Table of Contents

1 Introduction

Keyboard Builder allows to design on screen keyboards for touch panel applications.

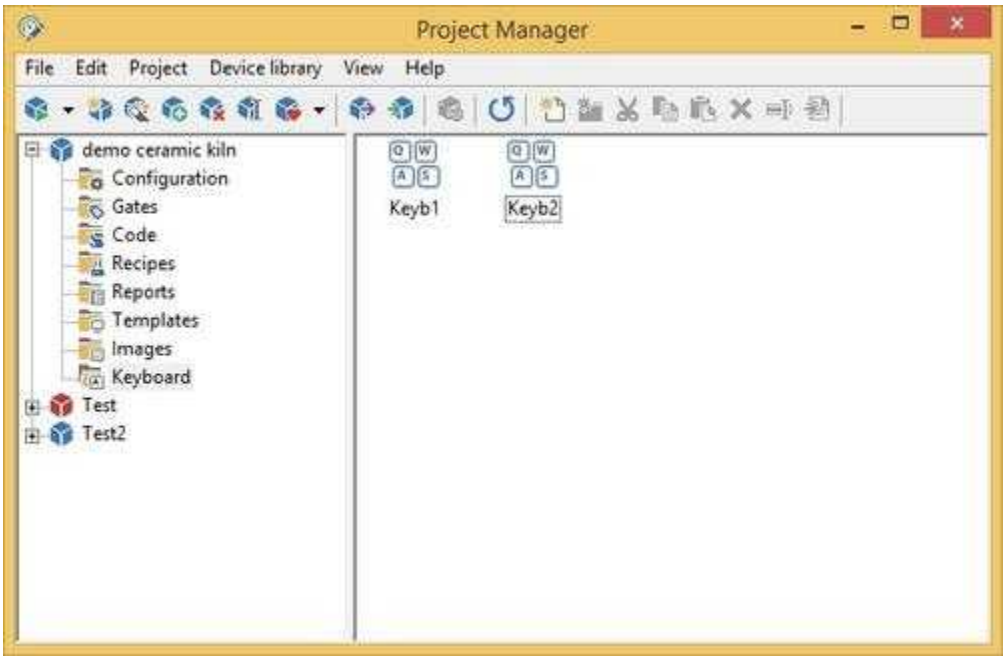
2 Examples

Here some examples:

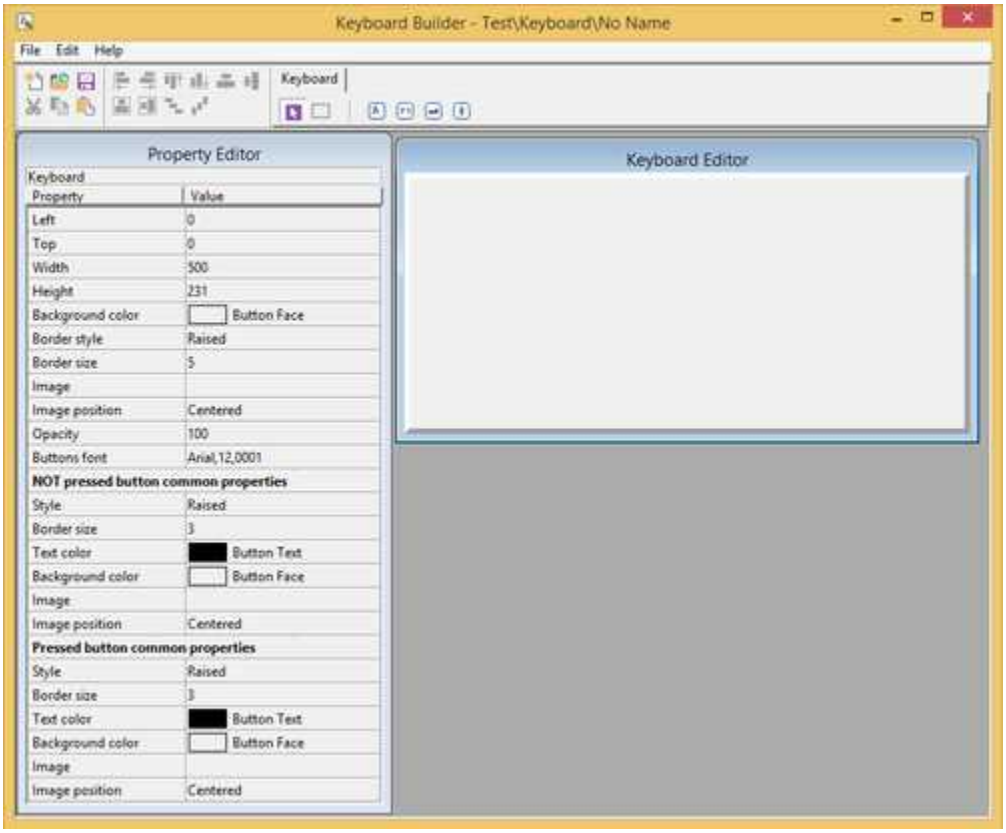
3 Program start

The creation of a new blank Keyboard is performed in the *Project Manager*, by selecting the *Keyboard* folder and then moving the mouse on the right window and pressing the right button: the *"New | Keyboard"* menu entry is then displayed

Keyboard Builder is started automatically by the *Project Manager* by double clicking on the icon to any Keyboard previously created.



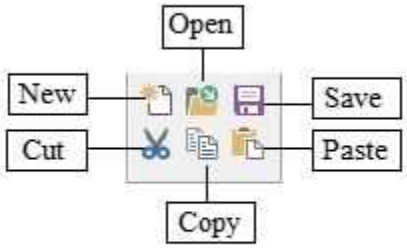
In figure below it is shown the *Keyboard Builder* working environment at the start (no keyboard loaded). You may note three main sections: at the top the tool bar, on the left the property editor, and on the right the Keyboard layout window.

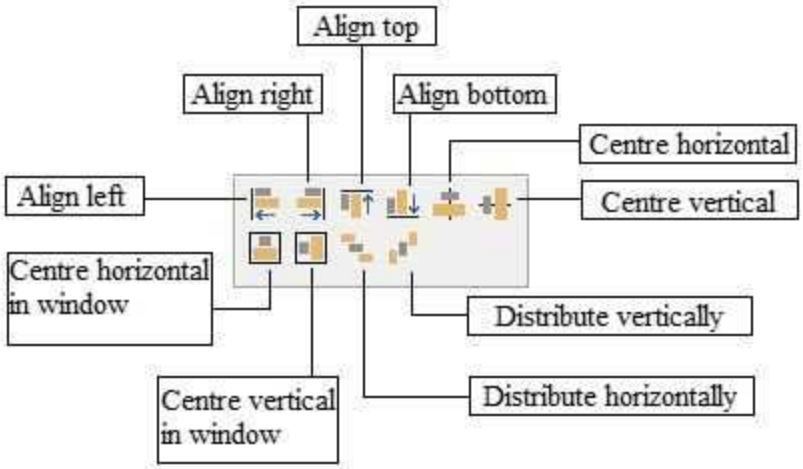


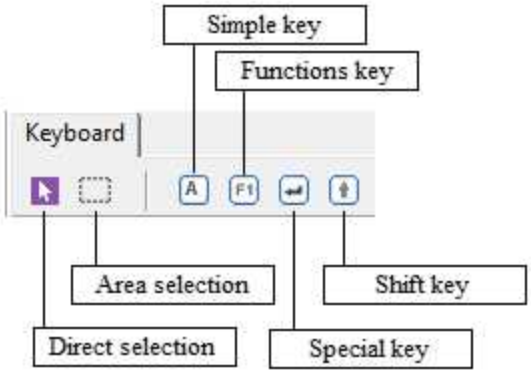
4 Tool bar

In figure there is a detailed description of the tool bar: it is divided into two parts, the one on the left contains the commands to create a new keyboard (function that can be called up also from the menu *File* | *New*), to open a keyboard (*File* | *Open…*), to save the current keyboard (*File* | *Save*) and to use the clipboard (functions that can be called up also from the menu *Edit* | *Cut*, *Copy*, *Paste*); while the part on the right contains the buttons to select the objects to be placed in the keyboard layout window. To place an object in the keyboard layout, you just need to press the corresponding button on the tool bar and click on the keyboard window in the desired position: the new component will be created and automatically selected, so as to be able to move and modify it in the desired manner (as described below).

Note that when a new component is created in the keyboard window , the button that had been selected on the tool bar, is deselected, and the button on its left is selected (the arrow).When this button is pressed, it is possible to select and modify the components already present in the keyboard layout by clicking on them, without creating new ones.



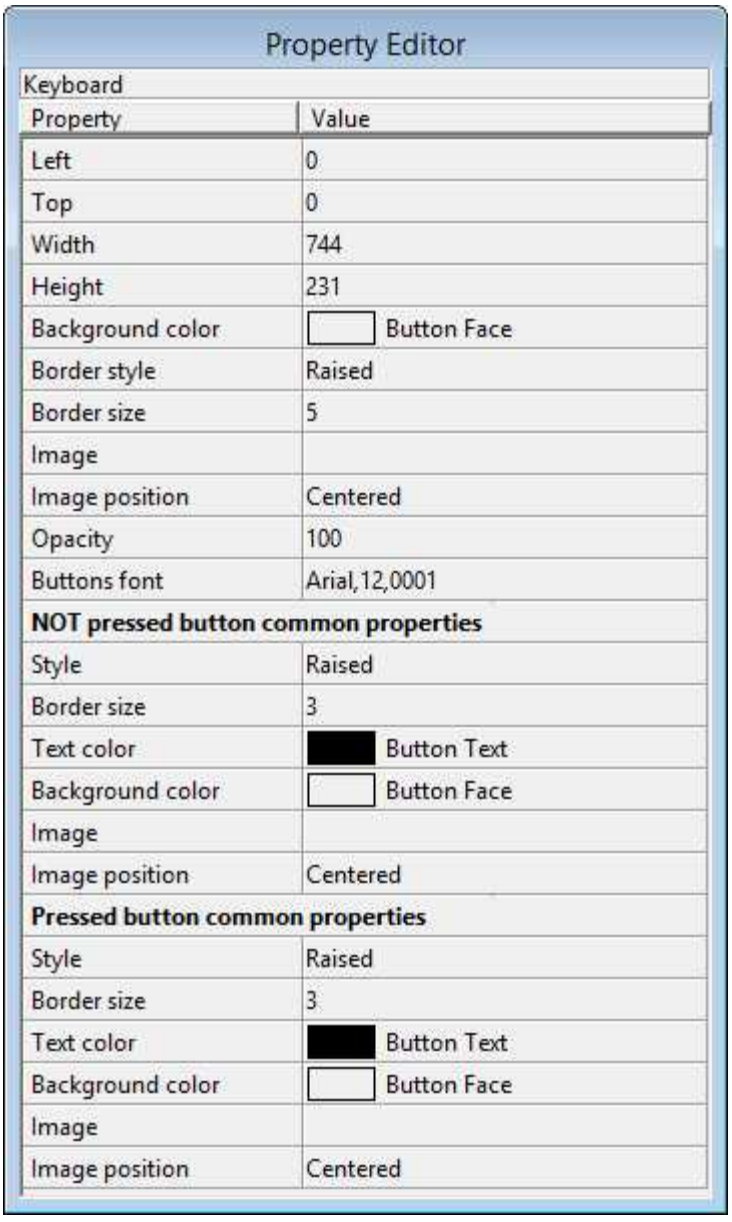




5 Property editor

Every component of keyboard layout has several properties. These properties can be modified at will, and *property editor* does this. Selecting a component in the keyboard layout, in the *property editor* will appear its properties. If, on the contrary, you select the keyboard layout window, in the *property editor*

will appear the rows with the keyboard properties. To modify the shown properties, first of all you have to position the selection on the desired row (with the mouse or the cursor keys). On the left part of the selected row (which looks lowered) there is a label indicating the name of the property, while on the right part there is the present value of the property. Basically there are three types of rows in the *property editor*: the edit rows (figure A), the multiple choice rows (figure B) and the rows with button (figure C).



With the first type of rows it is possible to modify the property straight from the *property editor*: you just need to write with the keyboard the value of the property in the edit square. The multiple choice rows allow to select the value of the property from a list that is shown by pressing with the mouse the small button to the right of the line. As far as the button rows are concerned, the matter is different: in the row it is shown the present value of the property, but to modify it you have to press the button on the right with the mouse .Then a specific window for the property will be opened. In it the present settings

of the property are shown in detail. So it will be possible to modify the property by acting on the parameters of this window, and then confirm the choices by pressing the *Ok* button: the window will be closed and the new settings of the property will be shown in the row of the *property editor*.

Every time that a property is modified using the *property editor*, it is possible to undo the change simply pressing the ESC key before selecting a new line.

Note that every time that in the *property editor* you modify a property concerning in some way the display of the component (for example the Color of a component, its font, its size and so on), the component selected in the template is updated consistently with the new settings of the property. The same can be said also for the opposite. As you'll see in the next paragraph, it is possible to modify the properties of a component straight from the keyboard layout (usually its position and size): in such cases the data displayed in the *property editor* will be updated accordingly.

6 Keyboard layout window

The keyboard layout window shows a draft of the user keyboard .

In this window will be placed all the components (Button) forming the keyboard, by simply selecting the desired component from the tool bar and clicking in the keyboard layout window. The object just inserted in the keyboard will be selected; so it will be soon and easily identified. When a component is selected, *Keyboard Builder* will place eight small black squares along the object outline, and in the *property editor* all its properties will be shown. The selected object is the one in which to carry out all the changes specified in the *property editor*. To select an object, you just need to click on it with the mouse: the selection will be removed from the component previously selected, and placed on the new component, and the properties of the *property editor* will be updated consistently with the new selected object.

As said before, it is possible to change the position and the size of the component straight from the keyboard layout window, without using the *property editor*.

To move an object, you just need to click on it an drag it in the new position, then release the mouse left button: while you are dragging it, you will see the object follow the mouse pointer, and when it is released you will note that the *property editor* rows corresponding to the position are updated.

As easily, you can change the size of the selected object: you just need to click on and drag one of the eight small squares forming the selection, and release it in the desired position. You will notice that the object will change its size according to your wishes and that the *property editor* rows will be updated when the mouse is released.

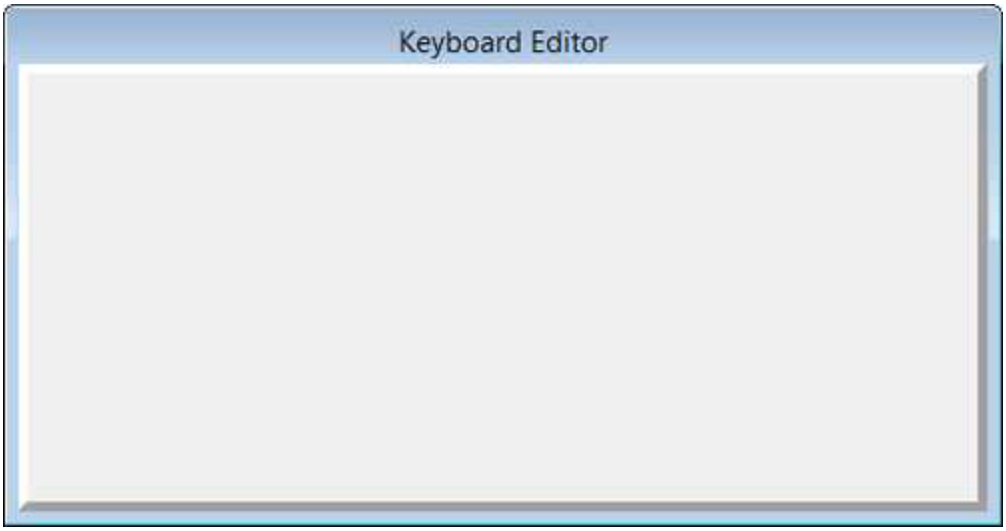
To make easier the positioning of the objects in the template, its position and size cannot have a value whatsoever. Normallly, in fact, position and size can only have values that are multiples of five: this way it is easier to line up the objects consistently in all the template. If you want to position or resize an object more precisely, it is possible to specify the desired position or size in the *property editor*, or, as an alternative, drag or resize the object by holding down the CTRL key: the object will be moved or resized pixel by pixel, and not any longer by five pixels at a time.

The moving and resizing operations can also be carried out using the keyboard: to move the selected object you just need to use the cursor keys, while to resize it you have to do the same while you hold down the SHIFT key. Also for the moving and resizing operations with the keyboard you can hold down CTRL key in order to allow movements of one pixel.

7 Description of the components

7.1 Keyboard

It is the container where will be put the buttons.



Left - horizontal position of the top left corner of the keyboard (in pixels).

Top - vertical position of the top left corner of the keyboard (in pixels).

Width - width of the keyboard (in pixels).

Height - height of the keyboard (in pixels).

Background color - background color of the keyboard.

Border style - border type of the keyboard (details).

Border size - size (in pixel) of the border of the keyboard.

Image - eventual background image of the keyboard.

Image position - image positioning mode (details)

Opacity - opacity percentage of the keyboard (details).

Buttons font - common font that will be associated to all buttons that have the "Buttons font" property inherited from the common properties.

NOT pressed button common properties

Style - border type (details) that will be associated to the "NOT PRESSED" status of all buttons that have the "Style" property inherited from the common properties.

Border size - button border size (in pixel) that will be associated to the "NOT PRESSED" status of all buttons that have the "Border size" property inherited from the common properties.

Text color - text color that will be associated to the "NOT PRESSED" status of all buttons that have the "Text color" property inherited from the common properties

Background color - background color that will be associated to the "NOT PRESSED" status of all buttons that have the "Background properties" inherited from the common properties

Image - image that will be associated to the "NOT PRESSED" status of all buttons that have the "Background properties" inherited from the common properties

Image position - image positioning mode (details) that will be associated to the "NOT PRESSED" status of all buttons that have the "Background properties" inherited from the common properties

Pressed button common properties

Style - border type (details) that will be associated to the "PRESSED" status of all buttons that have the "Style" property inherited from the common properties.

Border size - button border size (in pixel) that will be associated to the "PRESSED" status of all buttons that have the "Border size" property inherited from the common properties.

Text color - text color that will be associated to the "PRESSED" status of all buttons that have the "Text color" property inherited from the common properties

Background color - background color that will be associated to the "PRESSED" status of all buttons that have the "Background properties" inherited from the common properties

Image - image that will be associated to the "PRESSED" status of all buttons that have the "Background properties" inherited from the common properties

Image position - image positioning mode (details) that will be associated to the "PRESSED" status of all buttons that have the "Background properties" inherited from the common properties

7.2 Simple key

Simple Key component is used in the keyboard to send standard characters (like 0...9 or A..Z or !"£$%& ....) to the application .

Left - horizontal position of the top left corner of the button (in pixels).

Top - vertical position of the top left corner of the button (in pixels).

Width - width of the button (in pixels).

Height - height of the button (in pixels).

Font - font to use for the text. Push the button on the properties line to show the dialog for the choice of the font, size, style (normal, bold or italic) and effects (underlined and strikeout)
.
Key - char that will be sent when the button is pressed and the SHIFT button is NOT pressed.

Shift+Key - char that will be sent when the button is pressed whit the SHIFT button pressed.

Close keyboard - if it is set to "Yes" then when the button is pressed, the key will be sent and the keyboard will be automatically closed.

Inherits from common properties

Buttons font - if set to "Yes" then font will be inherited from common properties.

Style - if set to "Yes" then button style (PRESSED and NOT PRESSED) will be inherited from common properties.

Border size - if set to "Yes" then button border size (PRESSED and NOT PRESSED) will be inherited from common properties.

Text color - if set to "Yes" then button text color (PRESSED and NOT PRESSED) will be inherited from common properties.

Background properties - if set to "Yes" then background color, image and image position (PRESSED and NOT PRESSED button) will be inherited from common properties.

NOT pressed button properties

Style - border type (details) that will be associated to the "NOT PRESSED" status.

Border size - size (in pixel) of the border that will be associated to the "NOT PRESSED" status.

Text color - text color that will be associated to the "NOT PRESSED" status.

Background color - background color that will be associated to the "NOT PRESSED" status.

Image - image that will be associated to the "NOT PRESSED" status.

Image position - image position mode (details) that will be associated to the "NOT PRESSED" status.

Pressed button properties

Style - border type (details) that will be associated to the "PRESSED" status.

Border size - size (in pixel) of the border that will be associated to the "PRESSED" status.

Text color - text color that will be associated to the "PRESSED" status.

Background color - background color that will be associated to the "PRESSED" status.

Image - image that will be associated to the "PRESSED" status.

Image position - image position mode (details) that will be associated to the "PRESSED" status.

7.3 Function key

Function Key component is used in the keyboard to send function keys (F1...F12) to the application .

Left - horizontal position of the top left corner of the button (in pixels).

Top - vertical position of the top left corner of the button (in pixels).

Width - width of the button (in pixels).

Height - height of the button (in pixels).

Font - font to use for the text. Push the button on the properties line to show the dialog for the choice of the font, size, style (normal, bold or italic) and effects (underlined and strikeout)
.
Text to send - text that will be sent when the button is pressed (F1...F12).

Foreground - define what to show in foreground **: None**, **Text label**, **Predefined image**, **Custom image** (details).

Text to show - text to show in case of "Foreground = Text label" .

Foreground image - image to show in case of "Foreground = Custom image" .

Close keyboard - if it is set to "Yes" then when the button is pressed, the key will be sent and the keyboard will be automatically closed.

Inherits from common properties

Buttons font - if set to "Yes" then font will be inherited from common properties.

Style - if set to "Yes" then button style (PRESSED and NOT PRESSED) will be inherited from common properties.

Border size - if set to "Yes" then button border size (PRESSED and NOT PRESSED) will be inherited from common properties.

Text color - if set to "Yes" then button text color (PRESSED and NOT PRESSED) will be inherited from common properties.

Background properties - if set to "Yes" then background color, image and image position (PRESSED and NOT PRESSED button) will be inherited from common properties.

NOT pressed button properties

Style - border type (details) that will be associated to the "NOT PRESSED" status.

Border size - size (in pixel) of the border that will be associated to the "NOT PRESSED" status.

Text color - text color that will be associated to the "NOT PRESSED" status.

Background color - background color that will be associated to the "NOT PRESSED" status.

Image - image that will be associated to the "NOT PRESSED" status.

Image position - image position mode (details) that will be associated to the "NOT PRESSED" status.

Pressed button properties

Style - border type (details) that will be associated to the "PRESSED" status.

Border size - size (in pixel) of the border that will be associated to the "PRESSED" status.

Text color - text color that will be associated to the "PRESSED" status.

Background color - background color that will be associated to the "PRESSED" status.

Image - image that will be associated to the "PRESSED" status.

Image position - image position mode (details) that will be associated to the "PRESSED" status.

7.4 Special key

Special Key component is used in the keyboard to send special keys to the application .

Supported special keys are: **Esc**, **Enter**, **Tab**, **Backspace**, **Home**, **End**, **Del**, **Ins**, **Up**, **Down**, **Left**, **Right, Close**

Left - horizontal position of the top left corner of the button (in pixels).

Top - vertical position of the top left corner of the button (in pixels).

Width - width of the button (in pixels).

Height - height of the button (in pixels).

Font - font to use for the text. Push the button on the properties line to show the dialog for the choice of the font, size, style (normal, bold or italic) and effects (underlined and strikeout)
.
Text to send - text that will be sent when the button is pressed .

Foreground - define what to show in foreground **: None**, **Text label**, **Predefined image**, **Custom image** (details).

Text to show - text to show in case of "Foreground = Text label" .

Foreground image - image to show in case of "Foreground = Custom image" .

Close keyboard - if it is set to "Yes" then when the button is pressed, the key will be sent and the keyboard will be automatically closed.

Inherits from common properties

Buttons font - if set to "Yes" then font will be inherited from common properties.

Style - if set to "Yes" then button style (PRESSED and NOT PRESSED) will be inherited from common properties.

Border size - if set to "Yes" then button border size (PRESSED and NOT PRESSED) will be inherited from common properties.

Text color - if set to "Yes" then button text color (PRESSED and NOT PRESSED) will be inherited from common properties.

Background properties - if set to "Yes" then background color, image and image position (PRESSED and NOT PRESSED button) will be inherited from common properties.

NOT pressed button properties

Style - border type (details) that will be associated to the "NOT PRESSED" status.

Border size - size (in pixel) of the border that will be associated to the "NOT PRESSED" status.

Text color - text color that will be associated to the "NOT PRESSED" status.

Background color - background color that will be associated to the "NOT PRESSED" status.

Image - image that will be associated to the "NOT PRESSED" status.

Image position - image position mode (details) that will be associated to the "NOT PRESSED" status.

Pressed button properties

Style - border type (details) that will be associated to the "PRESSED" status.

Border size - size (in pixel) of the border that will be associated to the "PRESSED" status.

Text color - text color that will be associated to the "PRESSED" status.

Background color - background color that will be associated to the "PRESSED" status.

Image - image that will be associated to the "PRESSED" status.

Image position - image position mode (details) that will be associated to the "PRESSED" status.

7.5 Shift key

Shift Key component is used to simulate the shift key status .
When it is pressed a *Simple key* button, if the *Shift key* button is pressed, will be sent to the application the "Shift+Key" character defined in the *Simple key* button, while if the *Shift key* is not pressed, the "Key" character (defined in the *Simple key*) wil be sent.

Left - horizontal position of the top left corner of the button (in pixels).

Top - vertical position of the top left corner of the button (in pixels).

Width - width of the button (in pixels).

Height - height of the button (in pixels).

Font - font to use for the text. Push the button on the properties line to show the dialog for the choice of the font, size, style (normal, bold or italic) and effects (underlined and strikeout)
.
Foreground - define what to show in foreground **: None**, **Text label**, **Predefined image**, **Custom image** (details).

Text to show - text to show in case of "Foreground = Text label" .

Foreground image - image to show in case of "Foreground = Custom image" .

Initial status - specify which must be the initial status : "Pressed" or "Not pressed"

Inherits from common properties

Buttons font - if set to "Yes" then font will be inherited from common properties.

Style - if set to "Yes" then button style (PRESSED and NOT PRESSED) will be inherited from common properties.

Border size - if set to "Yes" then button border size (PRESSED and NOT PRESSED) will be inherited from common properties.

Text color - if set to "Yes" then button text color (PRESSED and NOT PRESSED) will be inherited from common properties.

Background properties - if set to "Yes" then background color, image and image position (PRESSED and NOT PRESSED button) will be inherited from common properties.

NOT pressed button properties

Style - border type (details) that will be associated to the "NOT PRESSED" status.

Border size - size (in pixel) of the border that will be associated to the "NOT PRESSED" status.

Text color - text color that will be associated to the "NOT PRESSED" status.

Background color - background color that will be associated to the "NOT PRESSED" status.

Image - image that will be associated to the "NOT PRESSED" status.

Image position - image position mode (details) that will be associated to the "NOT PRESSED" status.

Pressed button properties

Style - border type (details) that will be associated to the "PRESSED" status.

Border size - size (in pixel) of the border that will be associated to the "PRESSED" status.

Text color - text color that will be associated to the "PRESSED" status.

Background color - background color that will be associated to the "PRESSED" status.

Image - image that will be associated to the "PRESSED" status.

Image position - image position mode (details) that will be associated to the "PRESSED" status.