HYDRONIC UI Graphics Features

1. Contents

UI	User Interface
MCU	Microcontroller Unit

1 Programming Page Functional Features

In this section the functional features in the current version of UI related to Graphics, are described. The features include in this section refer to programming page.

1.1 CON Graphic Pages

Any ‘con’ routine shall include up to 10 graphic pages. The programmer can add or delete the pages, edit each on independently and save them (See 1 of Image 1). Graphics are either saved to the board or cache of the browser as described to the following section.

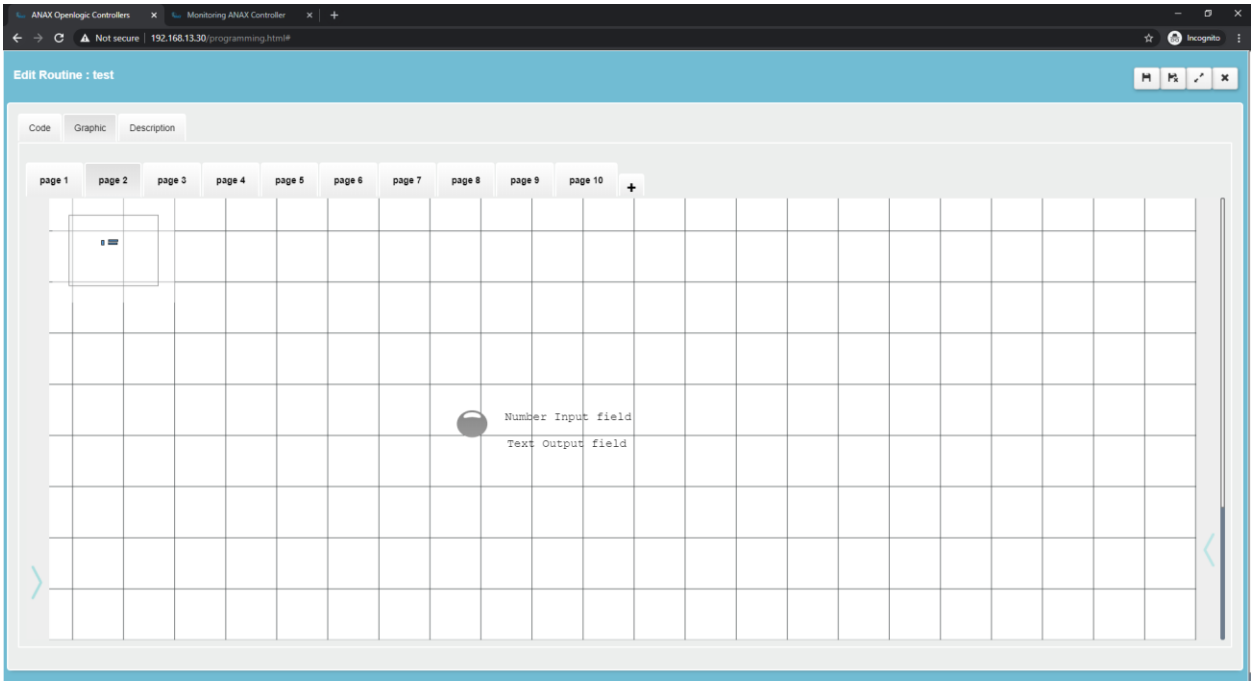


Image 1: Graphic pages of ‘con’ routine

1.2 Save Graphics

The functionality of ‘Save’ button has been implemented for both ‘scr’ and ‘con’ routines. The saving procedure of Graphics changes to the board is independent from the ‘Save to Board’ procedure which refers to the whole project. Upon clicking the ‘Save’ button, the Graphics of the specific routine are saved directly to the board (See 1 of Image 2).

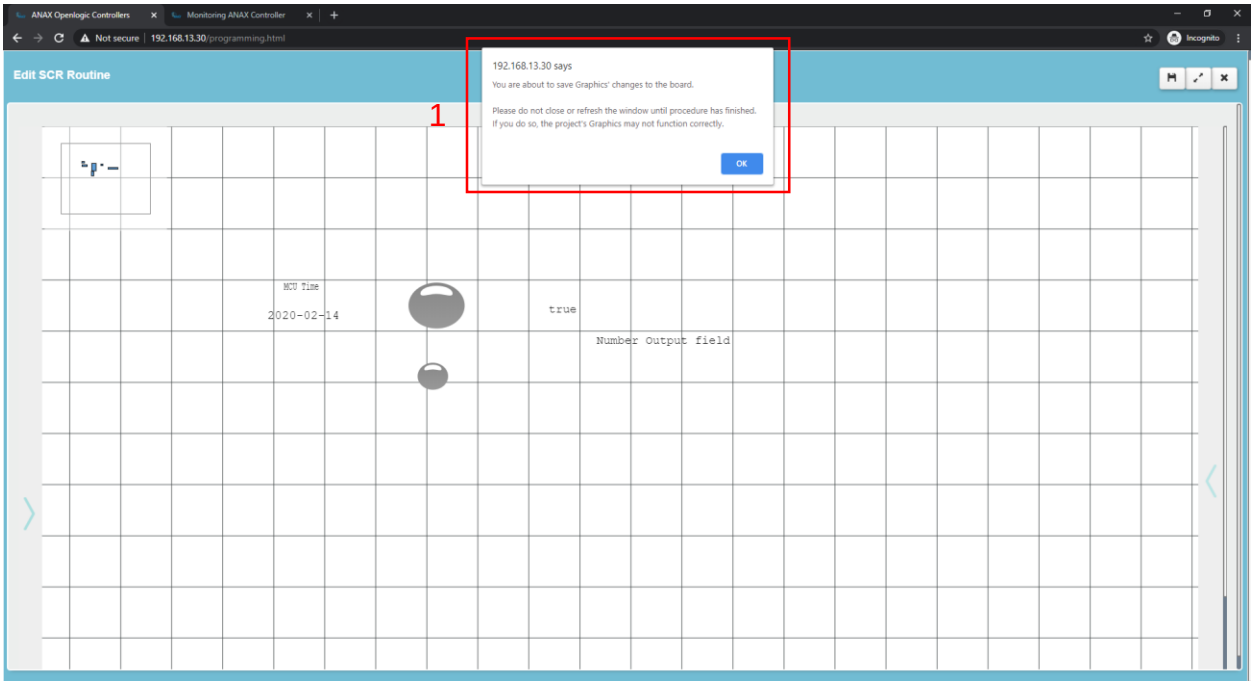


Image 2: Saving Graphics pf 'scr' routine

Apart from the 'Save' button, Graphics' changes are also saved to the cache during design. Then the programmer can save them to the board through the 'Save to Board' button. This functionality is implemented for the graphic pages for both scr and con routines.

1.3 Canvas

The handling of the Canvas has been reviewed in order to be properly resized and positioned. Any changes on the canvas shall affect accordingly the size and position of the included objects.

1.4 Commands Menu

1.4.1 Selection Type

The *Selection Type* command is implemented in a way that it supports three different selection methods on the canvas' objects.

By default, the *Single Object* selection method is selected meaning that the programmer can select only one object at a time on the canvas with a left click on it.

Clicking either the *Selection Type* command or the displayed selection method e.g. *Single Object* changes the selection method to the next available.

The *Sticky Selection* method allows the programmer to select multiple objects on the canvas by left clicking on them.

Finally, with the *Mouse drag* method the mouse can be dragged on the canvas in order multiple items to be selected inside a drawn rectangle.

1.4.2 Copy/Paste Functionality

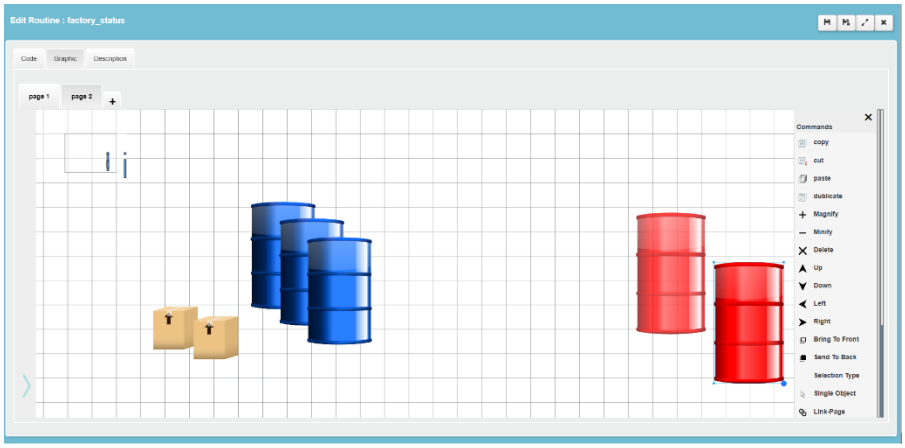
The Copy/Paste functionality is implemented so that the user can copy paste one or more objects in the same or different Canvas. In case a Group Object or a Show Hide Group Object is being copied, the connected objects are also copied.

1.4.3 Duplicate Functionality

The Duplicate functionality is implemented so that the user can duplicate the selected element or the selected elements. The user must follow the following procedure to successfully duplicate an element.

Step1: The user must select the desired element/s for duplication and click the duplicate button from the right-side menu.

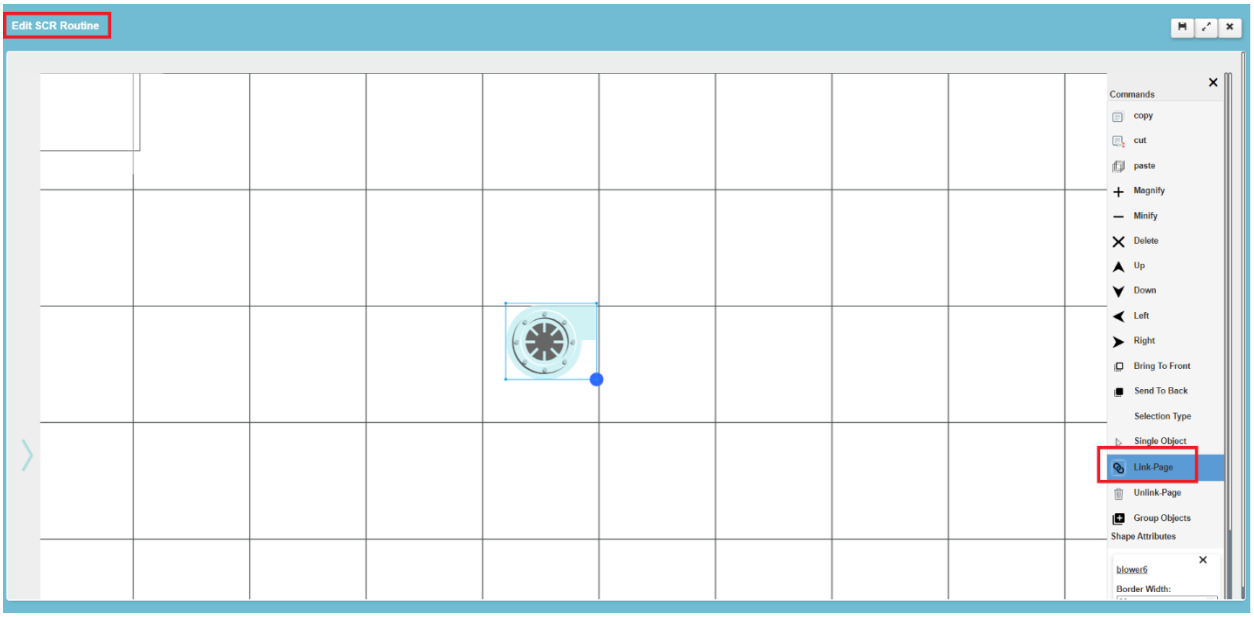
Step2: The element is being created on top of the initial element and the duplication is complete.



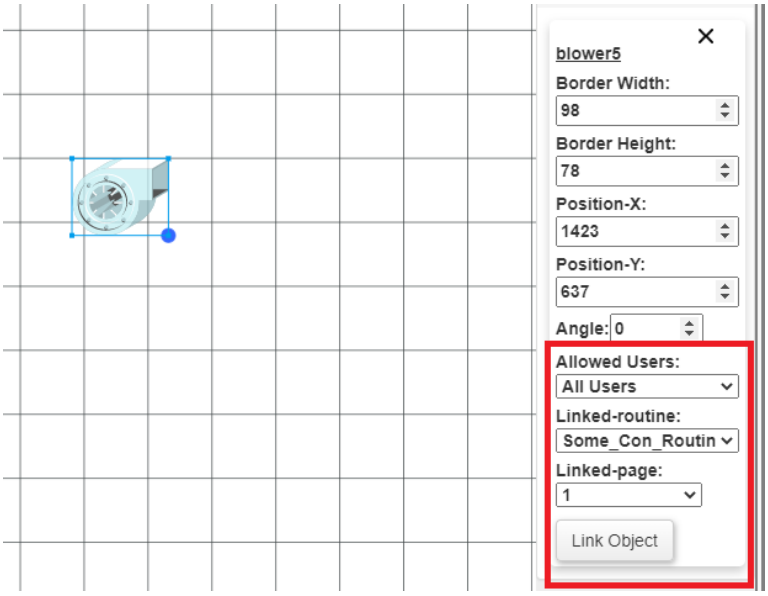
1.4.4 Link/Unlink Functionality

The Link/Unlink functionality allows in specific type of objects in scr routines to connect with graphic pages of con routines. So Linking is only possible from Scr pages to Con pages. The user must follow the following procedure to successfully create a link.

Step1: From an scr routine page and with the desired object to add a link selected, the user must press the Link-Page option from the right side menu.



Step 2: After click a new section in the object card will be added. The user must choose the routine to create a link to and then the page of this routine and press the “Link Object” button. If no error message appears, the Link was successful. (For the error messages see the Error notifications sections below).

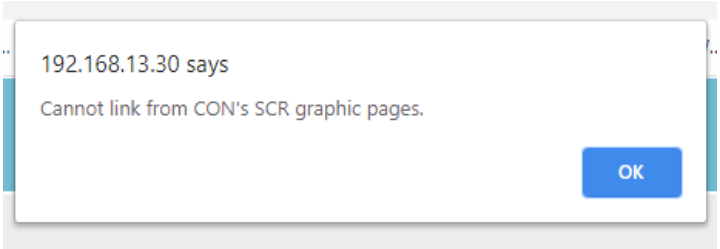


Step 3: The user can inspect the Link by double clicking on the object, both in programming & run page. The link-page window's title shows the name of the linked-routine and the number of the linked-page.

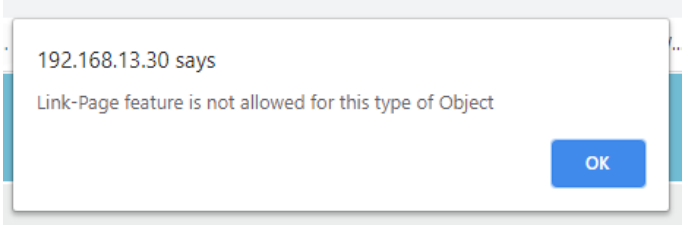
In order to Unlink the object, the user must select the object and press the "Unlink Page" option from the Right Side Menu.

Error notifications during connection:

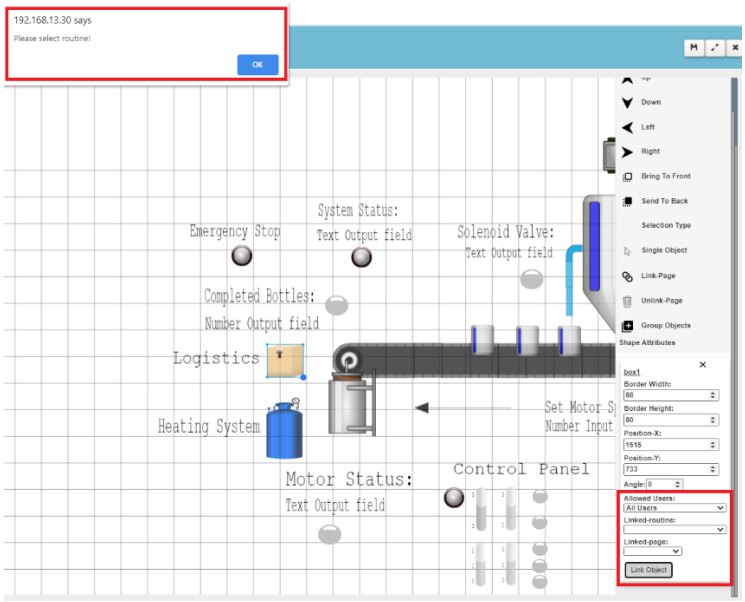
1) The Link-Page Feature is only allowed for graphic pages of scr routines. In case the user is in a con routine and presses the "Link-Page" button from the right side menu, the following notification will appear:



2) The Link-Page Feature is only allowed for specific types of graphics. In case the user tries link a page with a non-allowed type of graphic, the following notification will appear:



3) The Link-Page Feature will be completed if the user has selected a valid graphic-routine and a valid graphic-page. In case the user presses the “Link Object” button, without selecting first graphic-routine and graphic-page, relative notifications will appear. Bellow appears the notification when no valid routine was selected:



1.5 Objects Options

The ‘Allowed Users’ option is implemented so that an object can be visible to different user levels.

1.6 Interactive (Animation) objects

The properties of a number of the animation objects have been reviewed. The following properties have been optimized:

- ‘Width/Height’ is tested for correct functionality depending on the object. Checks have been made in order to restrict invalid input values.
- A lower limit of 2 seconds has been added as ‘Refresh Rate’ in order to ensure correct system’s functionality.
- ‘ColorON’ and ‘ColorOFF’ apply to each object on run page when the Read from Python option is turned to ON.

For all objects the programmer may define specific Viewer user privileges for each object in the canvas among 3 available levels: ‘All Users’, ‘Admin & High Level Users’ and ‘Admin User Only’. The default option is ‘All Users’ meaning that all users may see the object in the canvas.

Additionally, ONLY for the Interactive Input type objects he may define Editor user privileges. The default option is ‘Admin User Only’ meaning that only the Administrator User Level is able to interact with these objects.

The Viewer and Editor privileges are defined per object separately even for objects that belong to the same group. This means that even when making a grouping object invisible to some user levels the rest objects are not affected.

1.6.1 ShowHide Group

This item gives the ability to a specified items’ group to be hidden and appear again with double click. This functionality is only available for items that are port of the group this item belongs to. This item is displayed as a blue eye icon.

The programmer must select first all the desired items to be hidden and then from the **Commands menu** to click the option *Group Objects.* A new group of all the selected objects is now created. By double left click on the blue eye icon all the selected objects are hidden and the icon changes appropriately. Double clicking again the blue eye icon will appear all the items back to their positions.

1.6.2 LED DI

The ‘LED DI’ refers to digital values, hence, it shall be connected with a binary variable. The ‘LED DI’ object is completely functional.

The color of the LED DI remains in ColorOFF selection during the design in the programming page.

In case the option ‘Read from Python’ is selected as ON, the LED DI is set to color ON or OFF option in run page according to the value of the variable. The value of the variable is updated according to the selected refresh rate.

The LED DI remains in ColorOFF selection in run page when the option ‘Read from Python’ is turned to OFF. In this case the LED has no functionality in run page.

1.6.3 LED AN

The ‘LED AN’ refers to analog values, hence, it shall be connected with a variable which takes values in a pre-defined range. The ‘LED AN’ object is completely functional.

'Range (Min)' and 'Range (Max)' fields have been added to 'LED AN' object in order to calculate its min and max blinking frequency depending on the analog value readings.

The LED AN remains in ColorOFF selection during design in the programming page.

In case the option 'Read from Python' is turned to ON, the LED AN is blinking in run page according to the value of the analog variable, the calculated frequency and the colors selection. The value of the variable is updated according to the selected refresh rate.

1.6.4 Button Momentary

The momentary Button shall be connected with a binary variable.

During design mode in the programming page the button has no functionality at all.

When in run page, the button press indicates the ON state and the release the OFF state. The initial state of the button upon loading, is always set to OFF, thus it is affecting the connected variable immediately. Upon clicking, a high value (1) will be set to the connected variable. Upon releasing, a low value (0) will be set to the connected variable. The colors of the button should match the ones selected by the programmer for each state.

1.6.5 Button Maintained

The maintained Button shall be connected with a binary variable.

During design mode in the programming page the button has no functionality at all.

When in run page, the button can be clicked to toggle between ON/OFF state. The initial state of the button is decided based on the readings of the connected variable upon loading. Upon clicking for ON, a high value (1) will be set to the connected variable. Upon clicking for OFF, a low value (0) will be set to the connected variable. The colors of the button should match the ones selected by the programmer for each state.

1.6.6 Check box

The Check box shall be connected to a variable of Boolean type that takes the values True/False.

During design mode in the programming page the box has no functionality at all.

When in run page, the box can be checked/unchecked for True/False state. The initial state of the box is decided based on the readings of the connected variable upon loading. When checked a high value (True) will be set to the connected variable. Upon unchecking, a low value (False) will be set to the connected variable.

1.6.7 Radio button

The Radio box shall be connected to a variable of numeric type. The accepted values are either 0 for unchecked status or the number inserted in the field named "Value Checked" from the options menu in the programming page. In this way the run page will change the status of the Radio Box depending on the value of this field.

During design mode in the programming page the box has no functionality at all.

The initial state of the box in the Run page is decided based on the readings of the connected variable upon loading. When checked the value of the “Value Checked” field will be set to the connected variable. Upon unchecking, the zero value (0) will be set to the connected variable.

The programmer may use this button in order to create a multiple choice switch case and as an example.

1.6.8 Text or Number Output

The Text or Number Output objects shall be connected with a variable of string or number type respectively.

In case the option ‘Read from Python’ is turned to OFF, no connection is made with the MCU and the ‘Python Off Text’ that has been typed by the programmer in the options menu is displayed in the object’s position both for programming and run page.

In case the option ‘Read from Python’ is turned to ON, the value of the connected variable is displayed in the run page. The value of the variable is updated according to the selected refresh rate. Additionally, in the programming page, in the object’s position, the text ‘Text/Num Out: Routine.Variable’ is displayed, depending on the selected routine and variable to connect the Text or Number field with

1.6.9 Text or Number Input

The Text or Number Output object shall be connected with a variable of string or number type respectively.

In case the option ‘Send to Python’ is turned to OFF, no connection is made with the MCU and no input value can be set. The ‘Python Off Text’ that has been typed by the programmer in the options menu is displayed in the object’s position both for programming and run page.

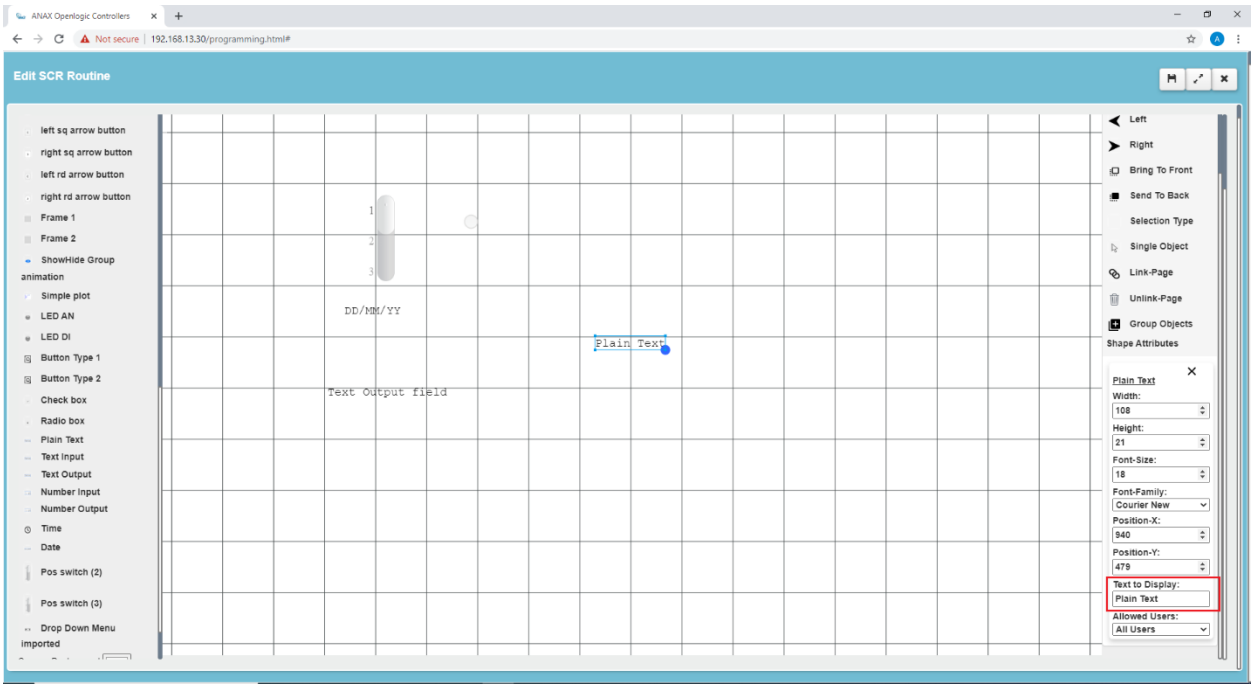
In case the option ‘Send to Python’ is turned to ON, the object is enabled in run page as a text or number input field. The initial value of the field, should match the existing value of the connected variable upon loading. By selecting the object, the user in the run page can type a new valid value and apply it to the variable connected to the specific object. The value of the variable is updated according to the selected refresh rate. Additionally, in the programming page, in the object’s position, the text ‘Text/Num In: Routine.Variable’ is displayed, depending on the selected routine and variable to connect the Text or Number field with.

1.6.10 Plain Text

Plain Text is the only animated object that cannot be connected to any routine/variables.

The programmer may use this object to display texts in his canvas by typing in the ‘Text to Display’ field (See Image below) the desired text. The best way to insert a new text is to

select the pre-defined text from the ‘Text to Display’ field (Ctrl-A) and then type the desired one, as when deleting it is again displayed the default value.



Plain Text object presentation

1.6.11 Time/Date Input

The Time or Date object shall be connected with a variable of string type that stores the time or date in the following formats respectively, hour: minutes: seconds or year/ month/ date. An example of such a variable usage may be seen below. (See Image 3) In this case the variable MCUTime gets content from the graphics and is compared with appropriate strings in the code. It could be also the opposite that these strings get content from the graphics and compared to the system time.

```
#   Out Variables: List your Out Variables here
#                  <name    ::  description>
#
# ================================================================================
# Please follow the given template in order to ensure your code is working.
# ================================================================================

class Graphics_Class:
    # This is your main class. You can add variables and methods.

    # ================================================
    #               Variables Section
    # ================================================
    # Please initialize your variables after this line, e.g. x = 0

    TimeOfActivation = '14:30:00'
    TimeOfDeactivation = '22:30:00'
    MCUTime = 0 # the MCUTime is connected to a 'Time' graphic object

    dout8 = 1

    def run(self):


        # ================================================
        #                   Code Section
        # ================================================
        # Your application must be written here. Add all your code.
        # You can add your own methods to be used inside the run() routine. Add as many as you wish.
        # Please insert your code after this line.

        # the MCUTime is loaded in the format of hours:minutes:seconds
        if self.MCUTime == self.TimeOfActivation:
            # activates channel 8 of DOUT8A when the time of MCU will be equal to 14:30:00
            self.dout8 = 1
        elif self.MCUTime == self.TimeOfDeactivation:
            # deactivates channel 8 of DOUT8A when the time of MCU will be equal to 22:30:00
            self.dout8 = 0

     # enddef

# endclass
```

Image 3: Activation/Deactivation of a channel depending on the MCU time.

In case the option ‘Read from System is turned to OFF, the ‘Default Value’ that has been typed by the programmer is displayed in the object’s position both for programming and run page. Additionally, the ‘Default Value’ is set to the connected variable upon the initial display of the object in run page.

In case the option ‘Read from System is turned to ON, the Time and Date field will display as values ‘MCU Time’ or ‘MCU Date’ respectively during design in the programming page to indicate that these fields will be updated from MCU in the run page. The actual time or date of the MCU will be displayed in run page and, will be updated according to the selected refresh rate. These values will then be set to the connected variable.

1.6.12 Position Switch (2) & (3)

The Position Switch object shall be connected to a variable, which has at least 2 different values in case of Position Switch (2) or 3 different values in case of Position Switch (3). The User can set these values (levels) during design. The switch changes state by double-clicking on the objects’ image or when the value of the connected variable is

automatically updated from the MCU. Depending on its position the switch assigns the selected value to the connected variable.

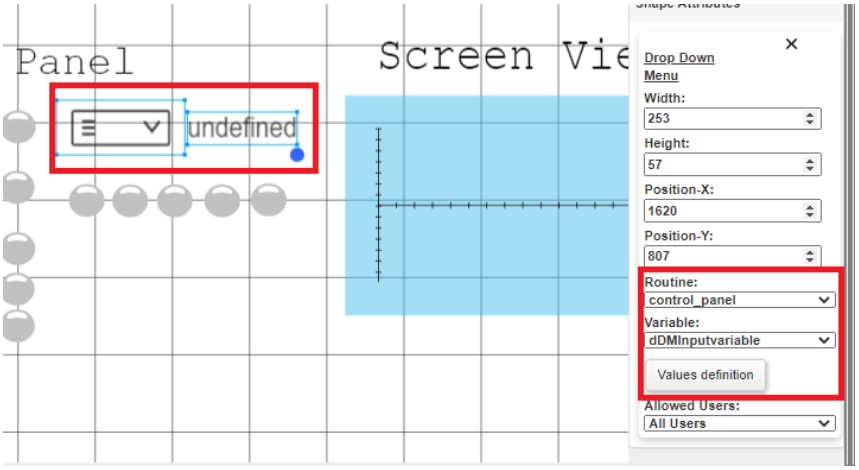
Both switches have also an UNDEFINED state which appears only when the connected variable has other value than the defined levels.

1.6.13 DropDown Menu

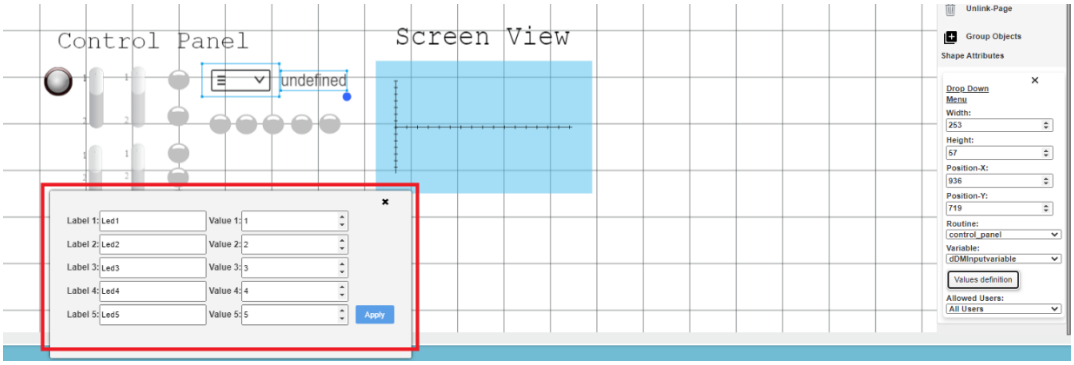
The DropDown Object shall be connected to a variable in which up to five 5 different values may be assigned. The programmer shall configure for each dropdown selection the displayed label and the value to be assigned to the connected variable.

In order for the user to assign labels to values must follow the following procedure:

Step1: From the programming page, the user must select the drop-down menu and after the selection of the Routine and the Variable, must press the “Values definition” button.



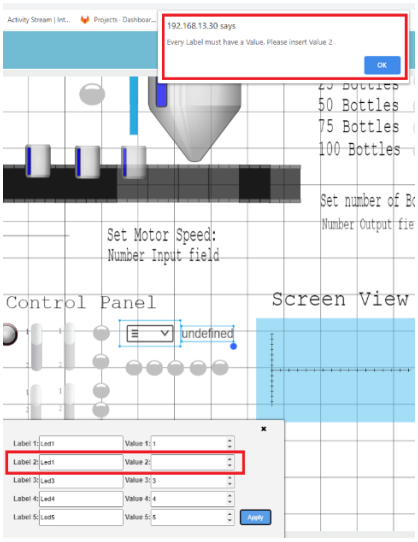
Step2: The user must insert the desired labels and values to the appropriate window and the press the “Apply” button.



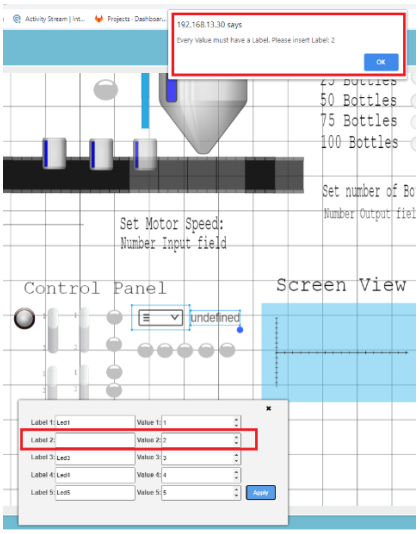
If the user clicks outside of the window or clicks on the close button the window will close **without saving the values or the labels**. After clicking on the Apply button, a number of checks are implemented. If everything is ok, the window will close without any error message and the procedure is complete.

In any other case the possible errors during the insertion of the values and labels are as follows.

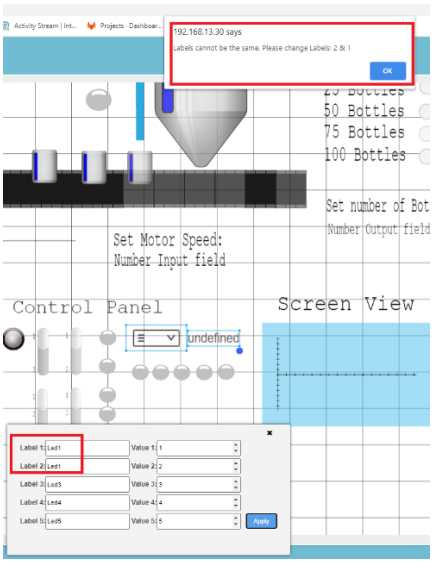
1) If there is a label without value, the error message *Every Label must have a Value. Please insert Value #* will appear:



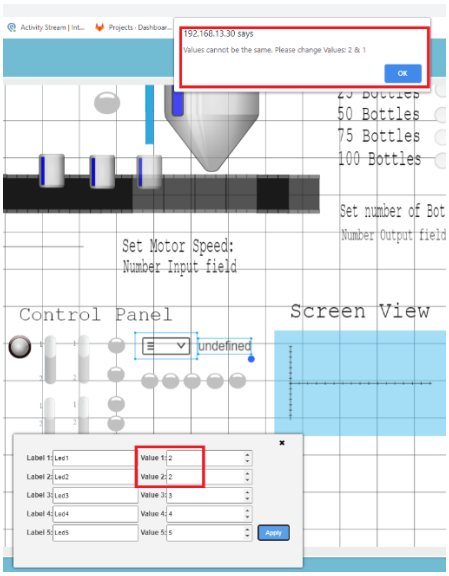
2) If there is a value without label, the error message *Every Value must have a Label. Please insert Label: #* will appear:



3) If at least 2 labels are identical, the error message *Labels cannot be the same. Please change Labels: X & Y* will appear:

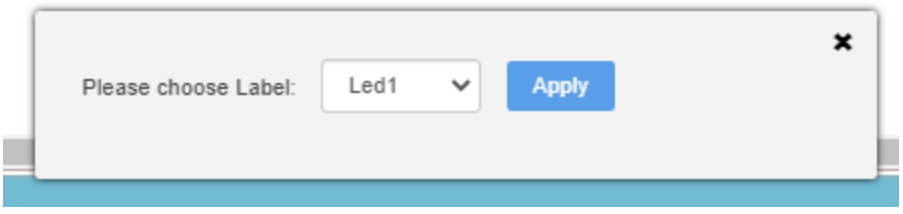


4) If at least 2 values are identical, the error message *Values cannot be the same. Please change Values: X & Y* will appear:



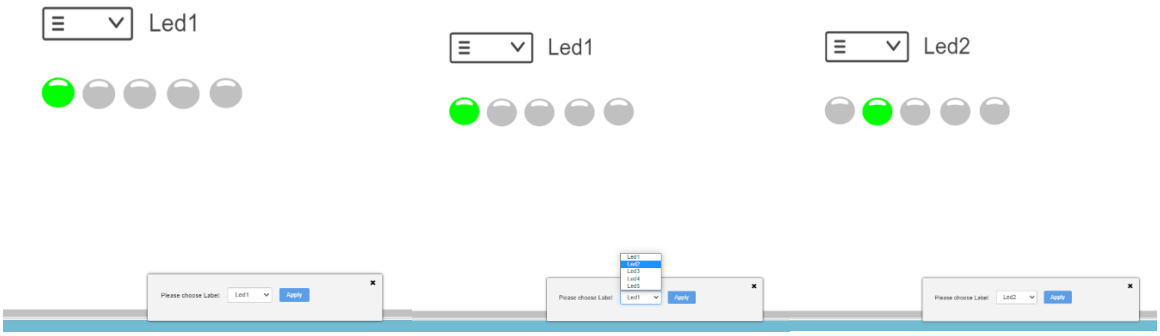
In the run page, the user can choose one of the labels by following the following procedure:

Step 1: The user must select the drop-down menu and the following window will appear.



Step2: The user can choose one of the labels and press the “Apply” button. If the user clicks outside of the window or clicks on the close button the new label **will not be set**.

The following 3 images show the procedure:



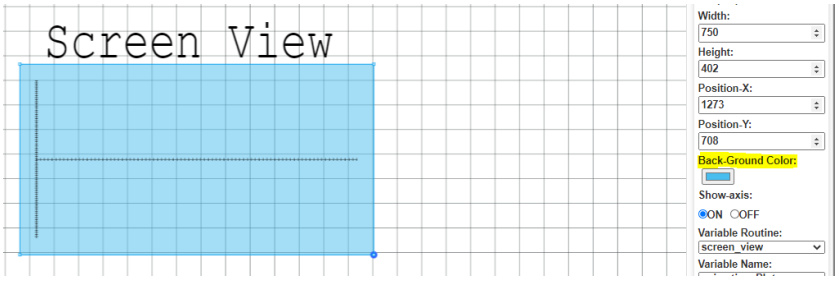
The displayed label of the Drop-Down Menu has also the ability to change automatically in case the connected variable changes (i.e. from the python code). In case the connected variable has a value other than the 5 predefined values, the displayed label shows the word “Undefined” (state Undefined).

Upon initialization the displayed label of the Drop-Down Menu starts with state “Undefined” till it is initialized with the connected variable value.

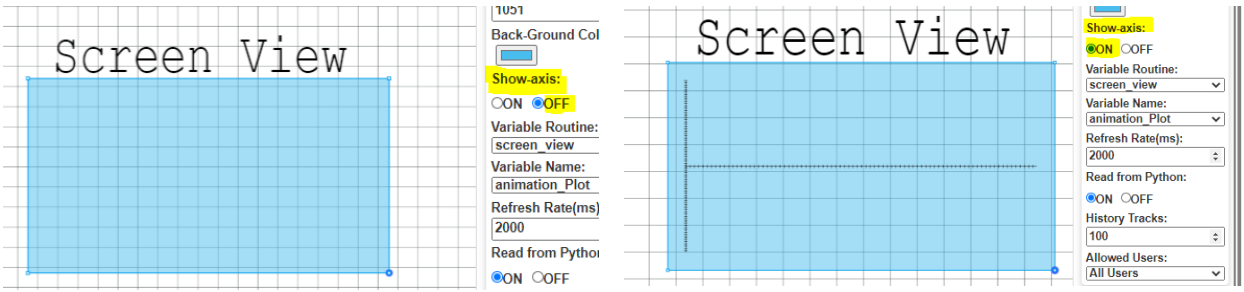
1.6.14 Simple Plot

The Simple Plot Object shall be connected to a variable of arithmetic type in order to graphically display its value in time.

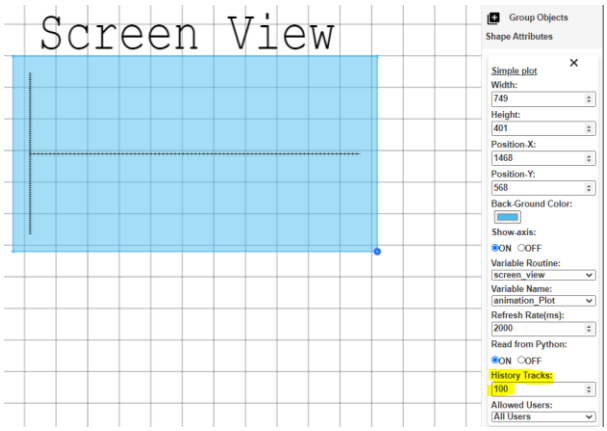
The user may change the background color of the plot from the field “Background Color”. The default color is white. Bellow appears an image with selected blue background:



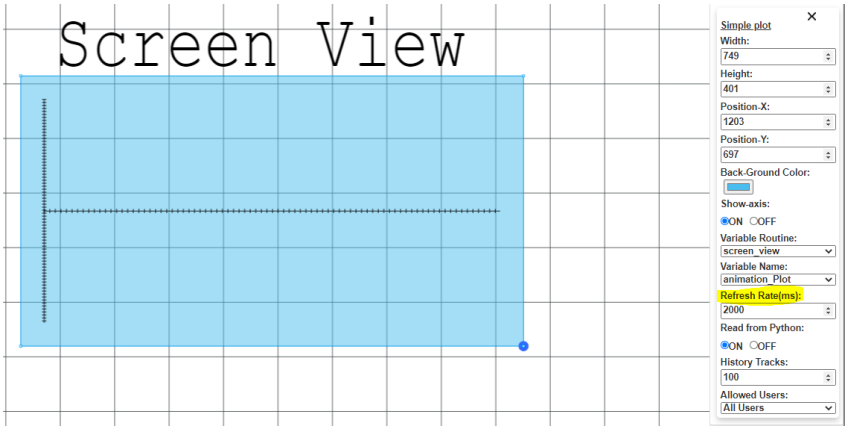
The user may choose whether axis will be visible in the “Show Axis” field. Bellow appears a plot with visible and invisible axis:



The user has the option to choose the number of the samples by inserting a number in the History Tracks field as shown below:



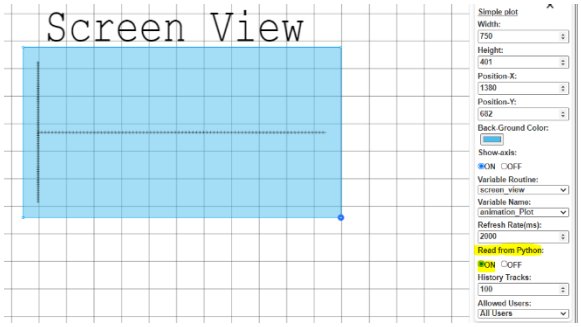
The user may also specify the refresh rate from the according field.

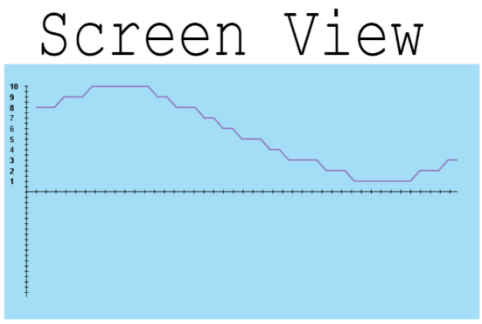


The user may choose whether the plot will be connected to a python-variable.

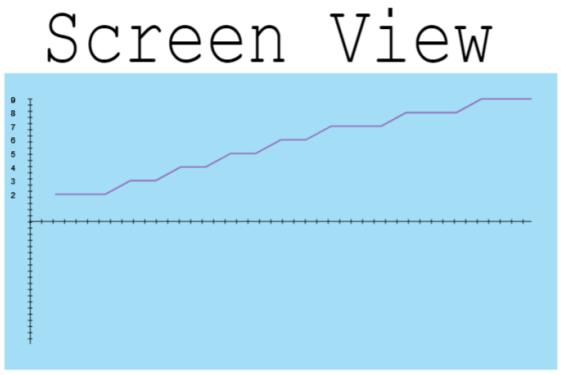
In case the option ‘Read from Python’ is turned to OFF, no connection is made with the MCU and the Plot object remains unchanged.

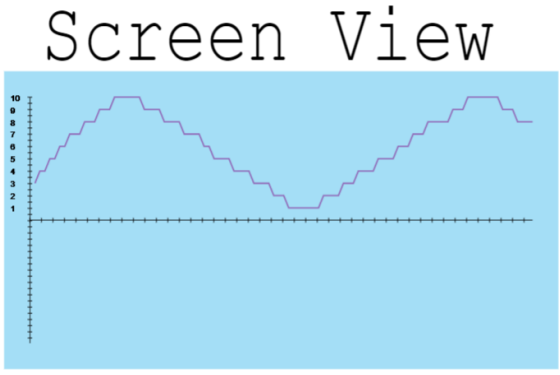
In case the option ‘Read from Python’ is turned to ON, the value of the connected variable is displayed in the run page. The values of the variable are depicted to the left side of the Plot. Below it is shown a plot with connected variable in programming and in run page respectively:





The curve of the Plot diagram adjust dynamically in every new value that is being inserted and rescales the X-axis and the Y-axis automatically. This way the form of the curve depends of the number of the samples that is being stored. Bellow there is an example of the same curve with few and many values respectively:





.

1.6.15 Power loss resistant values for Run page

For every input type animation object, there is a mechanism for saving any user changes made in the graphic pages in run page. This feature stores the value of the connected variable in order to restore it in a possible power-cycle of the uController.

The programmer is responsible to mark the objects to be power loss resistant during design. He has the ability to activate and deactivate this mechanismfor every input type object as seen in the Image below.

1.7 Embedded objects

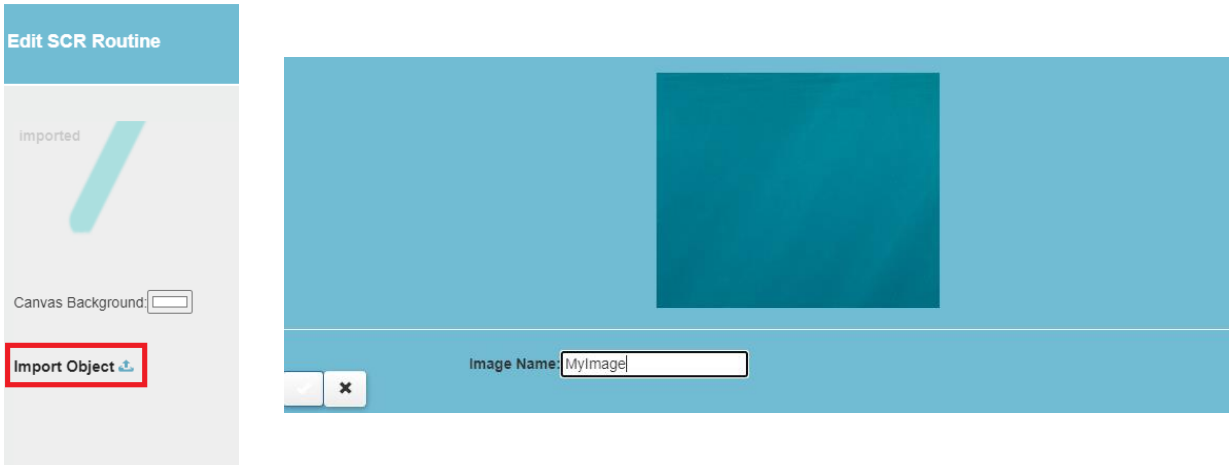
The majority of the actions related to the embedded objects function as expected. These actions are the Copy, Cut, Paste, Magnify, Minify, Delete, Up, Down, Left, Right, Multi-Select, change of Width/Height, Position-X, Position-Y and Angle. Upon each of these actions, the related filed is updated along with the display of the object inside the canvas.

1.8 Imported objects

1.8.1 User Image Import

The “import images” mechanism allows the programmer to import images and use them as graphic objects on the canvas.

Clicking the “Import Object” button as shown in the image below, the user is asked to



browse and choose an image to Upload. A modal box will open and ask the user to define a unique name for the imported image. After clicking the check button, the image will appear in the left side menu in the section Imported/UserImported.

He is now able to click on the imported image and it will appear on the canvas. All the main properties for the imported objects are available for this object too.

A delete icon is displayed next to the imported objects’ name so that the programmer may delete it if unnecessary.

Upon import/delete of an image refresh the page if the changes do not take effect.

2 Run Page Functional Features

In this section the functional features in the current version of UI related to Graphics, are described. The features include in this section refer to programming page.

2.1 Graphics Loading

The menu section related to graphics has been added under the ‘Directories’ option. This section includes the folders and routines that have graphics defined. All ‘scr’ routines will be added to the list. A ‘con’ routine will be added to the list only if it contains at least one page with graphics defined in it. Finally, a folder will be included if it contains either an ‘scr’ or a ‘con’ routine with active graphic pages. Any folder or routine that does not fulfill the above criteria will be excluded from the list of ‘Directories’

2.2 Graphics Update

Given that an active mode of run page exists focused on the graphics section, any changes performed by the programmer shall update the section. Upon detecting the graphics’ changes on board, the user is informed in order to refresh run page and load the new settings (See 1 of Image 3).

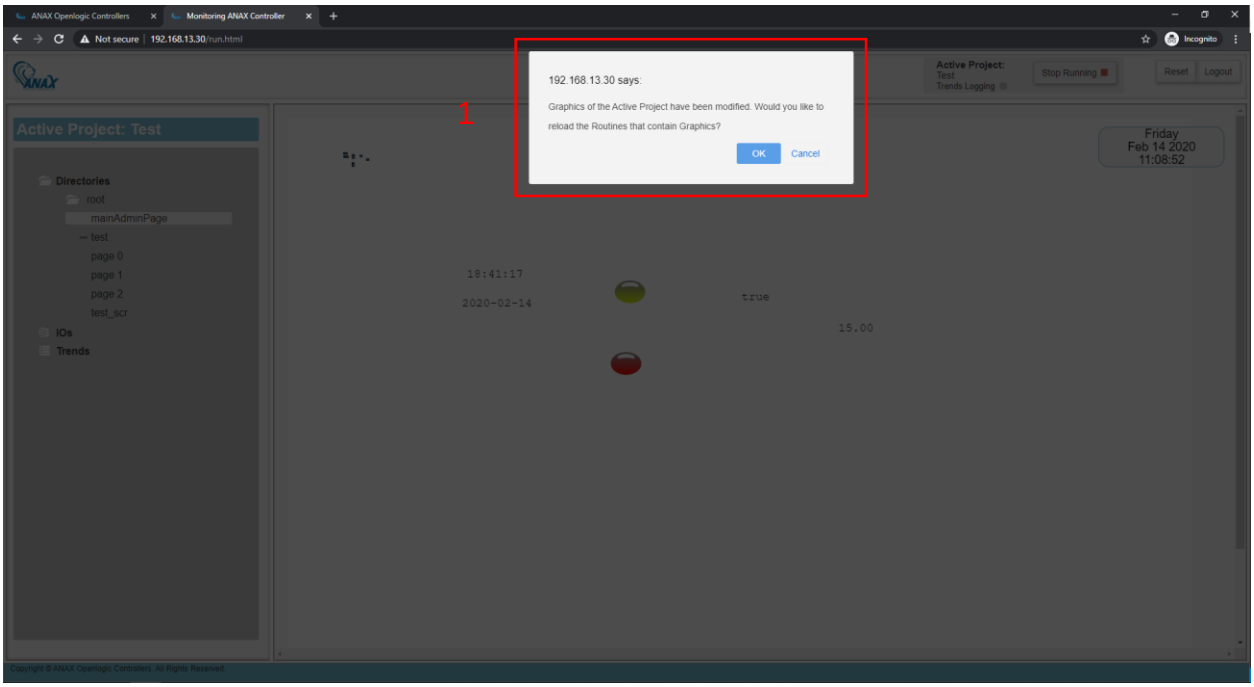


Image 4: Run page upon change of Graphics

2.3 Canvas

The handling of the Canvas has been reviewed in order for it to be properly resized and positioned. Any changes on the canvas shall affect accordingly the size and position of the included objects. In the Run page only, the majority of the objects cannot be selected, edited or moved, except the ones that allow editing.