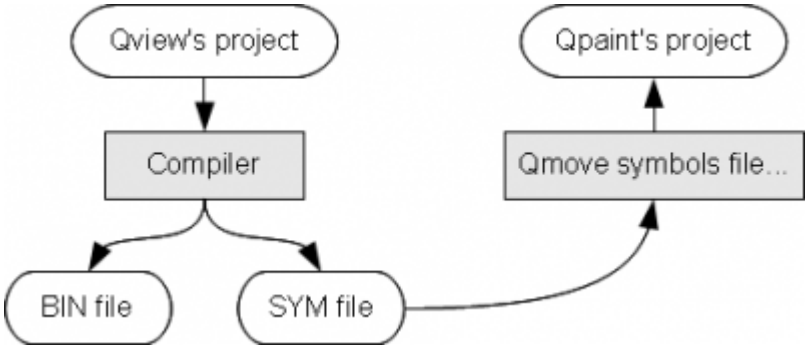
Sommario

QPAINT 5.1

0.1 Introduction

QPaint 5 is a graphics development environment for programming a QEM HMI terminal. This manual provides the main characteristics of a QPaint program. The description of the QPaint environment will often refer to concepts of the Qview development environment, which is the QEM software development environment for automation control.
The QPaint project can access all variables, parameters and other data structures declared in the Qview project. The “synchronisation” of the QPaint project with the Qview variables is represented in the figure below:



The compilation of the QView project generates a file with the same project name and “.sym” extension. The QPaint project can use this file to access all the data.
QPaint is a development environment that creates an operator interface without a programming language, using intuitive tools that give an immediate preview of the graphics in the end result.

0.2 Quick Start

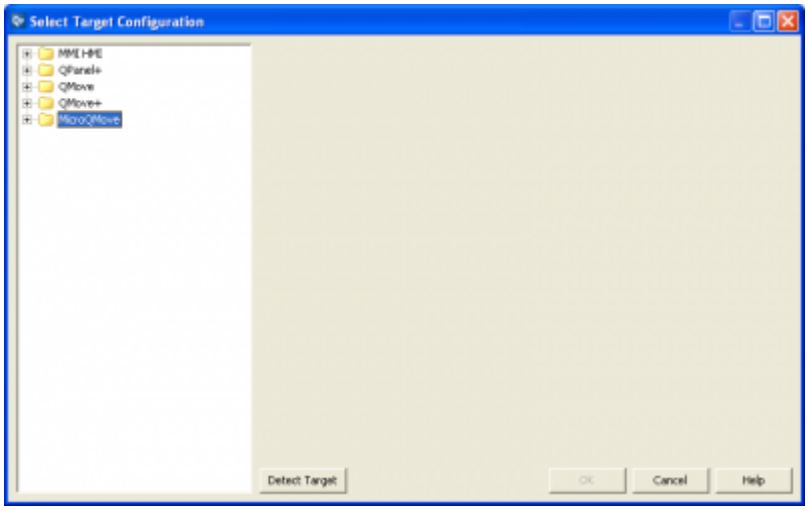
This quickstart gives a basic outline of information needed for a rapid startup of a new QPaint project .

0.2.1 New Project

Select:

* File - New Project...

QPaint asks the Qem hardware model for the Project:

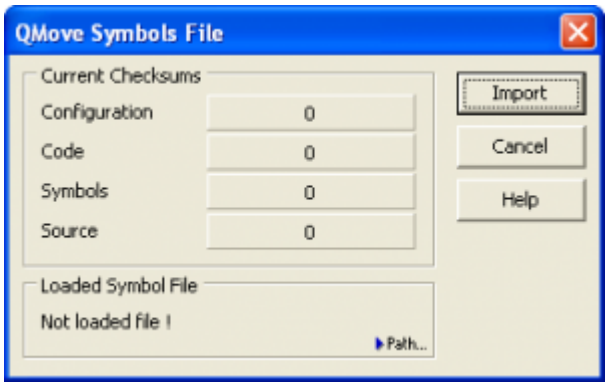


0.2.2 Symbols file importing

One first operation is to import the file that gives access to the symbols declared in the assiciated QView project. Select:

- Project - QMove Syumbols File...

To open the window:

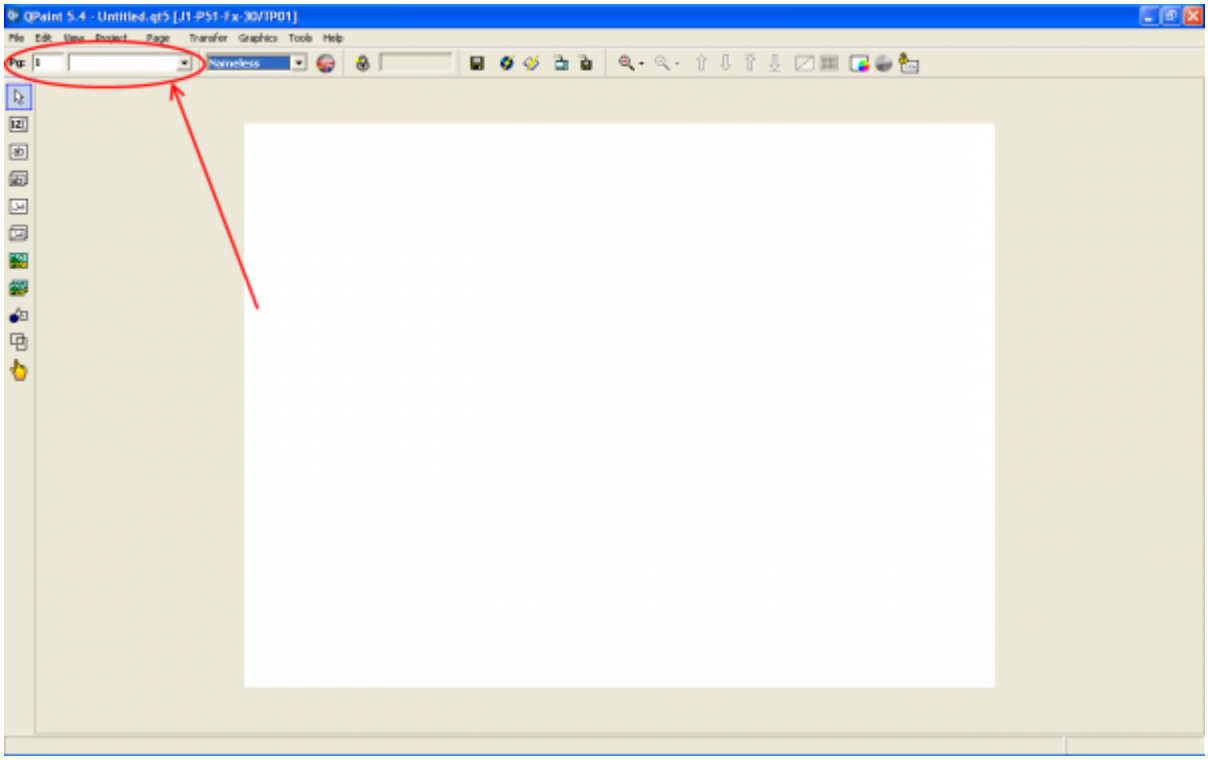


Click *Import* and specify the symbols file. This file is generated by QView during the project compilation and has the same name as the QView project, with a “.sym” extension.

0.2.3 FIXME Pages

A QPaint project is made up of a series of *Pages* for viewing on the HMI display, where the operator interacts. The list of pages, given in the top

lefthand corner, can be identified by a number or name:



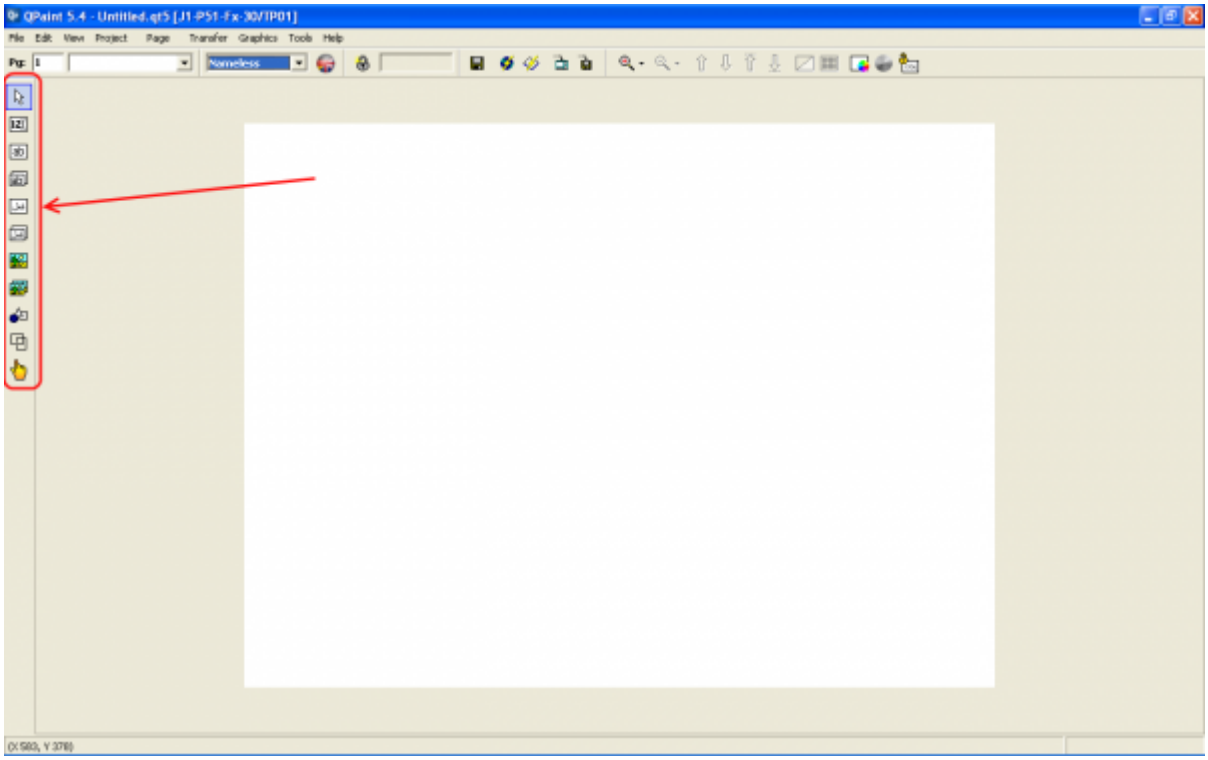
The *Page* menu Fix Me! manages the project pages Fix Me! :

- inserire, Fix Me!
- aggiungere,
- togliere,
- copiare,
- incollare,
- importare,
- esportare,
- impostare il colore dello sfondo,
- programmare gli eventi e le azioni di una pagina.

0.2.4 FIXME Oggetti

Graphic oggetti§ with various purposes can be added to the pages. To select what §oggetto to add to a §pagina use the vertical bar shown

below:



0.2.4.1 Value Object

The oggetto that views a variable or parameter value.

0.2.4.2 String Object

The oggetto used to show§write fixed§ text in the page.

0.2.4.3 ValString Object

The oggetto used to §apparire a dynamic message dependant on the value of a Qview variable.

0.2.4.4 UniString Object

The same oggetto as *String Object* but using PC fonts and especially Unicode fonts that implement different characters: Cyrillic, Greek, Arabic,

Chinese, etc.

0.2.4.5 UniValString Object

The same oggetto as *ValString Object* but with the same characteristics as *UniString Object*.

0.2.4.6 §Image Object

This oggetto adds an image to the page. The image must be taken from the project image library, accessed by selecting:

- Graphics - Image Manager

Image Manager can load images into the library and modify their dimensions.

0.2.4.7 ValImage Object

This oggetto adds a dynamic image to the page according to the value of a QView variable value. All images used in this type of oggetto must

have the same dimensions. The images must be taken from the image library in *Image Manager*.

0.2.4.8 Vector Image Object

This oggetto creates an area in the page for drawing elementary shapes. This oggetto must always be associated to a Word array. There are

Qview functions that provide basic drawing §primitive (e.g. lines, rectangles, arcs, circles, etc.).

0.2.4.9 Box Object

This oggetto creates rectangular frames 

boxes.

0.2.4.10 Touch Area

This oggetto creates a touch area that activates events 

legati to the user touching the screen. This oggetto can only be used on displays

with touch screens. The events:

- **On Touch Press**, the moment a finger 

presses the display
- **On Touch Release**, the moment the finger 

leaves the display
- **On Touch Press Double**, a rapid double press.

actions can be associated to each of these events. The actions are §tipo generico and not tied to touch-screen §concetto, so they will be

described later.

0.2.5 Events and Actions

§Oltre alla composizione della graphics, programming can also be implemented in the project. §Il principio è quello di avere a disposizione a series of *events* da cui scegliere. Ad ogni event io can be associated with *actions*. Moreover

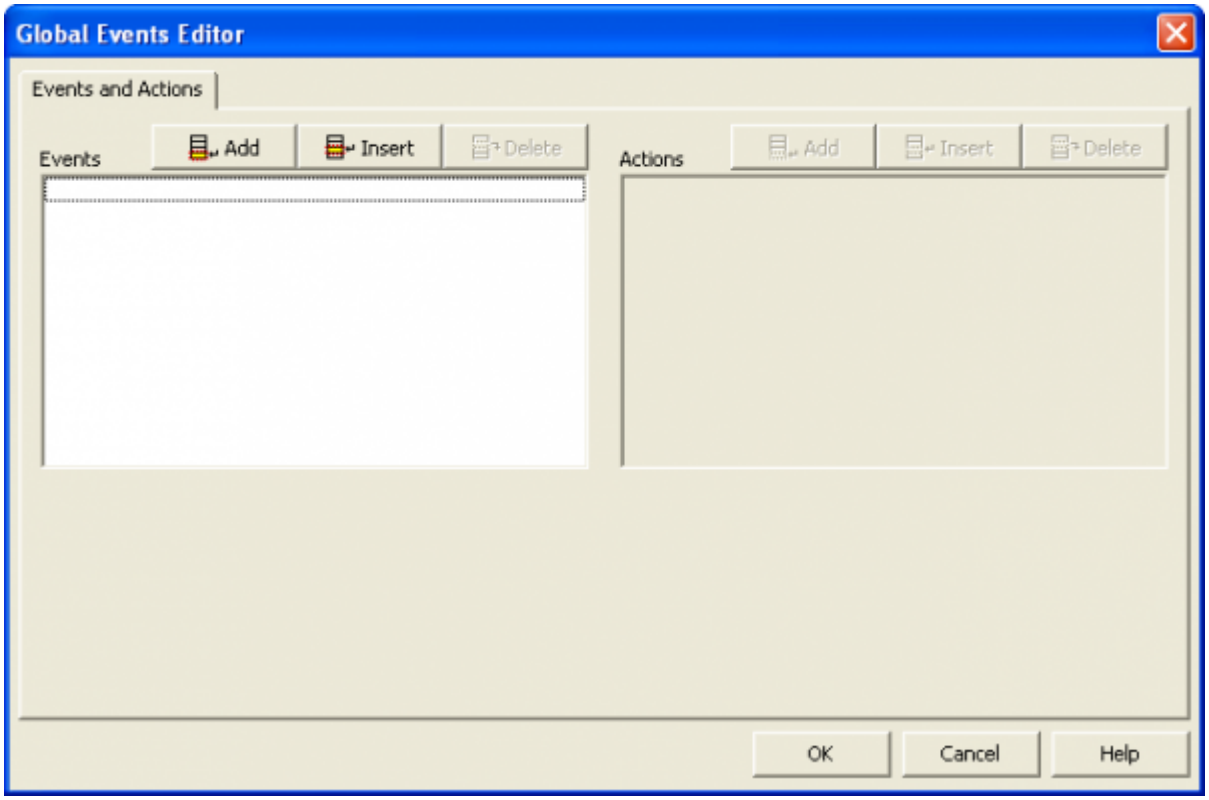
these *event-action* associations can be programmed so they are effective in all the pages or only in the page being viewed. For this there are

two 

contexts:

1. Project - Global Events Editor
2. Page - Page Events Editor

Both *Editors* have two areas as shown below



The *events* are entered on the left and the *actions* associated to a selected event are entered on the right .

0.2.5.1 Events

A list of the events:

- **On Key**, evento che si ripete continuamente finché è premuto il tasto specificato;
- **On Press**, evento che si attiva alla pressione del tasto specificato;
- **On Release**, evento che si attiva al rilascio del tasto specificato;
- **On Always**, evento che si ripete continuamente;
- **On Page In**, evento che si attiva all'ingresso della pagina;
- **On Time**, evento che si attiva in un certo momento specificato da una data e un'ora;
- **On Change Var**, evento che si attiva quando la variabile specificata cambia valore;
- **On Var**, evento che si attiva quando la variabile specificata assume il valore specificato.

0.2.5.2 Actions

An *event* can be associated with to several of the following *actions*:

- **Goto Page**
- **Next Page**
- **Previus Page**
- **Begin Data Entry**
- **Send Command** Fix Me! ad un device dichiarato nel progetto QView;
- **Set Variable** assegna un valore ad una variabile del progetto QView;
- **Led On**, attiva il LED specificato (usually the LED's are associated to function keys);
- **Led Off**, disattiva il LED specificato;
- **Led Blink**, attiva il lampeggio del LED specificato;
- **Backup**, invia un comando di Backup (verificare se questo comando è supportato dall'hardware);
- **Restore**, invia un comando di Restore (verificare se questo comando è supportato dall'hardware).

0.2.6 FIXME Terminal Variable

Ogni progetto QPaint project contains variables sempre presenti che supply information related to the actual §terminale. These variables have a pre-set name, some are only read and others are also write:

Name	Type	Read/Write	Description
`$KEY`	L	R	Fix Me! Codice del tasto premuto. Ad ogni bit corrisponde un tasto della tastiera.
`$KEYF`	L	R	Codice del tasto funzione premuto. Ad ogni bit corrisponde un tasto della tastiera.
`$KEYF2`	L	R	Codice del tasto funzione premuto (Secondo gruppo). Ad ogni bit corrisponde un tasto della tastiera.
`$LEDS`	L	RW	Codice per attivare/disattivare i LED. Ad ogni bit corrisponde un LED.
`$LEDS2`	L	RW	Codice per attivare/disattivare i LED (Secondo gruppo). Ad ogni bit corrisponde un LED.
`$HOUR, $MIN, $SEC, $DAY, $MONTH, $YEAR`	B, B, B, B, B, W	RW	Ora e data.
`$DATAENTRYON`	F	R	Segnala che in questo momento è attivo l'inserimento di un valore
`$PAGE`	L	R	Valore della pagina visualizzata in questo momento.

There are many others, since these are only the most common when developing a project. (see Fix Me!)

1. Controls by QView

Fix Me!

The management of the HMI (human-machine interface) can also be achieved by an exchange of information with a QView project. These are the

most important operations in managing an HMI:

- detect the pressure on a button Fix Me!
- understand what screen is being viewed
- command a screen change
- verify if the data entry is Fix Me! active (the page change is diabled).

Often it is best to manage the HMI basic operazioni directly from the QView project. Vediamo come questo sia possibile.

1.1 Detect the FIXME pressure on a button

The variables to know if a button has been pressed are

```
$KEY, $KEYF, $KEYF2.
```

Each bit of these variables is assigned to a

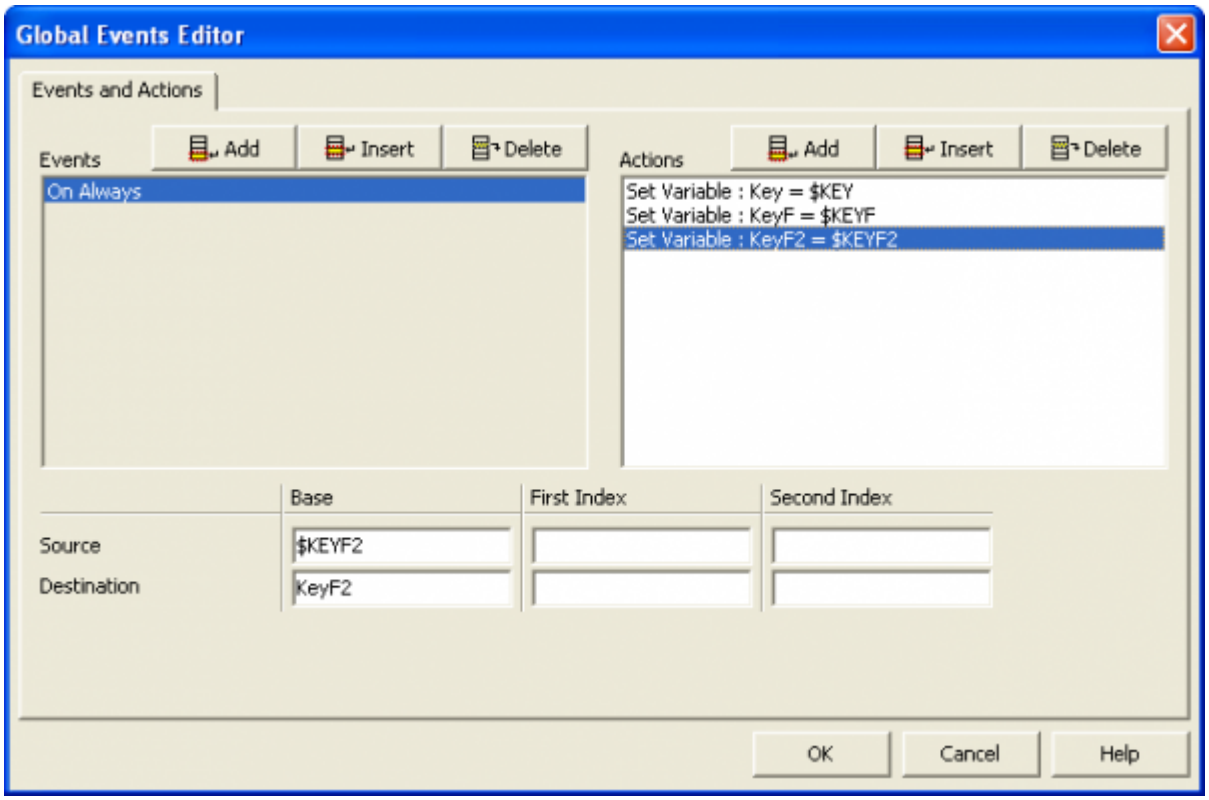
button Fix Me! della tastiera. The variables are only accessible by QPaint and are not Fix Me! visibili in the QView project. To be able to transfer

the variable values to the QView project, three variables have to be included in the QView project:

```
GLOBAL
  Key    L   ;This will be match to $KEY
  KeyF   L   ;This will be match to $KEYF
  KeyF2  L   ;This will be match to $KEYF2
```

In the QPaint project select:

- Project - Global Events Editor



This will make the QPaint project continously copy (*OnAlways*) the values of its three variables Fix Me! in§ the Qview variables. The QView

project can declare the constants related to the §pressione of each button:

```
CONST
KEY_1       268435456    ;  "1"
KEY_2       1048576      ;  "2"
KEY_3       4096         ;  "3"
KEY_4       536870912    ;  "4"
KEY_5       2097152      ;  "5"
KEY_6       8192         ;  "6"
KEY_7       1073741824   ;  "7"
KEY_8       4194304      ;  "8"
KEY_9       16384        ;  "9"
KEY_0       8388608      ;  "0"
KEY_CLR     -2147483648  ;  "CLR"
KEY_ENTER   128          ;  "ENTER"
KEY_HELP    64           ;  "HELP"
KEY_DECPT   32           ;  "."
KEY_SIGN    16           ;  "+/-"
KEY_ESC     33554432     ;  "ESC"
KEY_UP      131072       ;  "UP"
KEY_PGUP    512          ;  "PGUP"
KEY_LEFT    67108864     ;  "LEFT"
KEY_NEXT    262144       ;  "NEXT"
KEY_RIGHT   1024         ;  "RIGHT"
KEY_INS     134217728    ;  "INS"
KEY_DOWN    524288       ;  "DOWN"
KEY_PGDN    2048         ;  "PGDN"

KEY_F1      33554432     ; "F1"
KEY_F2      67108864     ; "F2"
KEY_F3      134217728    ; "F3"
KEY_F4      268435456    ; "F4"
KEY_F5      536870912    ; "F5"
KEY_F6      131072       ; "F6"
KEY_F7      262144       ; "F7"
KEY_F8      524288       ; "F8"
KEY_F9      1048576      ; "F9"
KEY_F10     2097152      ; "F10"
KEY_F11     1          ; "F11"
KEY_F12     2          ; "F12"
KEY_F13         4        ; "F13"
KEY_F14         8        ; "F14"
KEY_F15         16       ; "F15"
KEY_F16         32       ; "F16"
KEY_F17         64       ; "F17"
KEY_F18         128      ; "F18"
KEY_F19         256      ; "F19"
KEY_F20         512      ; "F20"
KEY_F21     1024         ; "F21"
KEY_F22     2048         ; "F22"
KEY_F23     4096         ; "F23"
KEY_F24     8192         ; "F24"
KEY_F25     16384        ; "F25"
KEY_F26     32768        ; "F26"
```

The code to execute a given action if a tasto Fix Me! is pressed:

```
IF Key EQ KEY_1
  ;Put here the code to do when the operator press "1" key
ENDIF
```

1.2 Knowing the page §visualizzata

As in the previous chapter, a variable has to be included in the Qview project to be able to accogliere the

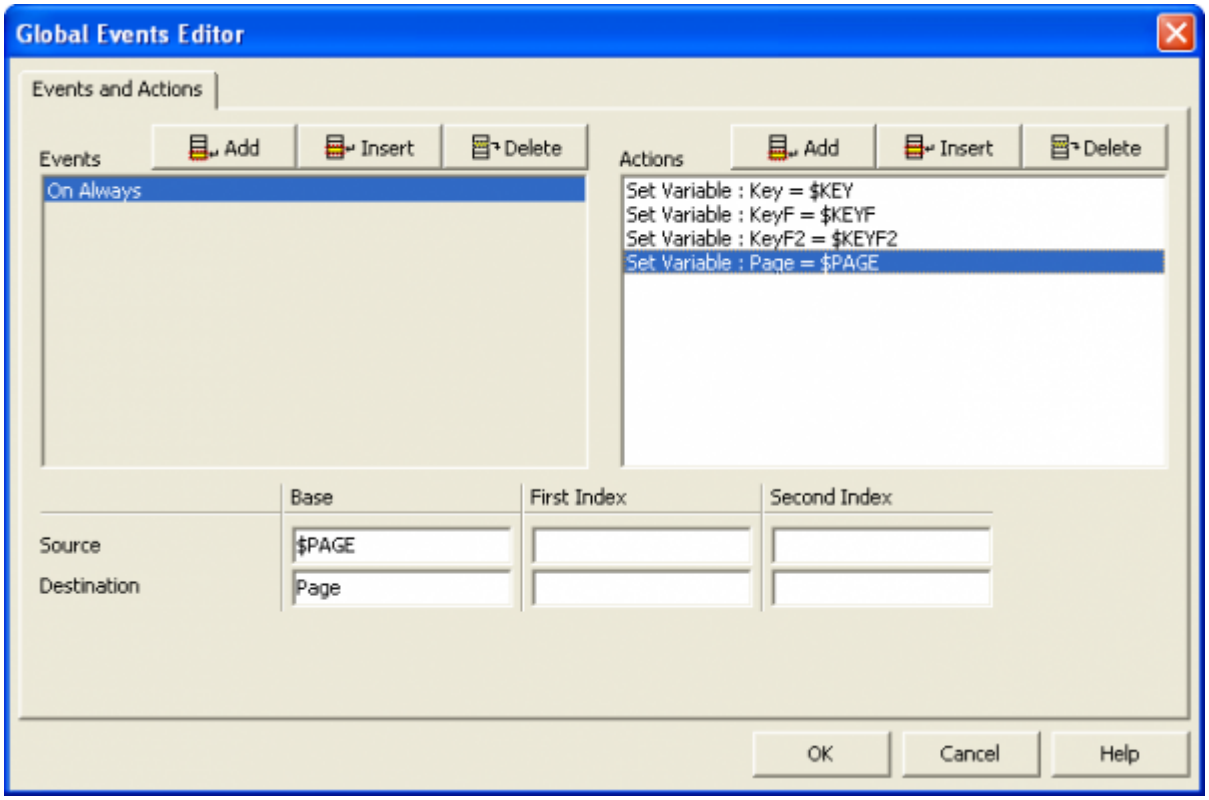
```
$PAGE
```

variable

of the QPaint project. §Quindi dichiaro:

```
GLOBAL
  Page  L  ;This will be match§ to $PAGE
```

e poi aggiungo an action associated to the *OnAlways* event:



§In questo modo posso eseguire different operations depending on the page being viewed:

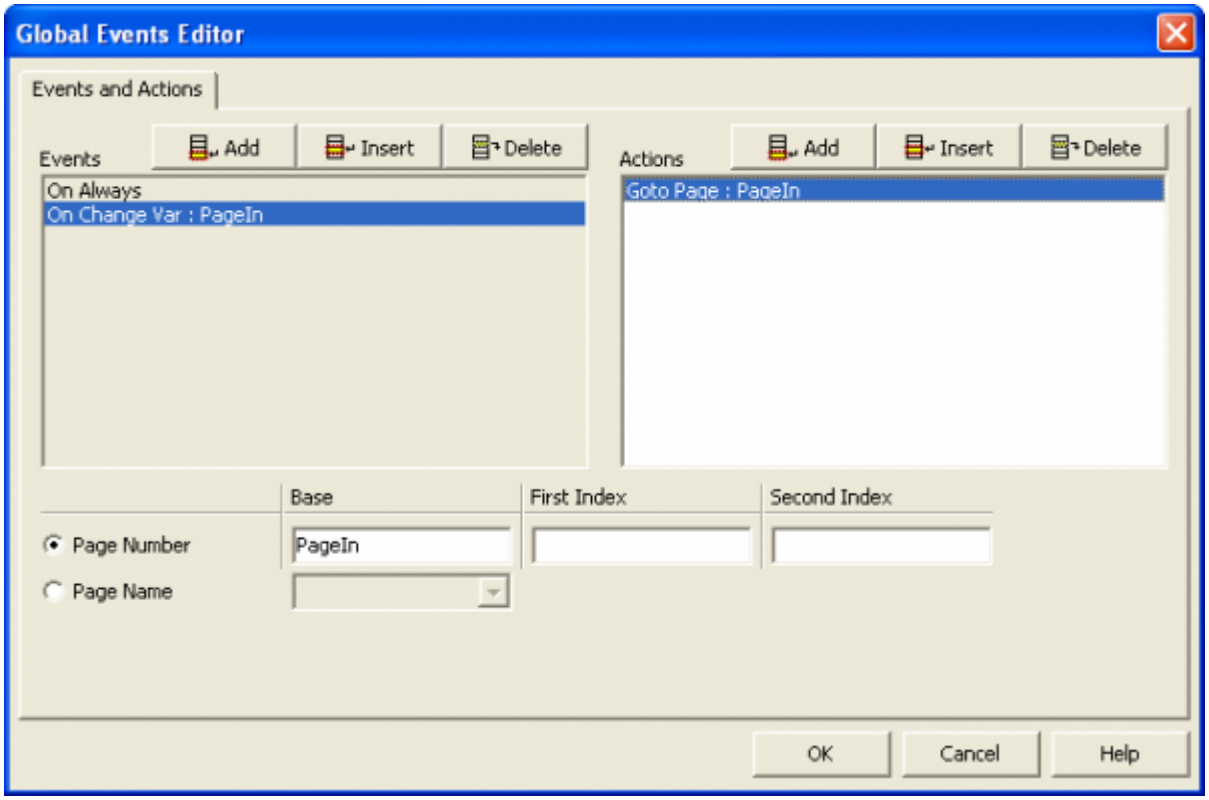
```
IF Page EQ MENU_PAGE
  IF Key EQ KEY_1
    ;Put here the code to do when the operator press "1" key on the "MENU_PAGE" page
  ENDIF
ENDIF
```

1.3 Commanding change page

To command a page change on the HMI by a single code line in the QView project, first declare a Qview variable:

```
GLOBAL
  PageIn  L  ;This will be used to request the page change
```

then add a *OnChangeVar* event as shown in the figure:



So when the value of *PagIn* variable is changed in a code line of the Qview project, the *Goto Page* action is activated and changes the page,

viewing the page Fix Me! con indice il new value of the *PageIn* variable.

The new QCL code:

```
IF Page EQ MENU_PAGE
  IF Key EQ KEY_1
    PageIn = PAGE_1       ;Change page request: go to "PAGE_1" page
    WAIT(Page EQ PAGE_1)  ;Wait the change page has done
  ENDIF
ENDIF
```

It can be seen that a *WAIT* instruction is added to wait until the requested page change is effectively made before continuing with the

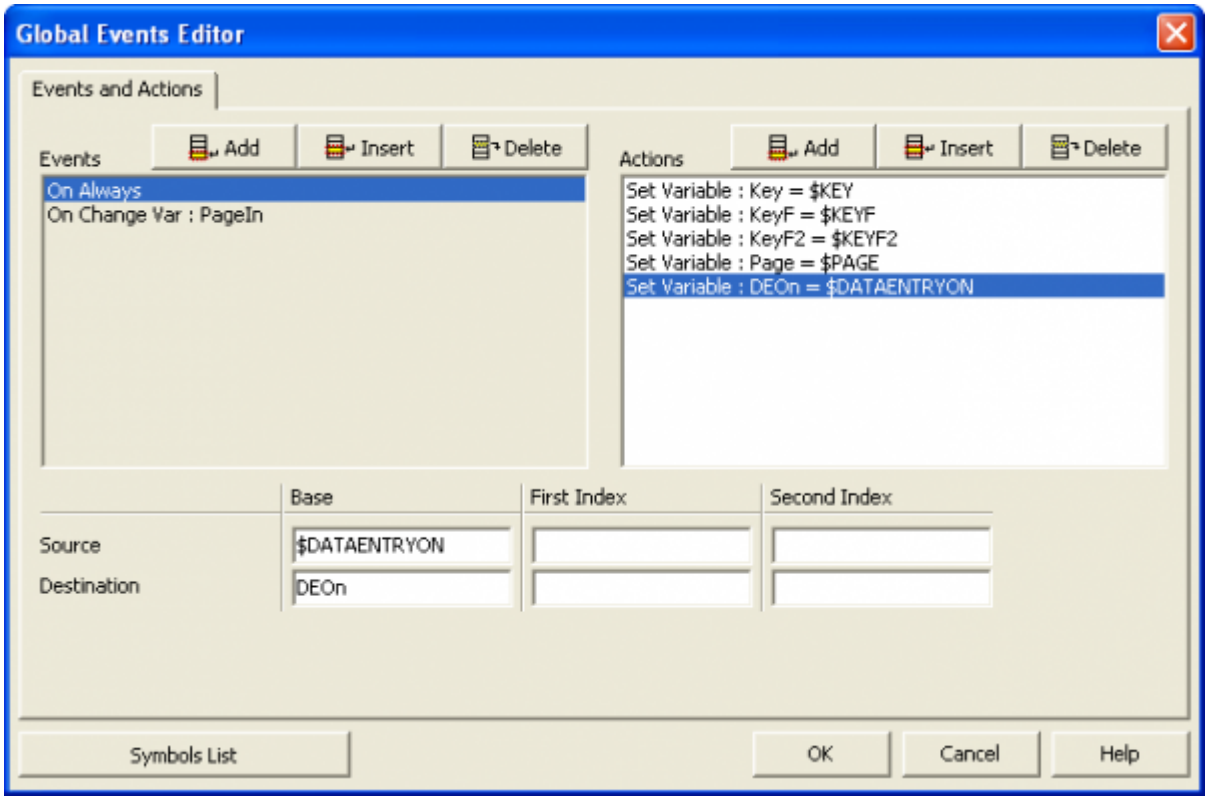
remaining code lines.

1.4 Dataentry active

To know the status of a data entry another variable must be included in the QVIEW project:

```
GLOBAL
  DEOn  F  ;This will match FIXME to $DATAENTRYON
```

add another action associated to the *OnAlways* event:



then if the page change is not possible during the data entry, the code must be changed:

```
IF Page EQ MENU_PAGE
  IF (Key EQ KEY_1) AND (DEOn EQ 0)
    PageIn = PAGE_1       ;Change page request: go to "PAGE_1" page
    WAIT(Page EQ PAGE_1)  ;Wait the change page has done
  ENDIF
ENDIF
```