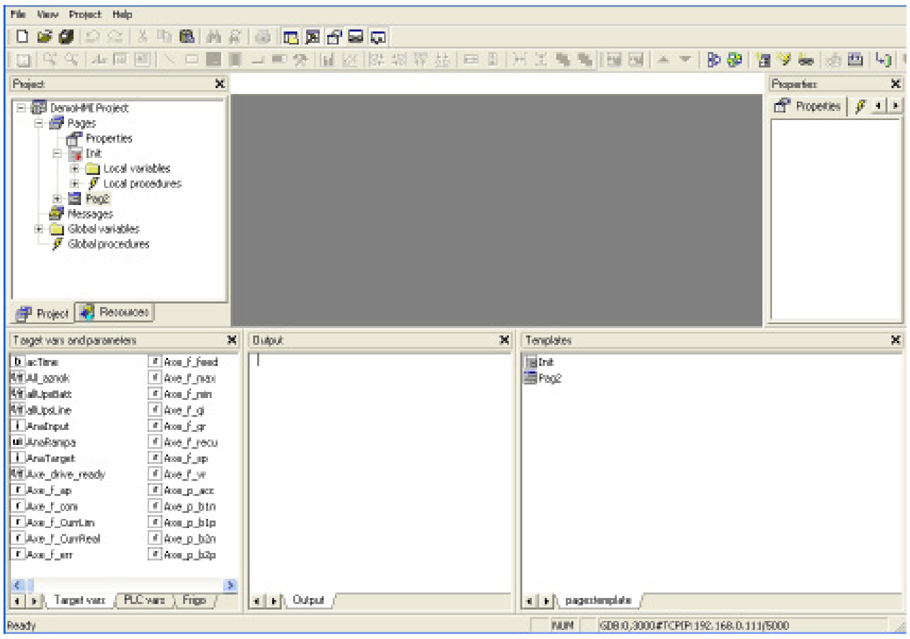
Template library has been included to the project. Press *Close* button, *Templates* window is shown: there is a tab for each library imported in current project.



Each tab shows the list of templates of the corresponding library.

4.7.5.3 USING A TEMPLATE

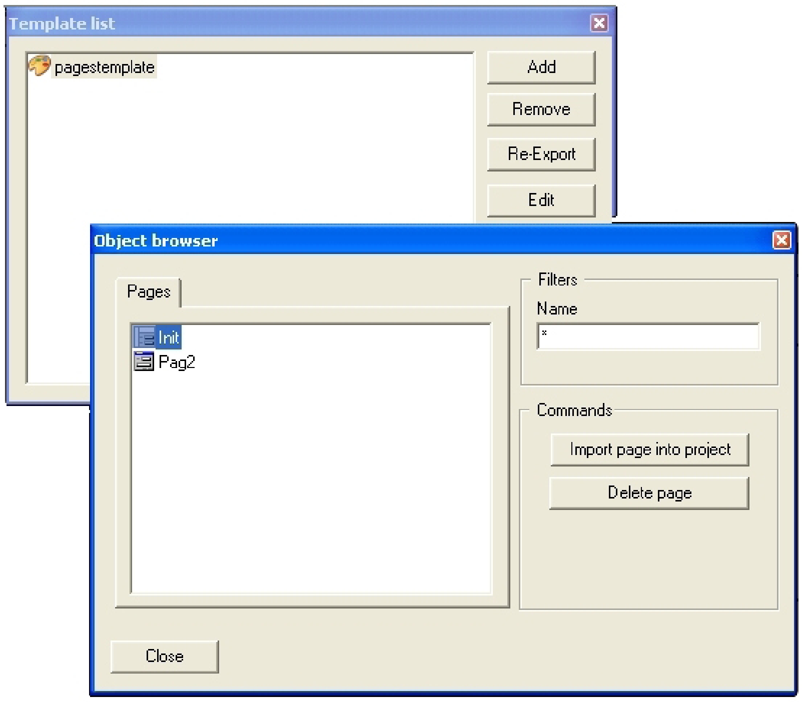
Once a template library has been added it is possible to use its elements simply dragging the chosen one from the template window and dropping it on the project tree.



Once the item has been dropped application asks the user for the name of the new page created (based on the template).

4.7.5.4 PROJECT TEMPLATE UPDATE

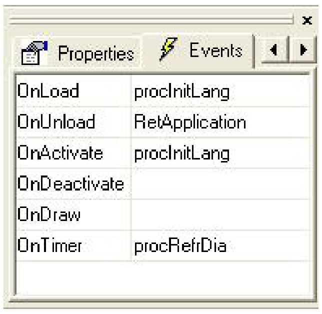
It is possible to delete templates from the (local) template library using `Edit` command in the `Template management` window.



4.8 EVENTS

There are different classes of events.

4.8.1 PAGE OR CONTROL EVENTS



OnLoad	procInitLang
OnUnload	RetApplication
OnActivate	procInitLang
OnDeactivate	
OnDraw	
OnTimer	procRefrDia

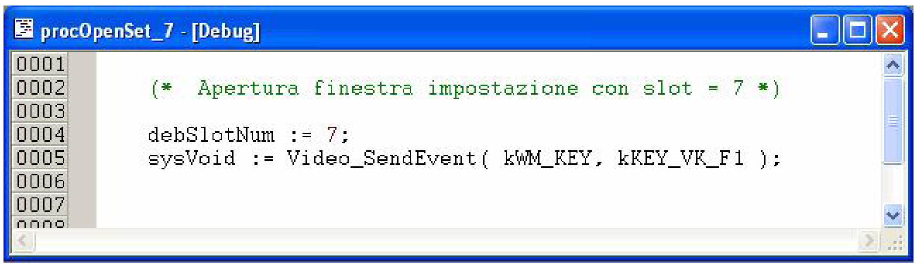
Each characteristic behaviour of a specific object can raise a specific event.

Each event can be associated to a procedure (see 4.8.4) that is executed each time the event takes place. The list of all available events for each PageLab object (page or control) is reported in Chapter 5.

4.8.2 KEY PRESSURE EVENTS

These events take place when a key is pressed, the raising of the event starts the execution of the associated action (see 4.8.5) if it is. The pressure of a key can be also simulated by software, see next paragraph.

4.8.3 EVENTS RAISED BY SOFTWARE



```

    (*  Apertura finestra impostazione con slot = 7 *)

    debSlotNum := 7;
    sysVoid := Video_SendEvent( kWM_KEY, kKEY_VK_F1 );
```

Programmer can raise events by software using the function *Video_SendEvent* inside the target software or in the body of the procedure, using following syntax:

```
  Video_SendEvent( event_id, param );
```

Where `event_id` is the identifier of the type of the event and param is an integer 16 bit parameter.

PageLabsupports software events defined in this table:

Event	Parameter	Description
`kWM_NULL`	Do not care	No event.
`kWM_KEY`	Key code	Simulates the pressure of the key specified as parameter then cause the associated action if it is.

Event	Parameter	Description
`kWM_MSG`	Window ID	Causes a system message that, once got by the system, causes the instant opening of the alarm page that has *Window ID* as identifier.
`kWM_SELECT`	Edit box handle	In touchscreen systems simulates the pressure on the edit box whose handle is passed as parameter, causing its selection or its transition to edit mode.
`kWM_PUSH`	Button handle	In touchscreen systems simulates the pressure on the button whose handle is passed as parameter, causing the execution of the associated action if it is.
`kEV_WM_ CHANGESETPAGE`	Page number	Shows the page specified by the parameter (if the context is a page in which sets are used).

4.8.4 PROCEDURES THAT CAN BE ASSOCIATED TO EVENTS

A procedure is a program that is executed when the event that has been associated to it, takes place. Events have been deeply described in previous paragraphs (see 4.8).

There are two classes of procedures:

Local procedures

This kind of procedures can be called only within the scope of the page in which are declared. In particular, they can be associated to the events of the page itself and of all their controls. The same can be said for software events raised when the page they refer to is active. Procedure code can contains references to all the types of variables, with local variables of the page too.

Global procedures

This kind of procedures can be called from every page and can be also used as periodic asynchronous routine of alarm management. They cannot contain variables references.

Here follow the description of the syntax to get the properties of a control from a procedure; similarly to C language printf it is:

```
“fb%s%s.%s”,   page_name, ctrl_name, prop_name
```

Where:

- `page_name` is the name of the page that has the control;
- `ctrl_name` is the name of the control;
- `prop_name` is the name of the property of the control.

So if we want to get the property *Foreground color* of the *Static* named *String_26* in *Main* page, we have to write:

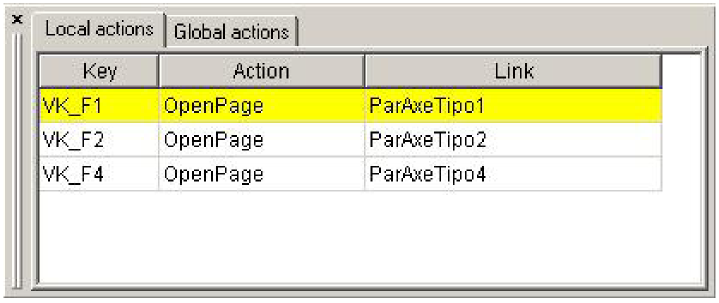
```
fbMainString_26.foreCol
```

N.B.: the name of the property to use in the scripts of the procedures is the name of the functional block exported by the software of the target (see 8.2), not the name in the properties window (see 3.3).

4.8.5 ACTIONS THAT CAN BE ASSOCIATED TO KEY PRESSURE

In common keyboard, not touchscreen systems, interaction between the user and the system is normally based on keys pressure.

PageLab shows the following table to the user.



Local actions | Global actions

Key	Action	Link
VK_F1	OpenPage	ParAxeTipo1
VK_F2	OpenPage	ParAxeTipo2
VK_F4	OpenPage	ParAxeTipo4

This table permits to associate a code of a key to one of the actions listed in the following table. In this way the pressure of that key causes the specified action.

Action	Link	Description
Call	Procedure name	Causes the invocation of the local or global procedure whose name is indicated in the *Link* field.
OpenPage	Page name	Causes the opening of the page whose name is indicated in the *Link* field.
Close	Do not care	Causes the closure of the current page
NextField	Do not care	Move the selection to the next edit box. If the system is not touchscreen moves selection to the buttons to allow their pressure.
PrevField	Do not care	Move the selection to the previous edit box.
Edit	Do not care	Access edit mode for the selected edit box. If the system is not touchscreen allows the user to simulate the pressure of the button.

There are two types of associations key-action:

- Local actions: local associations, valid only for the page currently open in the editor of the pages.
- Global actions: global associations, valid in any point of the project.

If the system has the touchscreen feature, normal interaction with user is made by the pressure of sensible area on the screen. However this table does not loss its meaning because allows the user to define virtual keys and to control their pressure by software causing in this way the dynamic execution of specific actions.

N.B.: if the same action is defined both at local and at global level, system does not give errors nor warnings because local declaration precedes global one.

4.9 RESOURCES

A resource is an interface element. User can get informations from resources or can use them to do actions.

PageLab supports different categories of resources that are managed by `Tab Resources project` window. (see 3.1). Categories are explained in details in the following paragraphs.

4.9.1 FONTS

Fonts are the different kinds of characters supported for the output of text strings on the screen. Fonts had been managed by PageLab old versions as text files with *.plf* extension and structured with the same syntax of the initialization definition of an array variable in IEC. Now, images are saved and loaded in binary format to optimize loading time on images of big size.

At project opening time, if PageLab finds this declaration, it searches in project folder for a file named *font_name.plk* and loads it in memory.

4.9.2 BITMAPS

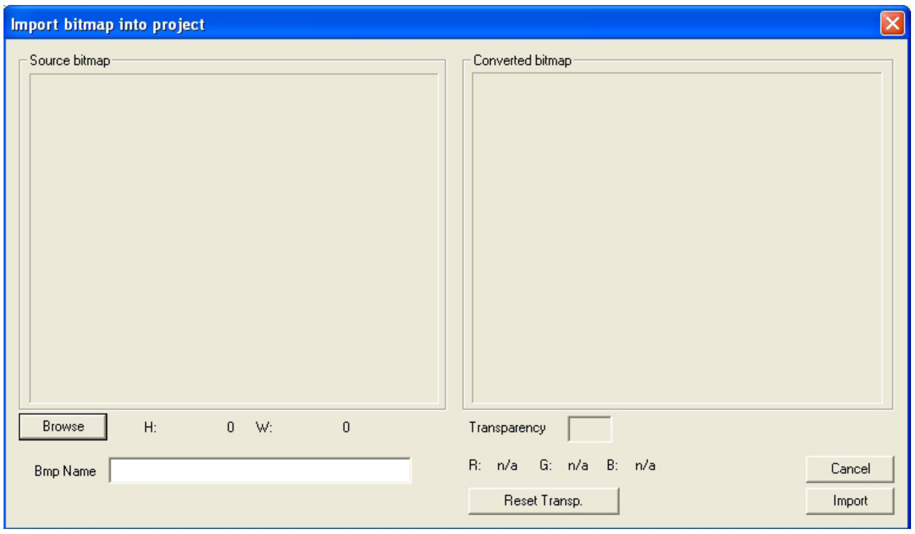
Bitmaps are pictures to associate to image controls (see 4.4.5). Bitmaps had been managed by PageLab old versions as text files with *.plb* extension and structured with the same syntax of the initialization definition of an array variable in IEC. Now, images are saved and loaded in binary format to optimize loading time on images of big size.

At project opening time, if PageLab finds this declaration, it searches in project folder for a file named *bitmap_name.plk* and loads it in memory.

PageLab provides a tool to convert bitmaps from Windows format to PageLab format.

To start this tool click on *Import resource > Bitmap* from *Project* menu, it is also possible to click on *Import bitmap* from context menu that can be shown by right click on *Bitmaps* node of resources tree.

This dialogue box will be shown as follows.



Click on *Browse* button to navigate computer resources to select desired source file.

In *Bmp Name* field user can personalize bitmap name that will be shown on resource tree; bitmap name is constituted by file name without extension and with *Bmp* prefix by default.

Transparency color field allows the user to specify transparency color, so a color that is not really drawn on the screen but a transparent color zone that does not cover elements previously drawn.

Transparency color can be personalized by choosing it by mouse from *Converted bitmap* window.

RGB indicates transparency color Red, Green, Blue components. *n/a* value indicates that no transparency color has been selected.

`Reset Transp.` button allows the user to undo last selected transparency color.

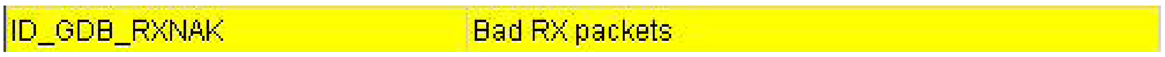
Once finished these operations it is possible to confirm bitmap importation by clicking on `Import` button.

4.9.3 STRINGS TABLE

In a PageLab project it is always possible to explicitly write the text to show on a text string or on a title of the page. It is also possible to refer to one of the strings of the resources specifying its ID.

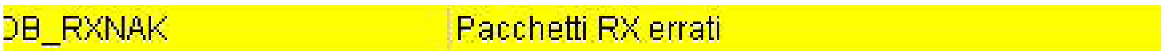
In first case text will be always the same, in second case the text that correspond to the active language will be shown.

So, English language string table contains the following record.



ID_GDB_RXNAK	Bad RX packets

And Italian language string table contains the following record.



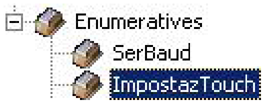
DB_RXNAK	Pacchetti RX errati

If we refer to the identifier *ID_GDB_RXNAK* from a page control or from a page, if current active language is English *Bad RX packets* will be shown, if current active language is Italian *Pacchetti RX errati* will be shown instead.

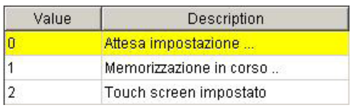
4.9.4 ENUMERATIVES

An enumerative is a data type defined by user, it is a set of constants named by user. Each element of an enumerative is treated as a constant and can be translated in all available languages of the project.

e.g.: we defined *ImpostazTouch* enumerative that is shown on resource tree as follows.



Enumerative records are shown by double clicking on `ImpostazTouch` node.

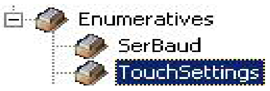


Value	Description
0	Attesa impostazione ...
1	Memorizzazione in corso ..
2	Touch screen impostato

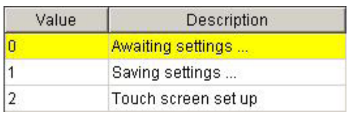
Now we introduce an *Edit box* control (see 4.4.3) and insert the name of the enumerative *ImpostazTouch* in its *Format property* field. Control will show the string associated to the value as it is in the table above, not the numeric value of the variable associated to the control.

If the numeric value of the variable does not match with any record of the enumerative table, an error string `########` is shown instead.

Even enumerative are supported by multi-language feature. In fact it is possible to personalize the name of the enumerative.



And its record values.



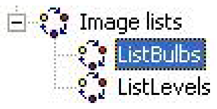
Value	Description
0	Awaiting settings ...
1	Saving settings ...
2	Touch screen set up

4.9.5 IMAGES LISTS

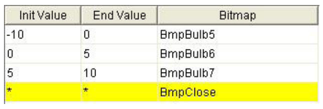
An images list is very similar to an enumerative but with the following differences:

- intervals of constants are supported, not only simple values;
- each value has an image associated;
- a list of images determines the content shown by an `Animation` control, while an enumerative can be associated to an Edit box.

e.g.: now we have an images list `ListBulbs` that is shown on the resource tree.



It is possible to see all the records of the list by double-clicking the node.



Init Value	End Value	Bitmap
-10	0	BmpBulb5
0	5	BmpBulb6
5	10	BmpBulb7
*	*	BmpClose

If we introduce an `Animation` control (see 4.4.6) in the page, and we set its property `Image` list with the name of the enumerative `ListBulbs`, the control will show the image whose specified interval includes the value of a variable associated to the control.

If the numeric value of the associated variable does not match any record in the list a default image (with init and end value set to *) will be shown if it is. If no default image is specified no image will be drawn.

4.9.6 SETS

As it is described above (see 4.6) sets are ensemble of global variable even of distinct type.

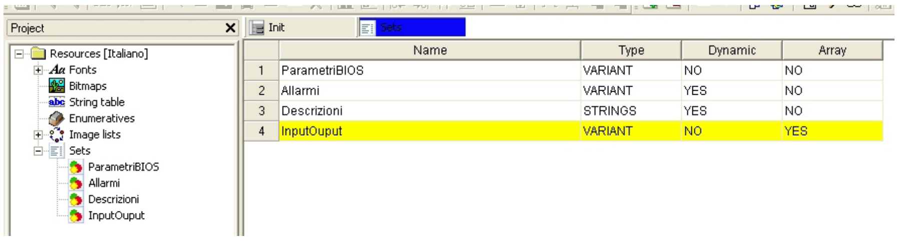
In particular there are two types of set:

- Variable/parameter sets even of not equal type (`VARIANT`);
- Strings sets (`STRINGS`).

The sets of the first type are defined indicating VARIANT as type. This kind of set has the following attributes:

- Dynamic: indicates that every n execution cycles target automatically reloads the elements of the set and hide those elements that have no visibility (boolean constant `FALSE` or associated visibility variable set to false at that moment).
- Array: indicates that the unique element of this set is a variable of type array.

N.B.: this kind of set can be assigned only to an edit-box control.

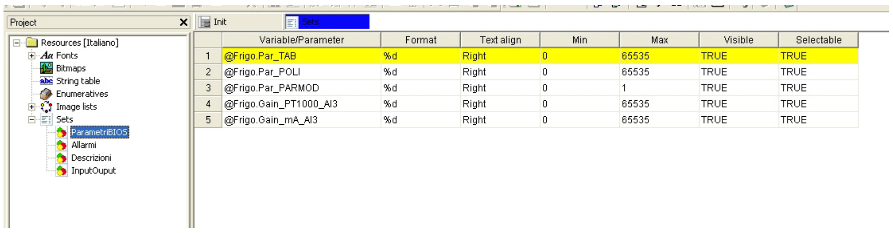


	Name	Type	Dynamic	Array
1	ParametriBIOS	VARIANT	NO	NO
2	Allarmi	VARIANT	YES	NO
3	Descrizioni	STRINGS	YES	NO
4	InputOuput	VARIANT	NO	YES

In this example four sets with different characteristics have been defined.

Once defined a set, each element of the set can be added via drag & drop from *Target vars and parameters* or can be manually inserted by user.

Lets see how to manage *ParametriBIOS* set.

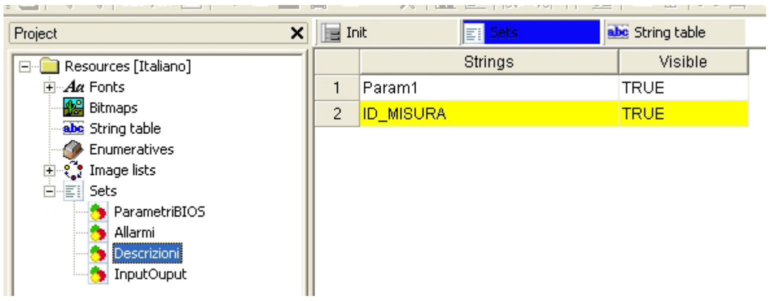


	Variable/Parameter	Format	Text align	Min	Max	Visible	Selectable
1	@Frigo.Par_TAB	%d	Right	0	65535	TRUE	TRUE
2	@Frigo.Par_POLI	%d	Right	0	65535	TRUE	TRUE
3	@Frigo.Par_PARMOD	%d	Right	0	1	TRUE	TRUE
4	@Frigo.Gain_PT1000_AI3	%d	Right	0	65535	TRUE	TRUE
5	@Frigo.Gain_mA_AI3	%d	Right	0	65535	TRUE	TRUE

Following attributes can be defined:

- Variable/Parameter = variable/parameter name.
- Format = indicates how to show associated variable value specifying a syntax analogous to C language printf (see paragraph 5.7.2).
- Text Align = the alignment of the text to show.
- Min/Max = minimum and maximum value for the element of the set.
- Visible = boolean variable or constant that defines the visibility of the element. If dynamic feature of the set is active the variable is periodically checked to hide or show the element.
- Selectable = indicates that the element can be selected. In this case a boolean variable or constant can be assigned too.

For a set of type STRING each element of the set is quite simple as it is shown in the next figure:



	Strings	Visible
1	Param1	TRUE
2	ID_MISURA	TRUE

We have to define only two attributes, the string or the ID of a string resource (see 4.9.3) and the variable/constant of visibility. As we said an element not visible will not be shown on the screen.

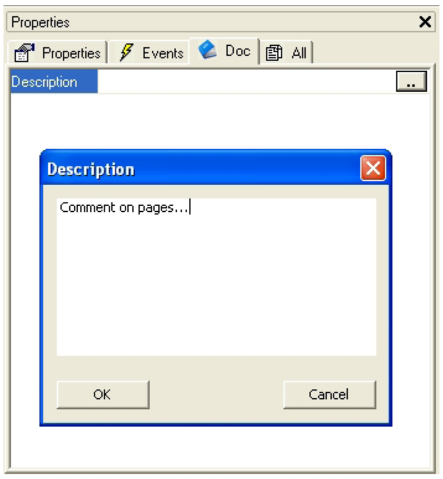
N.B.: this kind of set can be used with Static control only.

4.10 AUTOMATIC DOCUMENTATION

During project development it is usually necessary to write comments for each page in order to explain how the page works.

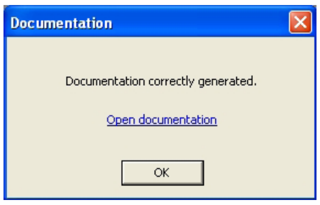
PageLab integrates into its development environment the automatic documentation feature that consists in the generation of a graphical report with all the previously inserted comments followed by the pages they refer to.

Comments related to controls and pages should be inserted in the *Doc* tab of the properties window.



Documentation is generated when the apposite button is pressed.

At the end of the process the following dialogue is shown. By clicking on the *Open documentation* link it is possible to view the generated report using the browser.


Documentation

Documentation correctly generated.

Open documentation

OK

It is also possible to manually open the *.html* file generated. This file is created in the project folder and is named *project name.html*.

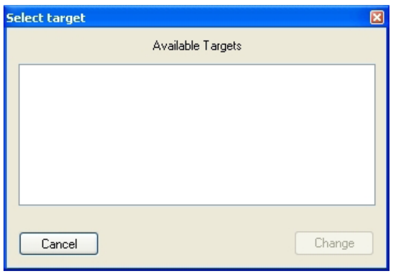
N.B.: documentation generation process requires the file *Documentation.xsl* to be in the project folder. This file can be personalized by user to redefine report style.

4.11 MANAGING PROJECTS

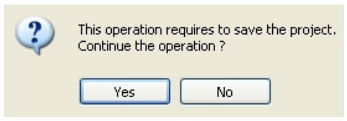
4.11.1 SELECTING THE TARGET DEVICE

You may need to port a PLC application on a target device which differs from that you originally wrote the code for. Follow the instructions below to adapt your PageLab project to a new target device.

1) Click *Select target* in the *Project* menu of the PageLab main window. This causes the following dialog box to appear.



2) Select one of the target devices listed in the combo box.
3) Click `Change` to confirm your choice, `Cancel` to abort.
4) If you confirm, PageLab displays the following dialog box.


This operation requires to save the project.
Continue the operation ?

Press `Yes` to complete the conversion, `No` to quit.

If you press `Yes`, PageLab updates the project to work with the new target.

It also makes a backup copy of the project file(s) in a sub-directory inside the project directory, so that you can roll-back the operation by manually (i.e., using Windows Explorer) replacing the project file(s) with the backup copy.

5. APPENDIX I: PAGE PROPERTIES AND OBJECT PROPERTIES

5.1 FRAME SET

5.1.1 PROPERTIES

Properties	Available values	Description
TopDim	>= 0	Top-height of the frame (#pixel).
BottomDim	>= 0	Bottom-height of the frame (#pixel).
LeftDim	>= 0	Left-width of the frame (#pixel).
RightDim	>= 0	Right-width of the frame (#pixel).
CharDimX	> 0	Horizontal space among grid points (#pixel).
CharDimY	> 0	Vertical space among grid points (#pixel).
Font	Name found in *Resources*	Default font used when inserting new objects in page.
Background Color	...	Background color selectable from palette. In addition this color is also set when inserting new objects in the frame.
Text Color	...	Foreground color selectable from palette. This color is set when inserting new objects in the frame.
Appearance	*Flat*, *Raised*, *Sunken*	- Flat - Raised - Sunken

5.2 CHILD PAGE

5.2.1 PROPERTIES

Properties	Available values	Description
CharDimX	> 0	Horizontal space among grid points (#pixel).
CharDimY	> 0	Vertical space among grid points (#pixel).
Font	Name found in *Resources*	Default font used when inserting new objects in page.
Background Color	...	Background color selectable from palette. In addition this color is also set when inserting new objects in the frame.

Properties	Available values	Description
Text Color	...	Foreground color selectable from palette. This color is set when inserting new objects in the frame.
Title bar	*Yes*, *No*	Title bar, settings can be found in *System options* dialog: - *Yes*: page has title; - *No*: page has not title.
Page Border	*Yes*, *No*	- *Yes*: page with outer border; - *No*: page without outer border.
Caption	Text otherwise *Resource ID*	Text on title bar or *Resource ID*. This property is not sensible if *Title Bar* field is set to *No*.
System menu	*Yes*, *No*	If *Yes* denotes that there is a button with 'X' image on it and the behaviour is similar to *Windows Dialog*: - *Yes*: page has close button; - *No:* page has not close button.
Appearance	*Flat*, *Raised*, *Sunken*	- Flat - Raised - Sunken

5.2.2 EVENTS

Event	Description
OnLoad	On loading this page, i.e. when calling from parent page.
OnUnload	On closing this page, when the page returns and the parent page will be restored.
OnDeactivate	On calling a child page and the current page is no more active. This event does not exist in main page.
OnActivate	When the previous opened child page will be closed. This event does not appear in leaf page, i.e in the pages which do not call child pages.
OnDraw	When the page starts drawing all the objects. The page has just drawn border, background, and title.
OnTimer	Asynchronous event. The user can link a procedure and it will be executed cyclically.

5.3 POP-UP PAGE

5.3.1 PROPERTIES

Properties	Available values	Description
XPos	>= 0	Top-left 'x coordinate' edge of full page.
YPos	>= 0	Top-left 'y coordinate' edge of full page.
XDim	> 0	Width of the page (#pixel).
YDim	> 0	Height of the page (#pixel).
CharDimX	> 0	Horizontal space among grid points (#pixel).
CharDimY	> 0	Vertical space among grid points (#pixel).
Modal	*Yes, No*	- *Yes*: all the parent page objects will be disabled; - *No*: all the parent page objects will be enabled if they are completely visible.
Font	Name found in *Resources*	Default font used when inserting new objects in page.
Background Color	...	Background color selectable from palette. In addition this color is also set when inserting new objects in the frame.
Text Color	...	Foreground color selectable from palette. This color is set when inserting new objects in the frame.
Title bar	*Yes, No*	Title bar, the settings can be found in *System options* dialog: - *Yes*: page has title; - *No*: page has not title.
Page Border	*Yes, No*	- *Yes*: page with outer border; - *No*: page without outer border.
Caption	Text otherwise *Resource ID*	Text on title bar or *Resource ID*. This property is not sensible if the *Title Bar* field is set to *No*.
System menu	*Yes, No*	If *Yes* denotes that there is a button with *X* image on it and the behaviour is similar to *Windows Dialog:* - *Yes*: page has close button; - *No*: page has not close button.
Appearance	*Flat, Raised, Sunken*	Flat Raised Sunken

5.3.2 EVENTS

Event	Description
OnLoad	On loading this page, i.e. when calling from parent page.
OnUnload	On closing this page, when the page returns and the parent page will be restored.
OnDeactivate	On calling a child page and the current page is no more active. This event does not exist in main page.
OnActivate	When the previous opened child page will be closed. This event does not appear in leaf page, i.e in the pages which do not call child pages.
OnDraw	When the page starts drawing all the objects. The page has just drawn border, background and title.
OnTimer	Asynchronous event. The user can link a procedure and it will be executed cyclically.

5.4 STATIC

5.4.1 PROPERTIES

Properties	Available values	Description
XPos	>= 0	Top-left 'x coordinate' edge relative to page.
YPos	>= 0	Top-left 'y coordinate' edge relative to page.
Name	Not empty	Name of object.
Text	Text otherwise *Resource ID*	Text or *Resource ID* shown in the object.
Font	Name found in *Resources*	Font used for drawing the text in object.
Background Color	...	Background color selectable from palette.
Text Color	...	Text color selectable from palette.
Sel. Background	...	Background color selectable from palette when the object is chosen. This property is not sensible if the *Select* field is constant *FALSE*.
Sel. Foreground	...	Text color selectable from palette when the object is chosen. This property is not sensible if the *Select* field is constant *FALSE*.
Appearance	*Flat, Raised, Sunken*	- Flat - Raised - Sunken
Border points	>= 0	Border thickness (#pixel). This property is sensible only if *Appearance* is set to *Flat*.
Border color	...	Border color selectable from palette. This property is sensible only if *Appearance* is set to *Flat*.

Properties	Available values	Description
Number of Chars	> 0	Number of chars that this object can show. If the value is *0* the object will show the complete text. Otherwise with another value it can be truncated or extended.
Alignment	*Right, Center, Left*	Text alignment in the object.
Refresh	*TRUE, FALSE*	Continuous redraw of the object: - *FALSE*: the *Text value* is read from memory and updated only when opening the page or when a child page is closed; - *TRUE:* the *Text value* is read from memory and always updated.
Select	*TRUE, FALSE, var_name*	Selected status of the object. It can be constant (*TRUE* or *FALSE*) or linked with a boolean variable *var_name:* if *var_name* is *TRUE* the object is selected and so it will show the colors *Select Back*, *Select Fore*.
Visible	*TRUE, FALSE, var_name*	Visible status of the object. It can be constant (*TRUE* or *FALSE*) or linked with a boolean variable *var_name*: if *var_name* is *TRUE* the object is visible, otherwise it is hidden.

5.4.2 EVENTS

Event	Description
BeforeUpdate	Before the object is redrawn.
AfterUpdate	Immediately after the object is redrawn.

5.5 LINE

5.5.1 PROPERTIES

Properties	Available values	Description
XPos	>= 0	Top-left 'x coordinate' edge relative to page.
YPos	>= 0	Top-left 'y coordinate' edge relative to page.
X2Pos	> 0	Bottom-right 'x coordinate' edge relative to page.
Y2Pos	> 0	Bottom-right 'y coordinate' edge relative to page.
Name	Not empty	Name of object.
Thickness pts	> 0	Line thickness (#pixel).
Border col	...	Line color selectable from palette.

5.6 RECTANGLE

5.6.1 PROPERTIES

Properties	Available values	Description
XPos	>= 0	Top-left 'x coordinate' edge relative to page.
YPos	>= 0	Top-left 'y coordinate' edge relative to page.
XDim	> 0	Width (#pixel).
YDim	> 0	Height (#pixel).
Name	Not empty	Name of object.
Border points	> 0	Border thickness (#pixel).
Border color	...	Border color selectable from palette.
Background Color	...	Background color selectable from palette. This property is sensible only if *Transparent* is set to *TRUE*.

5.7 EDIT BOX

5.7.1 PROPERTIES

Properties	Available values	Description
XPos	>= 0	Top-left 'x coordinate' edge relative to page.
YPos	>= 0	Top-left 'y coordinate' edge relative to page.
Name	Not empty	Name of object.
Appearance	*Flat*, *Raised*, *Sunken*	- *Flat:* plain with use of *Border pts* and *Border col;* - *Raised*; - *Sunken*.
Font	Name found in *Resources*	Font used for drawing the text in object.
Background Color	...	Background color selectable from palette.
Text Color	...	Text color selectable from palette.
Sel. Background	...	Background color selectable from palette when the object is chosen. This property is not sensible if the *Selectable* field is constant *FALSE*.
Sel. Foreground	...	Text color selectable from palette when the object is chosen. This property is not sensible if the *Selectable* field is constant *FALSE*.
Border points	>= 0	Border thickness (#pixel). This property is sensible only if *Appearance* is set to *Flat*.
Border color	...	Border color selectable from palette. This property is sensible only if *Appearance* is set to *Flat*.

Properties	Available values	Description
Number of Chars	> 0	Chars visible in the object. Width of entire object is calculated among this value and the size of *Font*. If *NumChar* are less than the value, the object shows this error string: #####.
Format	String as `printf` or `.enum_name`	The format can be numeric, to define as `printf` of C language (see 5.7.2), enumerative, if in this field there is `enum_name` defined in *Resources* (see 4.9).
Alignment	*Right, Center, Left*	Text alignment in the object.
Access	*RO, RW*	Accesses variable *Assoc var* used in object: - *RO* = read only; - *RW* = read/write.
Selection Order	>= 0	Selection order of the object. It can be selected by pressing a key or by means of a procedure. In this case the selection moves from the current object to the previous or next *Sel. Order* object.
Variable	Not empty	Name of the variable that can be shown and edited with this object. It can be any variable of the project, (local, global, imported from PLC or target - see 2.9.2), a parameter (see 2.9.2) or an element of a set (see 4.6).
Data type	*UNDEF, BOOL, SINT, USINT, BYTE, INT, UINT, WORD, DINT, UDINT, DWORD, REAL, STRING*	Type of *Assoc var*. If it is a variable, the type is defined automatically. This property is sensible if *Assoc var* is an explicit parameter.
Low limit	*CONSTANT, var_name*	Name of variable or numeric constant. This is the least number that the object can show. It can be any variable of the project, (local, global, imported from PLC or target - see 2.9.2). This object shows an error string (!!!!!!!) if condition does not holds. The * symbol means that there is no low limit.
High limit	*CONSTANT, var_name*	Name of the variable or numeric constant. This is the maximum number that the object can show. It can be any variable of the project, (local, global, imported from PLC or target - see 2.9.2). This object views an error string (*!!!!!!!*) if condition does not hold. The * symbol means that there is no high limit.
Refresh	*TRUE, FALSE*	Enables continuous update of the value: - *FALSE:* the *Assoc var* value is read from memory and updated only when open page or when a child page is closed; - *TRUE*: the *Assoc var* value is read from memory and always updated.

Properties	Available values	Description
Visible	*TRUE, FALSE, var_name*	Visible status of the object. It can be constant (*TRUE* or *FALSE*) or linked with a boolean variable *var_name*: if *var_name* is *TRUE* the object is visible, otherwise hidden.
Selectable	*TRUE, FALSE, var_name*	Selected status of the object. It can be constant (*TRUE* or *FALSE*) or linked with a boolean variable *var_name*: if *var_name* is *TRUE* the object is selected and so it will show the colors *Select Back*, *Select Fore*. If this field is *FALSE* the *Access* property is not sensible.
Label	*Text* otherwise Resource ID	This string can be visible on the touch screen keyboard if supported on the target and shows the label of the field.

5.7.2 FORMAT SPECIFICATION - PRINTF

If the object has not any enumerative format, the format string is composed as follows:

```
%[flags][width][.precision]type
```

The field has one or more characters, that describe the specification. The simplest format contains only percentage symbol and one char as type (for example: *%s*).

Next table explains in details functions and values.

Field	Available values	Description
flags	- + prints always the sign, even if the number is positive. - 0 prints zeros in head until *width* (if specified) or *NumChar*.	This char is an option for chars order, print sign, number of decimal digit. This field may have more than one flag.
width	> 0, <= *NumChar*	Maximum chars can be printed. Allows to view values that do not fill *NumChar* fully.
precision	>= 0	Decimal digits after the point. If the field is an integer and there is a precision the object shows a decimal point. E.g. the value is 102 integer, and precision is 2, with %.2d, the number is shown as 1.02.
type	- %d: Integer with sign. - %f: Real. - %x: Hexadecimal with lowercase chars. - %x: Hexadecimal with uppercase chars. - %s: String. - %@sdf: Password. - [%d,u,f,x]: Custom measure unit format.	Mandatory field.

5.7.3 EVENTS

Event	Description
BeforeUpdate	Before the object is redrawn.
AfterUpdate	Immediately after the object is redrawn.
OnEnter	Whenever the object is selected and receives the command for entering in edit-mode.
OnClick	Whenever HMI receives a pressure on the object, valid only for touchscreen systems.
OnChange	Whenever the user confirms the modifications and the value is different from start.

5.8 TEXT BOX

5.8.1 PROPERTIES

Properties	Available values	Description
XPos	>= 0	Top-left 'x coordinate' edge relative to page.
YPos	>= 0	Top-left 'y coordinate' edge relative to page.
Name	Not empty	Name of object.
Appearance	*Flat, Raised, Sunken*	- *Flat*: plain with use of *Border pts* and *Border col*; - *Raised*; - *Sunken*.
Font	Name found in *Resources*	Font used for drawing the text in the object.
Background Color	...	Background color selectable from palette.
Text Color	...	Text color selectable from palette.
Border points	>= 0	Border thickness (#pixel). This property is sensible only if *Appearance* is set to *Flat*.
Border color	...	Border color selectable from palette. This property is sensible only if *Appearance* is set to *Flat*.
Number of Chars	> 0	Chars visible in the object. Width of entire object is calculated among this value and the size of *Font*.
Number of Rows	> 0	Rows visible in the object. Height of entire object is calculated among this value and the size of *Font*.
Show line number	*TRUE, FALSE*	Flag for viewing number of lines.
Access	*RO, RW*	Access on variable *Assoc string* used in object: - *RO:* read only; - *RW*: read/write.

Properties	Available values	Description
Selection order	>= 0	Selection order on which the object can be selected with the pressure of a key or with a procedure. In this case the selection moves from the current object to the previous or next *Sel.Order object*.
String variable	*Not empty*	Name of variable that can be shown and edited with this object. It can be any string variable of the project, (local, global, imported from PLC or target - see 2.9.2).
Refresh trg	*TRUE, FALSE, var_name*	Enables update of the value: - *FALSE*: the *Assoc string* value is read from memory and updated only when opening page or when a child page is closed. - *TRUE*: the *Assoc string* value is read from memory and always updated. The runtime sets automatically the value to *FALSE*.
Visible	*TRUE, FALSE, var_name*	Visible status of the object. It can be constant (*TRUE* or *FALSE*) or linked with a boolean variable *var_name*: if *var_name* is *TRUE* the object is visible, otherwise hidden.

5.8.2 EVENTS

Event	Description
BeforeUpdate	Before the object is redrawn.
AfterUpdate	Immediately after the object is redrawn.
OnClick	Whenever HMI receives a pressure on the object, valid only for touchscreen systems.
OnChange	Whenever the user confirms the modifications and the value is different from start.

5.9 IMAGE

5.9.1 PROPERTIES

Properties	Available values	Description
XPos	const >= 0, variable	Top-left 'x coordinate' edge relative to page. It is possible to assign a variable only if *Style* is set to *Floating*.
YPos	const >= 0, variable	Top-left 'y coordinate' edge relative to page. It is possible to assign a variable only if *Style* is set to *Floating*.
XDim	> 0	Width (#pixel).
YDim	> 0	Height (#pixel).

Properties	Available values	Description
Name	Not empty	Name of object.
Appearence	*Flat, Raised, Sunken*	- *Flat* = plain with use of *Border pts* and *Border col*; - *Raised*; - *Sunken*.
Border points	>= 0	Border thickness (#pixel). This property is sensible only if *Appearance* is set to *Flat*.
Border color	...	Border color selectable from palette. This property is sensible only if *Appearance* is set to *Flat*.
Bitmap	Name found in *Resources*	Bitmap used for drawing the image in object.
Background image	Image object in the page	Name of another object that is redrawn when *Style* is set to *Floating*. It is sensible only if it is overlapped with this image.
Visible	*TRUE, FALSE, var_name*	Visible status of the object. It can be constant (*TRUE* or *FALSE*) or linked with a boolean variable *var_name*: if *var_name* is *TRUE* the object is visible, otherwise it is hidden.
Style	*Docking, Floating*	- *Docking*: fixed position; - *Floating*: variable position, according to *XPos* variable and *Ypos* variable.

5.10 ANIMATION

5.10.1 PROPERTIES

Properties	Available values	Description
XPos	>= 0	Top-left 'x coordinate' edge relative to page.
YPos	>= 0	Top-left 'y coordinate' edge relative to page.
XDim	> 0	Width (#pixel).
YDim	> 0	Height (#pixel).
Name	Not empty	Name of object.
Appearance	*Flat, Raised, Sunken*	- *Flat* = plain with use of *Border pts* and *Border col*; - *Raised*; - *Sunken*.
Border points	>= 0	Border thickness (#pixel). This property is sensible only if *Appearance* is set to *Flat*.
Border color	...	Border color selectable from palette. This property is sensible only if *Appearance* is set to *Flat*.

Properties	Available values	Description
Image list	Name found in *Resources*	It contains the images that the object can view and the value range.
Animation variable	*var_name*	Name of the variable that is compared with value range in *Image list*.
Data type	*SINT, USINT, BYTE, INT, UINT, WORD, DINT, UDINT, DWORD*	Type of *Animation var*. If it is a variable, the type is automatically defined.
Visible	*TRUE, FALSE, var_name*	Visible status of the object. It can be constant (*TRUE* or *FALSE*) or linked with a boolean variable *var_name*: if *var_name* is *TRUE* the object is visible, otherwise hidden.

5.10.2 EVENTS

Event	Description
BeforeUpdate	Before the object is redrawn.
AfterUpdate	Immediately after the object is redrawn.

5.11 BUTTON

5.11.1 PROPERTIES

Properties	Available values	Description
XPos	>= 0	Top-left ‘x coordinate’ edge relative to page.
YPos	>= 0	Top-left 'y coordinate' edge relative to page.
XDim	> 0	Width (#pixel).
YDim	> 0	Height (#pixel).
Name	Not empty	Name of object.
Text	Empty or explicit text or Resource ID	Text to view in the button: - string; - Resource ID.
Text in selection	Empty or explicit text or Resource ID	Text to view in the button when it is selected: - string; - Resource ID.
Text color	...	Text color selectable from palette.

Properties	Available values	Description
Text color in selection	...	Text color selectable from palette. This property is sensible if the 'Selection variable' is a variable, a TRUE costant value or in addition if the border style is 3D and the 'Selection variable' has the 'On Press' constant.
Text Alignment	Right, Center, Left	Text alignment in the object.
Bitmap	Empty or bitmap	Bitmap to view in the button.
Bitmap in selection	Empty or bitmap	Bitmap to view in the button. This property is sensible if the 'Selection variable' is a variable, a TRUE costant value or in addition if the border style is 3D and the 'Selection variable' has the 'On Press' constant.
Image Alignment	Right, Center, Left	Alignment of the bitmap/selected bitmap.
Font	Name found in *Resources*	Font used for drawing the text in object. This field is not sensible if it shows a bitmap.
Border style	*Flat*, *Raised*, *Sunken*, *3D*	- *Flat*: plain with use of *Border pts* and *Border col*; - *Raised*; - *Sunken*; - *3D*: raised if not selected, sunken if selected.
Border points	*>= 0*	Border thickness (#pixel). This property is sensible only if *Appearance* is set to *Flat*.
Border color	...	Border color selectable from palette. This property is sensible only if *Appearance* is set to *Flat*.
Background color	...	Background color selectable from palette.
Selection border	...	Border color when the object is selected. This property is not sensible if *Selection var* is *FALSE* fixed.

Properties	Available values	Description
Sel. background	...	Background color when the object is selected. This property is not sensible if *Selection var* is *FALSE* fixed.
Selection order	*>= 0*	Selection order on which the object can be selected with the pressure of a key or with a procedure. In this case the selection moves from the current object to the previous or next *Sel. Order* object.
Visible	*TRUE*, *FALSE*, *var_name*	Visible status of the object. It can be constant (*TRUE* or *FALSE*) or linked with a boolean variable *var_name*: if *var_name* is *TRUE* the object is visible, otherwise it is hidden.
Press variable	*Empty* or *var_name*	When the button is pressed *var_name* is set to *TRUE*. When the button is not pressed, *var_name* is set to *FALSE*.
Selection variable	*TRUE*, *FALSE*, *var_name*	Selected status of the object. It can be constant (*TRUE* or *FALSE*) or linked with a boolean variable *var_name*: if *var_name* is *TRUE* the object is selected and so it will show the properties for selection.
Type	*Normal*, *Toggle*	If toggle after the pressure the *Press Variable* has the negated value. If the *Selection variable* is *On Press* the appearance is sunken and selected, otherwise is raised and not selected.
Action	*Call*, *OpenPage*, *Close*, *NextField*, *PrevField*, *Edit*	Action executed on button release only when the button is released with coordinates inside the button.
Action par	*page_name* *proc_name*	Parameter associated with the action executed on button release. It is sensible only if *Action* is *OpenPage* (*Action par* = name of the page to open) or *Call* (*Action par* = name of the procedure to execute).

5.11.2 EVENTS

Event	Description
OnPress	The event is executed on pressure. It is always executed if the pressure starts inside the button.
OnAction	The event is executed before the execution of the defined action and only when the button is released with coordinates inside the button.
OnRelease	This event is executed on release, after the *OnAction* and it is always executed also if the coordinates are not inside the button.

5.12 PROGRESS BAR

5.12.1 PROPERTIES

Properties	Available values	Description
XPos	>= 0	Top-left 'x coordinate' edge relative to page.
YPos	>= 0	Top-left 'y coordinate' edge relative to page.
XDim	> 0	Width (#pixel).
YDim	> 0	Height (#pixel).
Name	Not empty	Name of object.
Appearance	*Flat*, *Raised*, *Sunken*	- *Flat:* plain with use of *Border pts* and *Border col*; - *Raised*; - *Sunken*.
Border points	>= 0	Border thickness (#pixel). This property is sensible only if *Appearance* is set to *Flat*.
Border color	...	Border color selectable from palette. This property is sensible only if *Appearance* is set to *Flat* or *Text* is not empty.
Bar color	...	Color of step bar, selectable from palette.
Background color	...	Background color selectable from palette.
Visible	*TRUE*, *FALSE*, *var_name*	Visible status of the object. It can be constant (*TRUE* or *FALSE*) or linked with a boolean variable *var_name*: if *var_name* is *TRUE* the object is visible, otherwiseit is hidden.
Refresh trigger	*TRUE*, *FALSE*, *var_name*	Object redraw: - *FALSE:* the *Progress var* value is read from memory and updated only when opening page or when a child page is closed. - *TRUE:* the *Progress var* value is read from memory and always updated. - *var_name:* the *Progress var* value is read from memory and updated only when the variable becomes *TRUE*. After the update the runtime sets it to *FALSE*.
Progress variable	Not empty	Step variable. This is the filling percentage of bar in relation with the range assigned by *Lo limit* and *Hi limit*. It can be any string variable of the project (local, global, imported from PLC or target) or a parameter (see 2.9.2).
Data type	UNDEF, BOOL, SINT, USINT, BYTE, INT, UINT, WORD, DINT, UDINT, DWORD, LWORD, REAL, LREAL, STRING	Type of *Progress var*. If it is a variable, the type is automatically defined. This property is sensible if *Progress var* is an explicit parameter.

Properties	Available values	Description
`Low limit`	Constant or `var_name`	Name of the variable or numeric constant. This is the least value for step bar. It can be any variable of the project, (local, global, imported from PLC or target - See. §) with type specified by `Data type`.
`High limit`	Constant or `var_name`	Name of the variable or numeric constant. This is the maximum value for step bar. It can be any variable of the project, (local, global, imported from PLC or target - See. §) with type specified by `Data type`.
`Orientation`	Horizontal, vertical	Direction of step bar.

5.12.2 EVENTS

Event	Description
`BeforeUpdate`	Before the object is redrawn.
`AfterUpdate`	Immediately after the object is redrawn.

5.13 CUSTOM CONTROL

5.13.1 PROPERTIES

Properties	Available values	Description
`XPos`	>= 0	Top-left 'x coordinate' edge relative to page.
`YPos`	>= 0	Top-left 'y coordinate' edge relative to page.
`XDim`	> 0	Width (#pixel).
`YDim`	> 0	Height (#pixel).
`Name`	Not empty	Name of object.
`Control ID`	> 0	Identifier of custom control type.
`Visible`	`TRUE`, `FALSE`, `var_name`	Visible status of the object. It can be constant (`TRUE` or `FALSE`) or linked with a boolean variable `var_name`: if `var_name` is `TRUE` the object is visible, otherwise it is hidden.
`Refresh`	`TRUE`, `FALSE`	Continuous redraw of the object: - `FALSE`: the body of the runtime object is updated only when opening page or when a child page is closed. - `TRUE:` the body of the runtime object is always updated.

5.13.2 EVENTS

Event	Description
BeforeUpdate	Before the object is redrawn.
AfterUpdate	Immediately after the object is redrawn.

5.14 CHART

5.14.1 PROPERTIES

Properties	Available values	Description
XPos	>= 0	Top-left 'x coordinate' edge relative to page.
YPos	>= 0	Top-left 'y coordinate' edge relative to page.
XDim	> 0	Width (#pixel).
YDim	> 0	Height (#pixel).
Name	Not empty	Name of object.
Track 1	*var_name*	Opens a dialog with the following options: - the array with data of track (*Data Source*); - the visibility condition of the track (*TRUE* or boolean variable); - *Color* of the track; - the scale factor (range among two horizontal divisions); - the offset (displacement of the track 0-Y); - step of print label for Y axis; - three horizontal bars with name, value and colors. If *var_name* is empty the track is not defined and not drawn.
Track 2	*var_name*	As *Track 1*, but for track 2.
Track 3	*var_name*	As *Track 1*, but for track 3.
Track 4	*var_name*	As *Track 1*, but for track 4.
Track 5	*var_name*	As *Track 1*, but for track 5.
Track 6	*var_name*	As *Track 1*, but for track 6.
Track 7	*var_name*	As Track 1, but for track 7.
Track 8	*var_name*	As *Track 1*, but for track 8.
Track Left	>=0, *var_name*	Integer value that is the track for Y axis left. It is admitted a constant value (ex 1). If empty means that the chart must not draw the label for Y axis left.
Track Right	>=0, *var_name*	As *Track Left* for the right side.
Format Left	String as *printf*	Format of Y axis left as c *printf* function.
Format Right	String as *printf*	As *Format Left* for the right side.
XLabel	>=0, *var_name*	Step for X-axis labels. How many divisions of horizontal bar must have labels.

Properties	Available values	Description
X Scale	*var_name*	Scale factor of x-Axis. Value range among two divisions of horizontal bars. An empty value indicates that the chart is in autoscale mode.
Len Data	*var_name*	Name of the variable that contains the array index samples. The chart adds the sample values when the *Refresh* is *TRUE*. If the value remains unchanged the chart does not add new values. The runtime maintains the last value of this field. Constant values are not allowed.
X Offset	*var_name*	Variable name for the deviation of 0 for x-Axis, left or right. A positive value moves the chart values to left.
Grid	*Yes, No*	Visibility of the grid.
X Div. Grid	value	Number of division on horizontal bar, used with scale factor and offset for drawing the chart tracks.
Y Div. Grid	value	Number of division on vertical bar, used with scale factor and offset for drawing the chart tracks.
Background color	...	Background color selectable from palette.
Appearance	*Flat, Raised, Sunken*	- *Flat*: plain with use of *Border pts* and *Border col*; - *Raised*; - *Sunken*.
Border points	>= 0	Border thickness (#pixel). This property is sensible only if *Appearance* is set to *Flat*.
Border color	...	Border color selectable from palette. This property is sensible only if *Appearance* is set to *Flat* or *Text* is not empty.
Font	Name found in *Resources*	Font used for drawing the label in object.
Refresh	*TRUE, FALSE, var_name*	Continuous redraw of the object: - *FALSE*: the chart is updated only when opening page or when a child page is closed; - *TRUE*: the chart object is always updated, synchronized with others objects; - *var_name*: the chart is drawn on rising edge of boolean variable. This value *TRUE* of this variable is the moment when the char adds the samples. (Len Data <> internal HMI index of data).
Visible	*TRUE, FALSE, var_name*	Visible status of the object. It can be constant (*TRUE* or *FALSE*) or linked with a boolean variable *var_name*: if *var_name* is *TRUE* the object is visible, otherwise it is hidden.
Format X	String as *printf*	Format of X axis as c *printf* function.

Properties	Available values	Description
X Data	>=0, *var_name*	X-Axis array values. If there is a constant value in this property each Y sample has a X value equal to the product among X Data and the index in array. Ex. Y= track[3] = 20 X = 3* X Data
X Color	...	Color of X-Axis label.
Grid Step	Value	Space among two points of grid in pixel. The property is sensible if the grid is visible.
Sample Buffer	Value	Number of samples that the runtime can store. The older are deleted if the size has exceeded.
Grid Color	...	Color of grid if it is visible.
Bord. Color	...	Color of border grid.
Vertical Bar 1	>=0, *var_name*	Name of variable for drawing a vertical fixed bar on chart. It is allowed also a constant value. The * symbol means that there is not this vertical bar.
Color bar 1	...	Color of vertical bar 1 if different from *.
Vertical Bar 2		As *Vert. Bar 1*, but relative to bar 2.
Color bar 2		As *Color bar 1*, but relative to bar 2.
Vertical Bar 3		As *Vert. Bar 1*, but relative to bar 3.
Color bar 3		As *Color bar 1*, but relative to bar 3.
Clear Data	*var_name*	Boolean variable. If it is *TRUE* and *Refresh* is *TRUE* the chart deletes all the previous data.

5.14.2 EVENTS

Event	Description
BeforeUpdate	Before the object is redrawn.
AfterUpdate	Immediately after the object is redrawn.

5.15 TREND

5.15.1 PROPERTIES

Properties	Available values	Description
XPos	>= 0	Top-left 'x coordinate' edge relative to page.
YPos	>= 0	Top-left 'y coordinate' edge relative to page.
XDim	> 0	Width (#pixel).
YDim	> 0	Height (#pixel).
Name	Not empty	Name of object.

Properties	Available values	Description
Track 1	*var_name*	Open a dialog with the following options: - the variable will be sampled each *Sampling Time* seconds; - the visibility condition of the track (*TRUE* or boolean variable); - color of the track; - the scale factor (range among two horizontal divisions); - the offset (displacement of the track 0-Y); - step of print label for Y axis; - three horizontal bars with name, value and colors. If *var_name* is empty the track is not defined and not drawn.
Track 2	*var_name*	As *Track 1*, but for track 2.
Track 3	*var_name*	As *Track 1*, but for track 3.
Track 4	*var_name*	As *Track 1*, but for track 4.
Track 5	*var_name*	As *Track 1*, but for track 5.
Track 6	*var_name*	As *Track 1*, but for track 6.
Track 7	*var_name*	As *Track 1*, but for track 7.
Track 8	*var_name*	As *Track 1,* but for track 8.
Track Left	>=0, *var_name*	Integer value that is the track for Y axis left. It is admitted a constant value (ex. 1). If empty means that the chart must not draw the label for Y axis left.
Track Right	>=0, *var_name*	As *Track Left* for the right side.
Format Left	String as *printf*	Format of Y axis left as c *printf* function.
Format Right	String as *printf*	As *Format Left* for the right side.
XLabel	>=0, *var_name*	Step for X-axis labels. How many divisions of horizontal bar must have labels.
X Scale	*var_name*	Scale factor of x-Axis. Value range among two divisions of horizontal bars. An empty value indicates that the chart is in autoscale mode.
Sampling Time	>0	Sampling time measured in seconds. Every *Sampling Time* seconds the trend sample the value even if the chart is not shown.
X Offset	*var_name*	Variable name for the deviation of 0 for x-Axis, left or right. A positive value moves the chart values to left.
Grid	*Yes, No*	Visibility of the grid.
X Div. Grid	Value	Number of division on horizontal bar, used with scale factor and offset for drawing the chart tracks.
Y Div. Grid	Value	Number of division on vertical bar, used with scale factor and offset for drawing the chart tracks.
Background color	...	Background color selectable from palette.

Properties	Available values	Description
Appearance	*Flat, Raised, Sunken*	- *Flat*: plain with use of *Border pts* and *Border col*; - *Raised;* - *Sunken.*
Border points	>= 0	Border thickness (#pixel). This property is sensible only if *Appearance* is set to *Flat*.
Border color	...	Border color selectable from palette. This property is sensible only if *Appearance* is set to *Flat* or *Text* is not empty.
Font	Name found in *Resources*	Font used for drawing the label in object.
Refresh	*TRUE, FALSE, var_name*	Continuous redraw of the object: - *FALSE*: the chart is updated only when opening page or when a child page is closed; - *TRUE*: the chart object is always updated, synchronized with others objects; - *var_name*: the chart is drawn on rising edge of boolean variable. This value *TRUE* of this variable is the moment when the char adds the samples. (Len Data <> internal HMI index of data)
Visible	*TRUE, FALSE, var_name*	Visible status of the object. It can be constant (*TRUE* or *FALSE*) or linked with a boolean variable *var_name*: if *var_name* is *TRUE* the object is visible, otherwise it is hidden.
Format Time	*Default list*	Format for X-axis label. The choices are: - *ss*: seconds; - *mm.ss*: minutes, seconds; - *hh.mm*: hours, minutes; - *hh.mm.ss*: hours, minutes, seconds.
X Color	...	Color of X-Axis label.
Grid Step	Value	Space among two points of grid in pixel. The property is sensible if the grid is visible.
Sample buffer	Value	Number of samples that the runtime can store. The older are deleted if the size has exceeded.
Grid Color	...	Color of grid if it is visible.
Bord. Color	...	Color of border grid.
Vertical Bar 1	>=0, *var_name*	Name of variable for drawing a vertical fixed bar on chart. It is allowed also a constant value. The *** symbol means that there is not this vertical bar.
Color bar 1	...	Color of vertical bar 1 if different from *.
Vertical Bar 2		As *Vert. Bar 1*, but relative to bar 2.
Color bar 2		As *Color bar 1*, but relative to bar 2.

Properties	Available values	Description
Vertical Bar 3		As Vert. Bar 1, but relative to bar 3.
Color bar 3		As Color bar 1, but relative to bar 3.
Clear Data	var_name	Boolean variable. If it is TRUE and Refresh is TRUE the chart deletes all the previous data.

5.15.2 EVENTS

Event	Description
BeforeUpdate	Before the object is redrawn.
AfterUpdate	Immediately after the object is redrawn.

5.16 COMBO BOX

5.16.1 PROPERTIES

Properties	Available values	Description
Xpos	>= 0	Top-left 'x coordinate' edge relative to page.
Ypos	>= 0	Top-left 'y coordinate' edge relative to page.
Name	Not empty	Name of object.
Number of chars	> 0	Length of the object expressed in number of chars of the selected font.
Font	Name found in Resources	Font used for drawing the text in object.
Appearance	Flat, Raised, Sunken	- Flat: plain with use of Border points and Border color; - Raised; - Sunken.
Text color	...	Text color selectable from palette.
Background color	...	Background color selectable from palette.
Sel. Foreground	...	Text color selectable from palette when the object is chosen. This property is not sensible if the selectable field is constant FALSE.
Sel. Background	...	Background color selectable from palette when the object is chosen. This property is not sensible if the selectable field is constant FALSE.
Border Color	...	Border color selectable from palette. This property is sensible only if Appearance is set to Flat.
Border points	>= 0	Border thickness (#pixel). This property is sensible only if Appearance is set to Flat.

Properties	Available values	Description
Variable	*Not empty*	Name of the variable that can be shown and edited with this object. It can be any variable of the project, (local, global, imported from PLC or target - see 2.9.2), a parameter (see 2.9.2) or an element of a set (see 4.6).
Format	*.enum_name*	Enumerative, if in this field there is *enum_ name* defined in *Resources* (see 4.9).
Access	*RO, RW*	Accesses variable used in object: - *RO* = read only; - *RW* = read/write.
Refresh	*TRUE, FALSE, var_name*	Object redraw: - *FALSE*: the object value is read from memory and updated only when opening page or when a child page is closed. - *TRUE*: the object value is read from memory and always updated. - *var_name*: the object value is read from memory and updated only when the variable becomes *TRUE*.
Visible	*TRUE, FALSE, var_name*	Visible status of the object. It can be constant (*TRUE* or *FALSE*) or linked with a boolean variable *var_name*: if *var_name* is *TRUE* the object is visible, otherwise it is hidden.
Selectable	*TRUE, FALSE, var_name*	Selected status of the object. It can be constant (*TRUE* or *FALSE*) or linked with a boolean variable *var_name*: if *var_name* is *TRUE* the object is selected and so it will show the colors *Sel. Background*, *Sel. Foreground*. If this field is *FALSE* the *Access* property is not sensible.
Selection Order	*>= 0*	Selection order on which the object can be selected with the pressure of a key or with a procedure. In this case the selection moves from the current object to the previous or next *Sel. Order* object.

5.16.2 EVENTS

Event	Description
OnChange	This event is raised after the value is changed from selection.

5.17 CHECK BOX

5.17.1 PROPERTIES

Properties	Available values	Description
Xpos	>= 0	Top-left 'x coordinate' edge relative to page.
Ypos	>= 0	Top-left 'y coordinate' edge relative to page.
Xdim	> 0	Width (#pixel).
Ydim	> 0	Height (#pixel).
Name	Not empty	Name of object.
Appearance	*Flat, Raised, Sunken*	- *Flat*: plain with use of *Border points* and *Border color*; - *Raised*; - *Sunken*.
Color	...	Color selectable from palette of the check sign.
Background color	...	Color selectable from palette of the space around check sign.
Sel. Foreground	...	Check color selectable from palette when the object is chosen. This property is not sensible if the Selectable field is constant *FALSE*. It is valid only for targets that have joypad feature.
Sel. Background	...	Background color selectable from palette when the object is chosen. This property is not sensible if the Selectable field is constant *FALSE*. It is valid only for targets that have joypad feature.
Border color	...	Border color selectable from palette. This property is sensible only if *Appearance* is set to *Flat*.
Border Points	*>= 0*	Border thickness (#pixel). This property is sensible only if *Appearance* is set to *Flat*.
Access	*RO, RW*	Accesses variable used in object: - *RO* = read only; - *RW* = read/write. Read only means that the user cannot check the object.
Selection Order	*>= 0*	Selection order on which the object can be selected with the pressure of a key or with a procedure. In this case the selection moves from the current object to the previous or next *Sel. Order* object.
Variable	*Not empty*	Name of the variable that can be shown and edited with this object. It can be any variable of the project, (local, global, imported from PLC or target - see 2.9.2), a parameter (see 2.9.2) or an element of a set (see 4.6).

Properties	Available values	Description
Refresh	*TRUE, FALSE, var_name*	Object redraw: - FALSE: the object value is read from memory and updated only when opening page or when a child page is closed. - TRUE: the object value is read from memory and always updated. - var_name: the object value is read from memory and updated only when the variable becomes TRUE.
Visible	*TRUE, FALSE, var_name*	Visible status of the object. It can be constant (TRUE or FALSE) or linked with a boolean variable var_name: if var_name is TRUE the object is visible, otherwise it is hidden.
Selectable	*TRUE, FALSE, var_name*	Selected status of the object. It can be constant (TRUE or FALSE) or linked with a boolean variable var_name: if var_name is TRUE the object is selected and so it will show the colors Sel. Background, Sel. Foreground. If this field is FALSE the Access property is not sensible. It’s valid only for targets that have joypad feature.

5.17.2 EVENTS

Event	Description
OnClick	This event is raised before the value is changed from selection.