iX Developer

User's Guide

English

MAEN832F, 2013-12

iX Developer User's Guide

Foreword

The iX Developer software is used to configure iX Panels and PC operated control applications, including applications for IPCs (Industrial PCs).

iX Developer makes it easy to create logical, flexible and effective HMI applications that provide the right information on the right occasion to operators and to other systems.

This manual is based on an example project that describes a step-by-step design of a project for iX Developer.

For more detailed information, please refer to the iX Developer help file.

Order no: MAEN832F

Contents

1 Introduction

The iX Developer software is used to configure iX Panels and PC operated control applications, including applications for IPCs (Industrial PCs) from Beijer Electronics.

iX Developer contains all basic functions needed in an application. The functions are tested and developed with customer needs and preferences in focus.

Pre-defined objects in iX Developer can be used to create complete process images, providing an overview of a complex application. You can customize the pre-defined objects or create objects of your own.

Communication drivers for a large number of controllers and automation equipment are available.

1.1 Controller

Operator panels can be connected to many types of automation equipment, such as PLCs, servos, and drives. Further on, the expression *controller* is used as a general term for the connected equipment.

1.1.1 Tags

Data values in a controller are referred to as *tags*. Tags may also belong to the system or be internal. A tag has a symbolic name and can be of different data types.

Objects connected to tags can change values in the controller, and tag values can be reflected by changing object appearance in various ways. Objects in a screen will remain static until connected to a tag.

1.2 Manual Structure

The USERS GUIDE is based on an example project that makes it easier to start using iX Developer. If the instructions in the example are followed carefully, this should result in a functional project that can be further developed, or used for inspiration. The target of the example is a PC, but all functions works similarly for all supported operator panels.

Detailed information about iX Developer is available in the help file, displayed by pressing **F1** while using the software.

The instructions in the USERS GUIDE are more detailed in the beginning. As the example progresses and you become familiar with iX Developer, instructions for tasks that are of a repetitive nature may be brief or omitted.

2 Installation and Start

iX Developer is installed on a DEV PC, where projects are developed, designed and compiled. The project is then run in an OPERATOR PANEL, PC or PC TARGET to observe and control a controller (or a group of controllers).

2.1 System Requirements

2.1.1 iX Developer

Parameter	Recommendation	
RAM	2 GB	
Processor	2 GHz or higher	
Operating system	Microsoft Windows 7	
	Microsoft Windows Vista	
	Microsoft Windows XP SP3	
Graphics card	Tier 2:	
	DirectX version	Must be greater than or equal to 9.0.
	Video RAM	
	Pixel shader	Version level must be greater than or equal to 2.0.
	Vertex shader	Version level must be greater than or equal to 2.0.
	Multitexture units	Number of units must be greater than or equal to 4.

Updates

Software, drivers and protocols may have been updated since the USB stick was produced. Therefore, it is recommended that you use the built-in update function in iX Developer before creating a project.

Therefore, it is recommended that the built-in update function in iX Developer is used before creating a project

2.1.2 iX Runtime

Parameter	Recommendation
RAM	1 GB
Processor	1.3 GHz or higher

Parameter	Recommendation	
Operating system	Microsoft Windows 7	
	Microsoft Windows Vista	
	Microsoft Windows XP SP3	
Graphics card	Tier 2:	
	DirectX version	Must be greater than or equal to 9.0.
	Video RAM	Must be greater than or equal to 120 MB.
	Pixel shader	Version level must be greater than or equal to 2.0.
	Vertex shader	Version level must be greater than or equal to 2.0.
	Multitexture units	Number of units must be greater than or equal to 4.

2.1.3 Special Requirements for Some Objects

For some objects to be included in the iX Developer project, specific software versions are required. Simulation of the project on the DEV PC may also be limited for some targets.

Object	Minimum requirement	Simulation on PC target	Simulation on panel target
Media Player	Windows Media Player 10	Supported	Not supported
PDF Viewer	Acrobat Reader 9	Supported	Not supported
Web Browser	Internet Explorer 7	Supported	Not supported

2.1.4 Starting iX Developer

The installation creates an icon for the configuration tool in the group named iX Developer in Windows Start menu.

Click on **Start/All Programs/ iX Developer/ iX Developer** to start the configuration tool.

2.1.5 Help

Help topics are provided when pressing **F1** while iX Developer is running.

3 New Project

Objective:

- Creating a new project.
- Getting familiar with the tool windows and the layout of the desktop area.

3.1 Creating a New Project

1. Start iX Developer and select **Create New Project**.
   This starts a wizard that will guide you through creating the project.
2. Choose a **PC** with a **1024 × 768** resolution as target for the application. Click **Next**.
3. Select **DEMO** in the list of controllers. Click **Next**.
   The demo controller, including regular tags (data containers) and counters, is used to design and test a project directly on the DEV PC without connection to an external controller.
4. Give the project a name. For this tutorial use **DEMO_TEST**. Check that the suggested location is appropriate. If not, click **Browse** to select another location.
5. Click **Finish**.

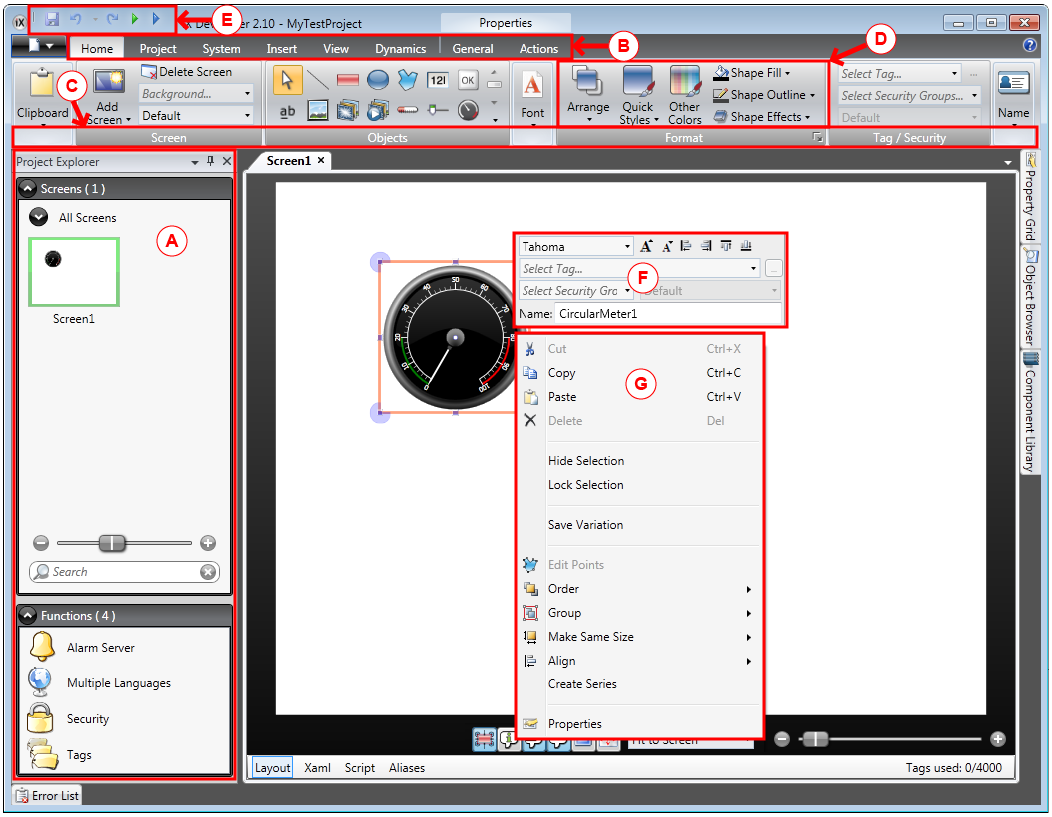
Project files can be stored anywhere in the computer environment where you have write access rights.

The project opens automatically.

3.2 Desktop Area

The desktop area displays screens and configuration pages for project components such as controllers and functions. The desktop area shows only one screen or component at a time. When multiple screens or components are opened a row of tabs are shown in the upper part of the desktop area. Clicking on a tab activates its contents for editing.

If there are more tabs open than can be displayed, navigation arrows in the upper part of the desktop area can be used to scroll between them.



Indication in picture	Desktop area component	Described in section
A	Project Explorer	*Project Explorer*
B	Ribbon tabs	*Ribbon Groups and Controls*
C	Control groups	
D	Controls	
E	Quick Access Toolbar	*Quick Access Toolbar*
F	MiniToolBar	*MiniToolBar and ContextMenu*
G	ContextMenu	

3.2.1 Project Explorer

When a new project is opened, an empty screen is active in the desktop area. The **Project Explorer** is docked to the left.

3.2.2 Ribbon Groups and Controls

Ribbon tabs are located in the top section of the tool window. Each ribbon tab holds one or several control groups. Each group contains a set of related controls. You use the controls to design screens, and to make settings for objects and controls in the project.

If you are not used to software with ribbon tabs, please spend a minute to get familiar with the ribbon concept.

3.2.3 Quick Access Toolbar

The **Quick Access** toolbar is always visible at the top of the desktop area. It contains the commands **Save**, **Undo**, **Redo**, **Run** and **Simulate** as iX Developer is started.

3.2.4 MiniToolBar and ContextMenu

When right-clicking objects in iX Developer, a **MiniToolBar** and a **ContextMenu** are displayed. The MiniToolBar contains commands that are specific for iX Developer, to for example connect objects to controller tags. The ContextMenu contains regular Microsoft application commands such as **Copy**, **Paste** etc.

4 Controller Tags

Objective:

- Defining a tag list for the project.
- Saving the project.

4.1 Adding Tags

Objects connected to tags can change values in the controller, and tag values can be reflected by changing object appearance in various ways. Objects in a screen will remain static until connected to a tag.

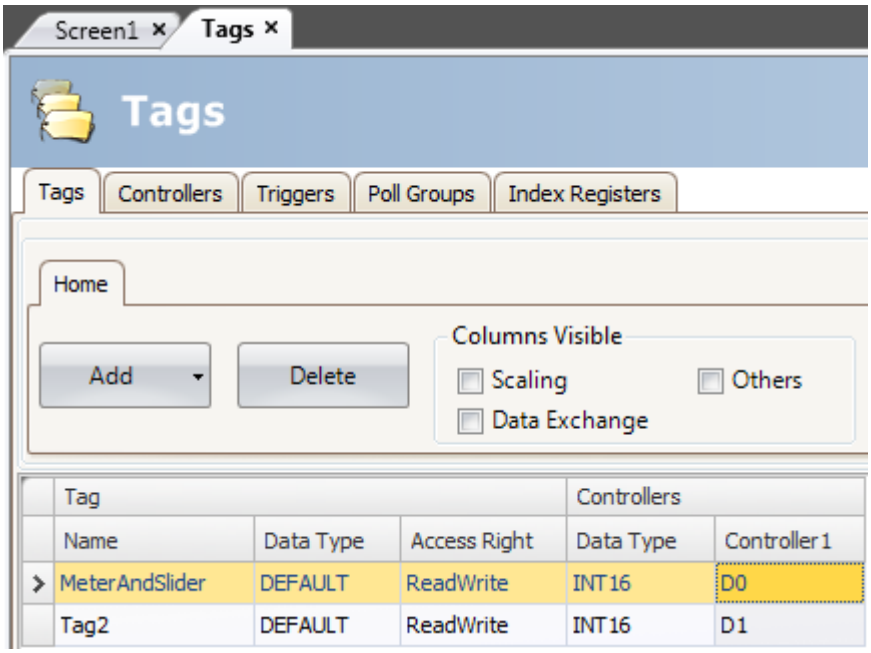
1. Click on **Tags** in the **Project Explorer**.
   The tags configuration page opens on the desktop. There is one elements in the tags list by default.
2. Click on the first field (**Name**) in the first row.
   A highlighted row is added and the text **Tag1** appears.
3. Press [**TAB**] on the keyboard.
   The selection moves to next field (**Data Type** for Tags). You do not need to change the data type.

Note:
The data type for the tag can be used as a presentation format in some situations; for example to show the correct engineering unit when using scaling. DEFAULT follows selection made for the Controller data type.

---

4. Press [**TAB**] on the keyboard again.
   The selection moves to next field (**Access Right**). You do not need to change the access rights now.
5. Press [**TAB**] on the keyboard again.
   The selection moves to next field (Controller **Data Type**). You do not need to change the data type now.
6. Press [**TAB**] on the keyboard again.
   The selection moves to next field (**Controller 1**).
7. Type "D0" in the **Controller 1** field.
   The entries in the **Controllers** column correspond to tags in the selected controller. There are predefined tags in the DEMO controller that can be accessed by their respective tag address, e.g. **D0** denotes an integer tag field.
8. Press [**TAB**] until the two first rows are completely filled in. Type "D1" for the second controller tag.
   Some fields are automatically filled and when needed, incremented. The data type is automatically changed depending on what you type in the **Controllers** column.
   The **Name** of a tag is an identifier for the tag and can be any alphanumeric string, beginning with a letter (a-z, A-Z).

9. Rename **Tag1** to "MeterAndSlider".



The **D0** tag will be used in the next section to control and observe a controller tag value in a screen.

4.2 Saving the Project

1. Click on the **Save** symbol in the **Quick Access Toolbar**.
   The project will be saved in the location that you selected when creating the project.

5 Editing Objects

Objective:

- Inserting a slider and a meter.
- Learning how to format and align objects.
- Testing the project; controlling and observing a controller tag value with objects in a screen.

5.1 Adding Objects

5.1.1 Meter

1. Click on the **Screen1** tab in the desktop area and make sure that the **Home** tab in the ribbon area is selected. Select a circular meter from the **Objects** group. Place it somewhere in the upper left section of the screen.
2. Drag a corner handle to resize the meter to a suitable size where the meter needle and the scale are clearly visible.
3. Right-click on the meter. Click on **Select Tag...** in the MiniToolBar. Select the **MeterAndSlider** tag by clicking on it in the drop-down menu, and then click **OK**.

5.1.2 Slider

1. Select a slider from the **Objects** group. You may need to expand the **Objects** group by clicking the small arrow at the lower right in order to select the slider. Place it just below the circular meter on the screen.
2. Right-click on the slider. Click on **Select Tag...** in the MiniToolBar. Select the **MeterAndSlider** tag by clicking on it in the drop-down menu, and then click **OK**.

5.1.3 Align

An object that is dragged snaps into position relative to other objects.

1. Slowly drag the slider up and down.

   Notice that the slider snaps into position at a short distance below the meter.
2. Slowly drag the slider to the left and to the right.

   Notice that the slider snaps into position and that snap lines appear when the slider is aligned with the meter.
3. Arrange the slider in a position closely below the meter and with its left edge aligned with the left edge of the meter.

5.1.4 Resize

1. Make a multiple selection of the two objects (point at an empty area in the screen and drag diagonally across the objects).

   A multiple selection (group) has one primary object. The primary object has an orange frame; the other objects have blue frames. When enforcing format commands on the group the primary object is used as a template.

   If the meter is not the primary object then:
2. Click on it to change the primary selection of the group to the meter.

   Now adjust the width of the objects in the group:
3. Click on the **Arrange** control, located in the **Format** group of the **Home** tab, and select **Make Same Width**.

5.1.5 Changing Appearance

1. Select the slider on Screen1.
2. Click on the **Quick Styles** control in the **Format** group and select a new color style.
3. Click the small arrow at the lower right of the **Format** group in order to make additional settings for outline, shadow/fill effects etc.
4. Select the meter on Screen1.
5. Select the **General** ribbon tab and locate the **Style** group. Try the different pre-defined styles and evaluate which style suits your preferences the best.

5.2 Running Project Test

The project can be compiled and run at almost any time. This allows you to test your design continuously in an iterative manner.

1. Click on the **Run** icon in the **Quick Access Toolbar**.

   The project is now validated, and when no errors are found, the project will be compiled and executes in the development environment.
2. Drag the slider handle back and forth.

   Since both objects are connected to the same tag, the meter needle follows as you change the value of the slider control.
3. Close the **Run** window.

6 Navigation and Screen Jumps

An iX Developer project consists of screens with objects, usually connected to controller tags. All screens have the same basic functions. A screen can be given specific properties to specialize its behavior in the project:

- **Startup Screen**: The Startup Screen is the first screen that is displayed in runtime. By default, Screen1 is used as Startup Screen, but any screen can be designated Startup Screen by right-clicking on the screen and selecting **Set as Startup Screen**.
- **Background Screen**: Any screen, except screens with Aliases, can be used as a Background Screen by the other screens in the project. For more information about Aliases, please refer to the iX Developer help file.
- **Screen Template**: A screen that is saved as a Screen Template can be used not only in the current iX Developer project, but also in all future projects.

Screen jumps are made with actions that can be assigned to e.g. buttons. When using the **Navigation Manager** to add screens and create links between screens, buttons are added automatically in the upper left corner of the screen that the link originates from.

Objective:

- Adding new screens and setting up screen jumps with buttons.

6.1 Screen Navigation

6.1.1 Navigation Manager

- Click on the **View** tab in the ribbon area. Click on **Navigation Manager**.

  The **Navigation Manager** opens in the desktop.

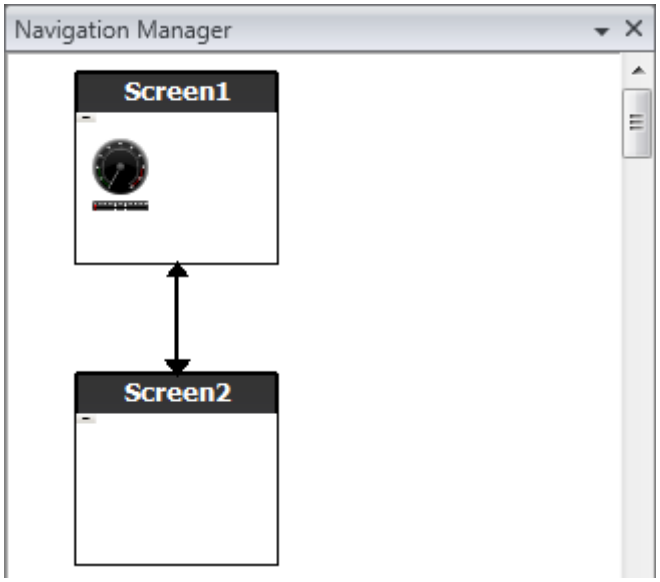
6.1.2 Adding a Screen

- Point at Screen1. Click and drag a connection from Screen1 to anywhere in the **Navigation Manager** area.

  A new screen appears (Screen2). A button labeled Screen2 appears in the upper left corner of Screen1.

6.1.3 Screen Jump

- Click and drag a connection from Screen2 to Screen1 in the **Navigation Manager**.
  A button labeled Screen1 appears in the upper left corner of Screen2.



6.2 Background Screen

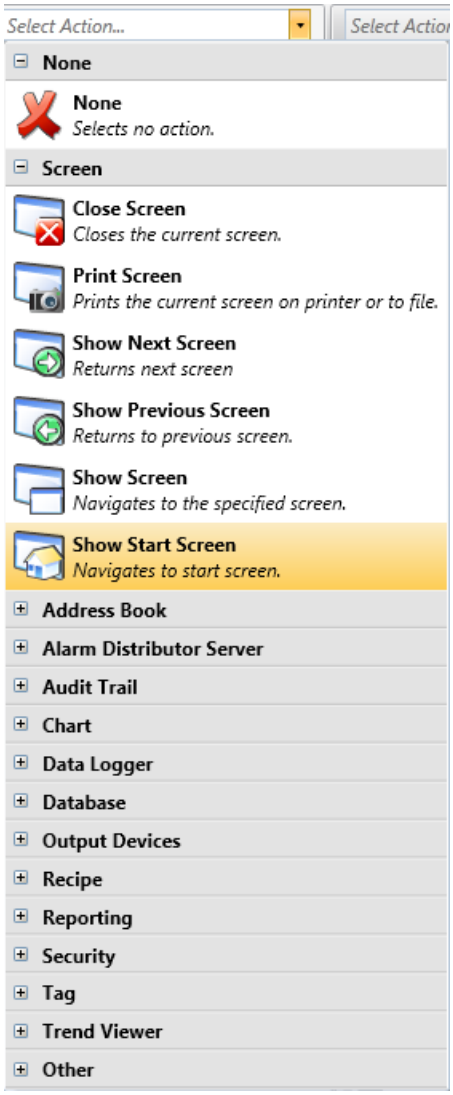
6.2.1 Adding a Screen

- Click on **Screen** on the **Insert** ribbon tab.
  A new screen (Screen3) is created in the project, and opens for editing.

6.2.2 Background Screen

1. Make sure that Screen3 is open for editing on the desktop.
2. Select a **Button** from the **Objects** group (located on the **Home** tab), and place it in the lower left area of Screen3. Label the button "Start Screen".

3. Keep the button selected and click on the **Actions** tab. Select **Show Start Screen**, located in the **Screen** group, from the drop-down list for the **Click** action.



4. Open Screen2 for editing by clicking on it in the **Project Explorer**.
5. Select the **Home** tab. Select Screen3 in the **Background Screen** drop-down list in the **Screen** group.
6. Try to change location of the Start Screen button in Screen2. This is not possible. Notice that changes made to Screen3 are reflected in Screen2.

There are now two navigation facilities from Screen2 to Screen1 (the Start Screen).

6.3 Running Screen Navigation Test

1. Run the project.
2. Verify that each of the buttons in Screen2 performs a jump to Screen1.
   Since no screen has been set to be Startup Screen, Screen1 remains Startup Screen for this project.
3. Close the **Run** window.

7 Trend Viewer

The trend viewer function stores register information from the controller in the operator panel. Real-time trend viewer as well as historical trend viewer is available.

Objective:

- Adding a trend viewer object with two curves.

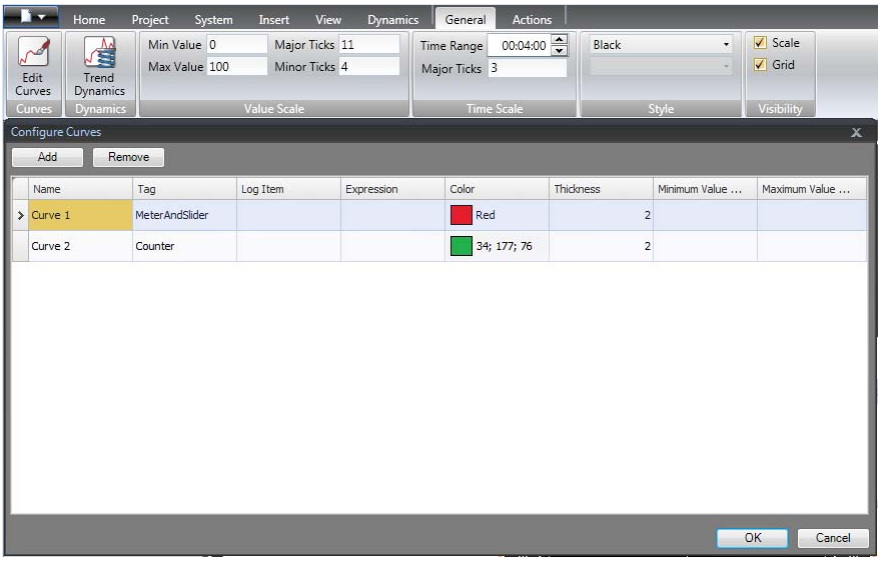
7.1 Adding a Real-Time Trend Viewer

1. Open Screen1 for editing in the desktop.
2. Select the **Trend Viewer** object from the **Objects** group and place it on the screen.

7.1.1 Curves

1. Click on **Tags** in the Project Explorer and add a tag. Type "Counter" in the **Name** field and connect it to **C0** in Controller1.

   C0 is a counter that counts from 0 to 100 and back to 0 with a frequency of 1 Hz.
2. Open Screen1, make sure that the trend viewer object is selected, and click **Edit Curves** on the **General** tab.
3. Add a curve and connect it to the same tag that you used for the slider in Screen1.
4. Add a second curve and connect it to the Counter tag you just added, and select another color for this curve.



5. Click **OK**.

7.2 Running Real-Time Trend Viewer Test

- Run the project and check that both curves are visible in the trend viewer. Test that **Curve 1** changes with the slider.

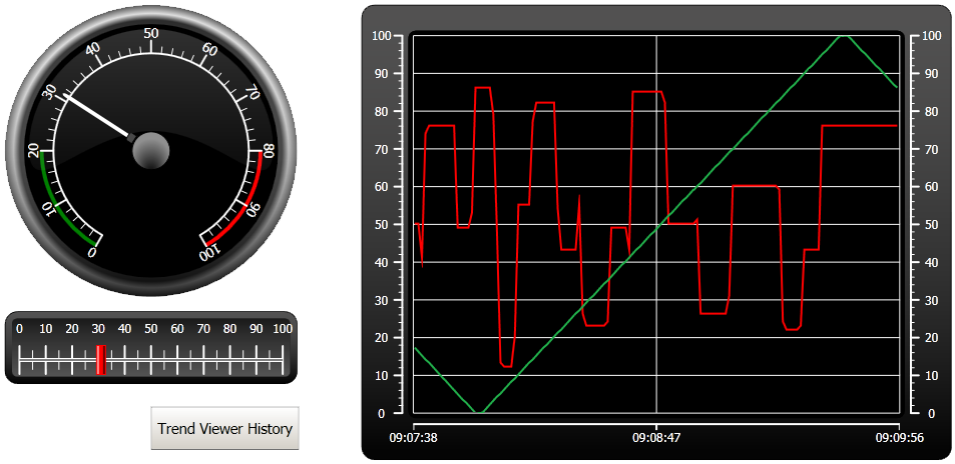
7.3 Trend Viewer History

7.3.1 Actions

1. Open Screen1 for editing in the desktop.
2. Place a button to the left of the trend viewer object. Label the button "Trend Viewer History".
3. Keep the button selected and click on the **Actions** tab. Select **Trend Viewer History**, located in the **Trend Viewer** group, from the drop-down list for the **Click** action. Select **TrendViewer1** from the **Select Trend Viewer** drop-down list, and **On**-mode.
4. Select the trend viewer object and click on the **Actions** tab. Select **Trend Viewer History** for the **MouseDown** action. Select **TrendViewer1** from the **Select Trend Viewer** drop-down list, and **Off**-mode.

7.4 Running Historical Trend Viewer Test

1. Run the project.



2. Test that you can switch to the historical trend viewer with the Trend Viewer History button.
3. Go back to real-time trend viewer by clicking on the trend viewer object.

8 Alarm Management

Alarms are used to make the operator aware of events that require immediate action. An alarm is set when a certain condition is met. An alarm condition is designed as a logical evaluation of a tag value. Alarms can be divided into groups to create an order of priority.

Objective:

- Configuring the alarm list and designing an alarm object.

8.1 Alarm Indicator

When an alarm goes active, the Alarm Indicator becomes visible to alert the operator, regardless of which screen is active.

The appearance of the alarm indicator depends on the current alarm status.

Select **General** settings in alarm server properties page to decide for which alarm statuses to show the alarm indicator.

The Alarm Indicator will show the most severe alarm in the alarm list with the following indications:

- Flashing red when there is any active, unacknowledged alarm.
- Flashing green when no active alarms exist, but inactive unacknowledged alarms exist.
- Flashing green when there are only active acknowledged alarms.

The Alarm Indicator disappears when all alarms are both acknowledged and have returned to inactive status.

8.2 Alarm Server

- Click on **Alarm Server** in the Project Explorer to open the Alarm Server configuration page.

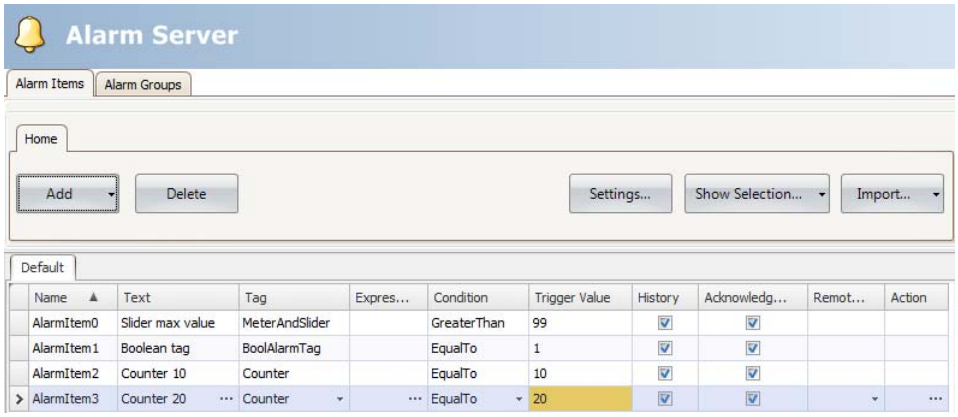
8.2.1 Alarm Groups

The **Alarm Groups** tab is used to set up multiple alarm groups, e.g. when a project needs separate management of alarms for independent functions. In this example, the default alarm group is used.

8.2.2 Alarm Items

1. Select the **Alarm Items** tab. Add alarms based on tags in the controller tag list.
2. Define a digital tag (named BoolAlarmTag) directly in the alarm list by clicking **Add** in the tag selection list.

   This tag will be internal unless connected to a controller on the Tags configuration page, and using an internal tag works just fine for this example project. See chapter *Internal Tags* for information.



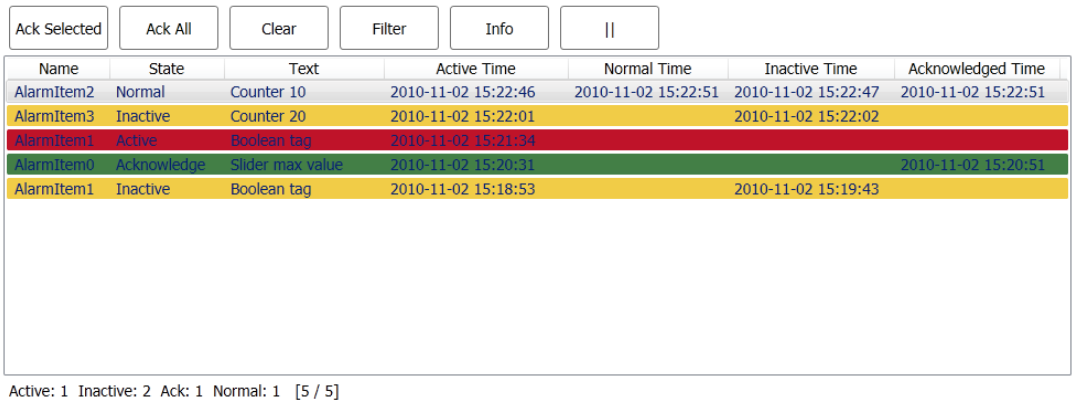
3. Leave the columns for **Acknowledge Required** and **History** checked. Leave the other fields empty.
4. Make sure that all alarm tags can be controlled from the project screens or that they will be triggered by other mechanisms (the counter will trigger AlarmItem2 and AlarmItem3 after 10 and 20 seconds respectively).
5. Place a button to the left of the trend viewer object. Label the button "Set Alarm".
6. Keep the button selected. On the **Actions** tab, select **Toggle Tag**, located in the **Controller** group, from the drop-down list for the **Click** action. Select **BoolAlarmTag** in the **Select Tag** field.

8.3 Alarm Viewer

1. Open Screen2 for editing in the desktop.
2. Select the **AlarmViewer** from the **Objects** group and place it in the screen.
   The columns and button placement can be customized in an alarm object.
3. Select the alarm object on the screen, and click on the **General** tab. In the **Buttons** group, click on **Position** and select to place the buttons against the **Top** border.
4. Adjust the size so that all button controls in the alarm object are visible.
5. Use **Show Columns** in the **Settings** group to customize the alarm information and the order of the columns in the Alarm Viewer.

8.4 Running Alarm Test

1. Run the project.
2. Test to trigger the alarms.



3. Press the **Ack All** button and observe the Alarm Indicator.
4. Make sure that all alarm tags are inactive. Press the **Ack All** button to acknowledge all alarms.
5. Press **Clear** to remove all alarms of normal status (acknowledged and inactive).

9 Recipes

Recipes are used to set or save a predefined group of tags in one operation.

Objective:

- Creating and using recipes to change multiple values.

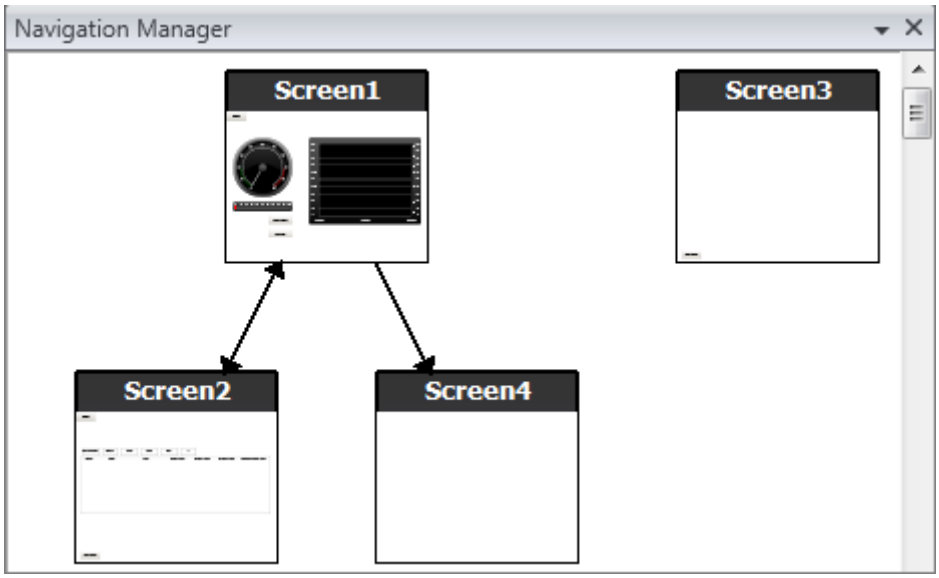
9.1 Creating Recipe Tags

- Create a group of controller tags that the recipe should affect. Use three integer values to set weight, length and width of an imaginary item.

	Tag			Controllers	
	Name	Data Type	Access Right	Data Type	Controller 1
	MeterAndSlider	DEFAULT	ReadWrite	INT16	D0
	Tag2	DEFAULT	ReadWrite	INT16	D1
	Counter	DEFAULT	ReadWrite	INT16	C0
	BoolAlarmTag	DEFAULT	ReadWrite	DEFAULT	
	Weight	DEFAULT	ReadWrite	INT16	D2
	Length	DEFAULT	ReadWrite	INT16	D3
I	Width	DEFAULT	ReadWrite	INT16	D4

9.1.1 Adding a Screen

1. Open the Navigation Manager. Point to Screen1 and drag a connection to an empty spot in the screen navigation area.



   A new screen (Screen4) is created in the project.
2. Open Screen4 and select the **Home** tab. In the **Screen** group, select Screen3 in the **Background Screen** drop-down list.

   This enables navigation from Screen4 to Screen1.

9.1.2 Adjusting Navigation Buttons

- Open Screen1. Select the button labeled Screen4 (in the upper left corner) and move it so the button placed under it (Screen2) becomes fully visible.

9.2 New Objects

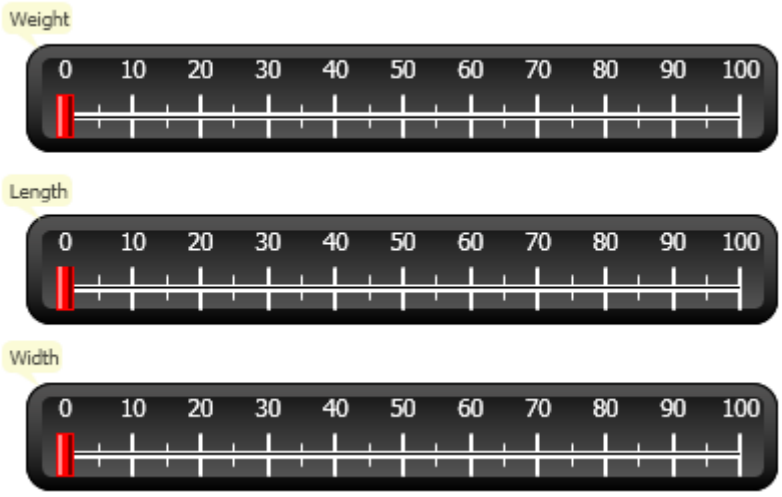
1. Open Screen4 for editing in the desktop, and add a slider.
2. Right-click on the slider and point to **Select Tag...** to open the controller drop-down list in the MiniToolBar. Select **Weight**.

   This connects the Weight tag to the object and the list closes.
3. Press **Ctrl**, then drag the slider across the screen to make a copy of it. Position the new slider and connect it to **Length**. Repeat for the **Width** tag.

9.2.1 Show Info

It is possible to show information about which tag each object is connected to, and if dynamics or actions are configured for the object, by clicking on the **Show/Hide Info** button in the desktop area, or by using the keyboard shortcut **Ctrl + D**.

- Press **Ctrl + D** on the keyboard to check that the tags are correctly bound to the sliders in the screen.



9.3 Recipe Items

1. Click on **Recipe** on the **Insert** tab to add a new recipe.

   The recipe configuration page opens in the desktop. The new recipe is also available from the Project Explorer.
2. Enter a group of tags to set in the recipe on the **Tag Configuration** tab.

	Name	Tag
	RecipeItem1	Weight
	RecipeItem2	Length
I	RecipeItem3	Width

9.4 Saving a Recipe

1. Open Screen4 for editing in the desktop and place a button next to the set of sliders for the recipe tags. Label the button "Save Recipe".
2. Click on the **Actions** tab and select **Save Recipe**, located in the **Recipe** group, from the drop-down list for the **Click** action. Make sure that **Recipe1** is selected.
3. Leave **Recipe data** empty.

9.5 Loading a Recipe

1. Open Screen4 for editing in the desktop, make a copy of the Save Recipe button.
2. Change the label to "Load Recipe", and to load Recipe1 by selecting it for the **Click** action **Load Recipe**.
3. Leave **Recipe data** empty.

9.6 Recipe Data

Create a predefined recipe by defining values on the Runtime Data tab of the recipe configuration page.

1. Open the recipe configuration page by clicking on Recipe1 in the Project Explorer.
2. Click on the **Runtime Data** tab. Enter tag values to set in a recipe. Enter a name (**Runtime Recipe Title**) for the recipe.

	Runtime Recipe Title	RecipeItem1	RecipeItem2	RecipeItem3
	Book	2	25	15
I	TV	30	45	60

3. Open Screen4 for editing. Place a new button next to the set of sliders. Label the button "Load Book".
4. Select **Load Recipe** in the **Click** drop-down list.
5. Select Recipe1, and select **Book** for recipe data.

9.7 Running Recipe Test

1. Run the project.
2. Test to set the sliders to various values and save the values in recipes.
3. Test to load the recipes.
   Check that the sliders change according to the recipe values.

10 Dynamics

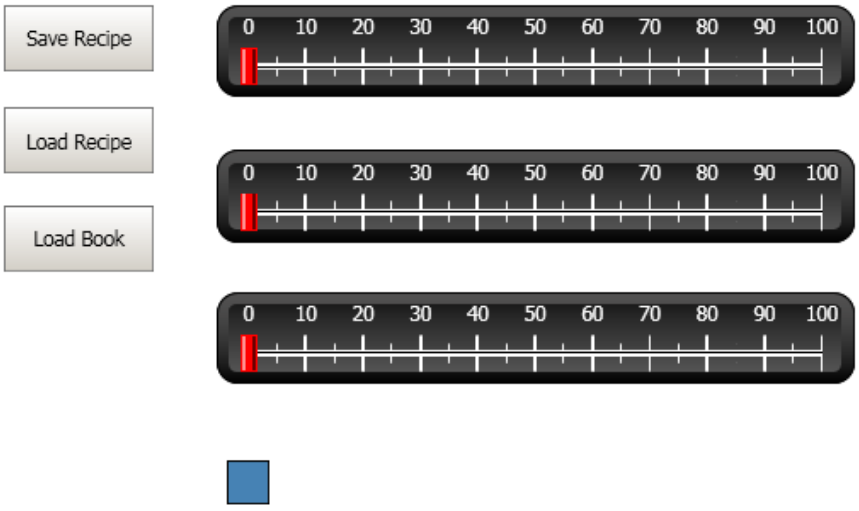
Dynamic object properties are used to move and resize objects based on controller tag values.

Objective:

- Changing size and color of an object based on tag value changes.

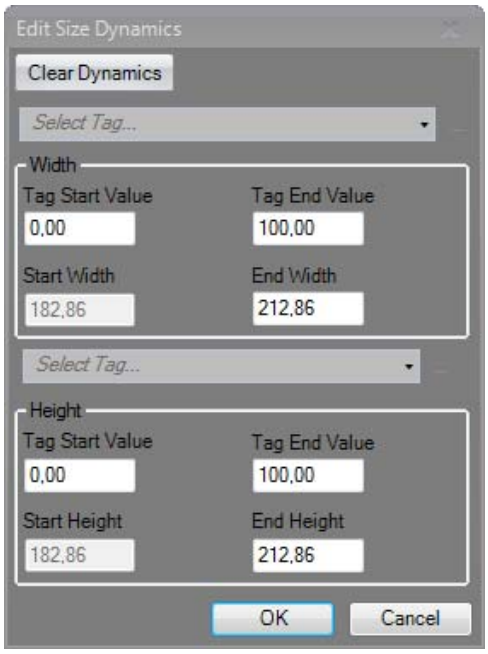
10.1 Creating an Object

- Open Screen4 for editing. Select the **Rectangle** in the **Objects** group of the **Home** tab and place a small square below the set of sliders for the recipe tags.



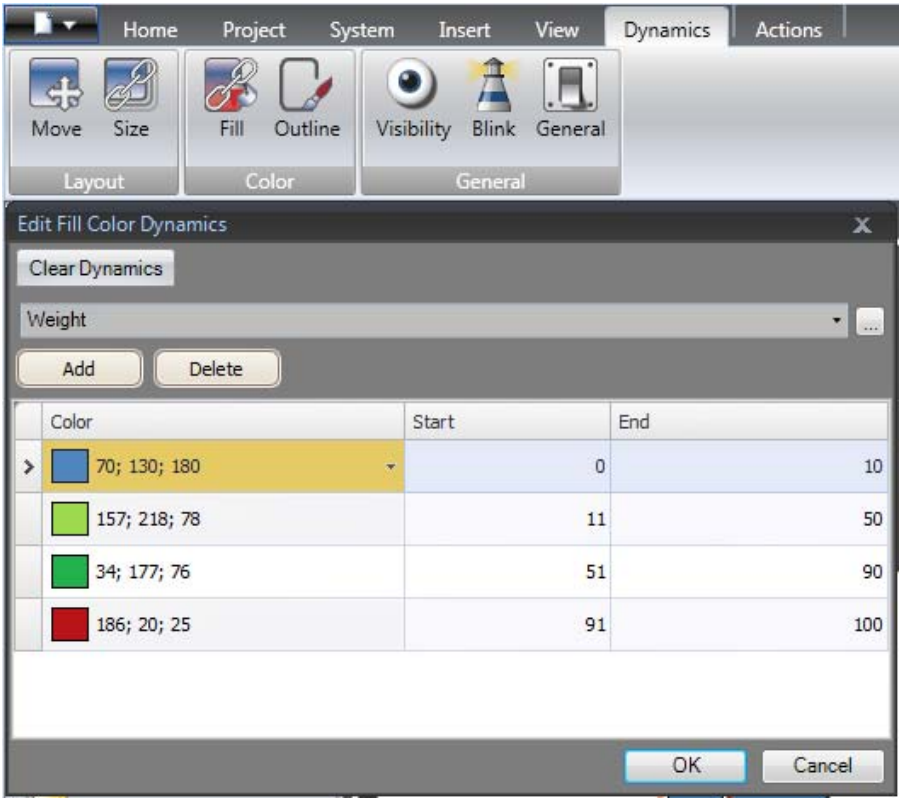
10.2 Resizing an Object

1. Select the square. Click on **Size** on the **Dynamics** tab. Select the **Width** tag for **Width** and the **Length** tag for **Height**.
2. Adjust the enlarged size of the square directly on the screen and note the change of values in the Resize Dynamics Editor window.
3. Set the fields for **Tag Start Value** to the start size values of the square (Start Width, Start Height). Set both fields for **Tag End Value** to "100".



10.3 Coloring an Object

1. Select the square. Click on **Fill** in the **Color** group of the **Dynamics** tab. Assign the Weight tag in the **Select Tag** drop-down list.
2. Adjust the tag values to change the color of the square depending on the Weight tag value. The example in the picture below uses fill color in combination with a gradient.



10.4 Running Dynamics Test

1. Run the project.
2. Test to change the tag values with the sliders and by loading recipes. Observe what happens with the size and color of the small square.

11 Script

Scripts are used to manage functionality for objects. Scripts are written in C#.

Objective:

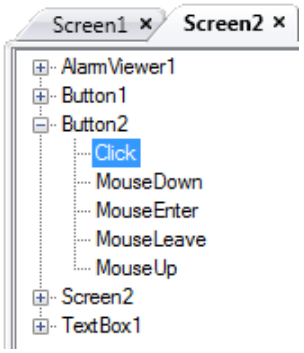
- Inserting a button and text box.
- Writing a script for the button to affect the text in the text box.

11.1 Adding Objects

1. Open Screen2 for editing and add a **TextBox** from the **Objects** group (located under Windows Controls) to the screen.
2. Place a button on the screen and label it "Write Test".

11.2 Script Tab

1. Select the Write Test button.
2. Click on the **Script** tab located in the lower left the desktop area. This will change from Layout view mode to Script view mode.
3. Click on the **Button2** node.



4. Double-click on the **Click** node to start typing script code for the Click event for Button2.

   A context sensitive name completion feature (IntelliSense) can be activated during typing with Ctrl + [Spacebar] and it triggers automatically when a period ('.') is typed after a code element.

5. Type the following as the click event code:

```
TextBox1.Text= "Test";
```

C# code on the Script tab:

```
public partial class Screen2
{

    void Button2_Click(System.Object sender, System.EventArgs e)
    {
        TextBox1.Text= "Test";
    }
}
```

11.3 Running Script

1. Run the project.
2. Click on the **Write Test** button and check that the text string assigned with the script code now appears in the text box.

12 Internal Tags

Internal tags can be used to calculate values that need not be represented in the controller, for example information only for the operator.

Objective:

- Writing a script to perform a calculation of the area using the length and width tags.
- Showing the result with an internal tag.

12.1 Adding Internal Tags

- Click on **Tags** in the **Project Explorer**.
  The tags configuration page opens on the desktop.

12.1.1 Area Tag

- Add a tag and label it "Area". Change the data type to **FLOAT**. The data type for this tag and the following tag is set for the tag; not for the controller.

12.1.2 Calculation Tag

- Add a tag and label it "Calc" and select the data type **BIT**.

	Tag			Controllers	
	Name	Data Type	Access Right	Data Type	Controller1
	MeterAndSlider	DEFAULT	ReadWrite	INT16	D0
	Tag2	DEFAULT	ReadWrite	INT16	D1
	Counter	DEFAULT	ReadWrite	INT16	C0
	BoolAlarmTag	DEFAULT	ReadWrite	DEFAULT	
	Weight	DEFAULT	ReadWrite	INT16	D2
	Length	DEFAULT	ReadWrite	INT16	D3
	Width	DEFAULT	ReadWrite	INT16	D4
	Area	FLOAT	ReadWrite	DEFAULT	
I	Calc	BIT	ReadWrite	DEFAULT	

  Leaving the Controllers column empty keeps the tag internal, not connected to a controller.
- Switch to the **Script** view mode and locate the **Calc** tag. Click on the **Calc** tag node and double-click to open the **Value Change** node.
  To access data and methods in C# control code the keyword "Globals" is used. The example uses explicit type casting ("(double)"), which is necessary for an overloaded operator (multiplication).

- Calculate the area in the **Value Change** node:

```
Globals.Tags.Area.Value =
    (double) Globals.Tags.Longitud.Value *
    (double) Globals.Tags.Ancho.Value / 100;
```

12.2 Creating an Analog Numeric

1. Open Screen4 for editing. Place an **Analog Numeric** object below the set of sliders for the recipe tags, clear from the rectangle object.
2. Right-click on the analog numeric and connect it to the **Area** tag.
3. Keep the object selected, select **Decimal** for **Display Format** and set **Number of Decimals** to **2** on the **General** ribbon tab.

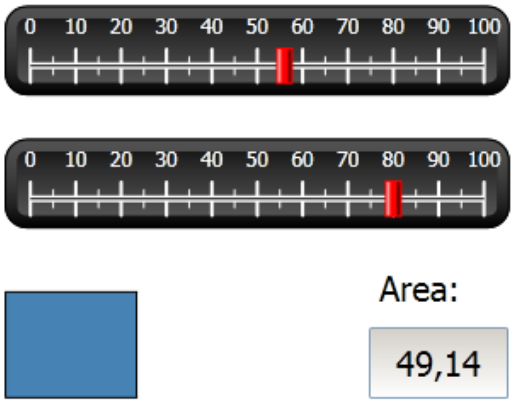
4. Use a **Text** object to place an explaining text (e.g. "Area:") in connection with the analog numeric object.

12.2.1 Calculation Trigger

1. Open Screen4 for editing. Select the Rectangle object.
2. Go to the **Actions** tab and select **Toggle Tag**, located in the **Controller** group, from the drop-down list for the **MouseDown** action. Select the **Calc** tag.

12.3 Running Internal Tags Test

1. Run the project.
2. Test to set the sliders to various values. Click on the dynamic rectangle area and observe the change of the Analog Numeric control.



13 Object Browser

An overview of all objects included in a screen can be displayed in the Object Browser.

Objective:

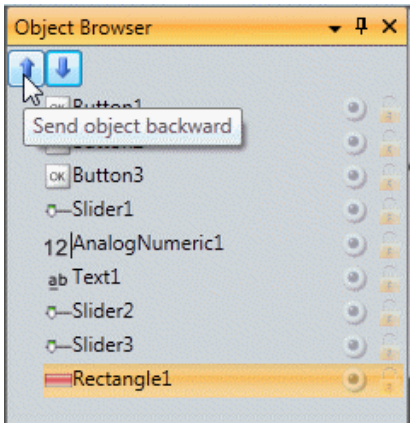
- Positioning objects in depth
- Locking objects
- Making objects invisible

13.1 Adding a Graphical Element

1. Open Screen4 for editing.
2. Select the **Rectangle** from the **Objects** group of the Home ribbon tab.
3. Resize the rectangle to fit as background of the group of sliders and buttons.
   The rectangle now totally obscures the other objects.

13.2 Using the Object Browser

1. Select **Object Browser** from the View ribbon tab.
2. Click the **Send object backward** button while the rectangle is selected until all the buttons and sliders are visible.



3. Click the **Lock** button while the rectangle is selected.
4. Try to move the rectangle on the screen by dragging it.
   The object is locked and cannot be moved. It cannot be selected in any way.
5. Select one of the buttons and click the **Visibility** button.
   The button is hidden. But if the project is run in iX Runtime, the object will show as normal.

14 Multiple Texts

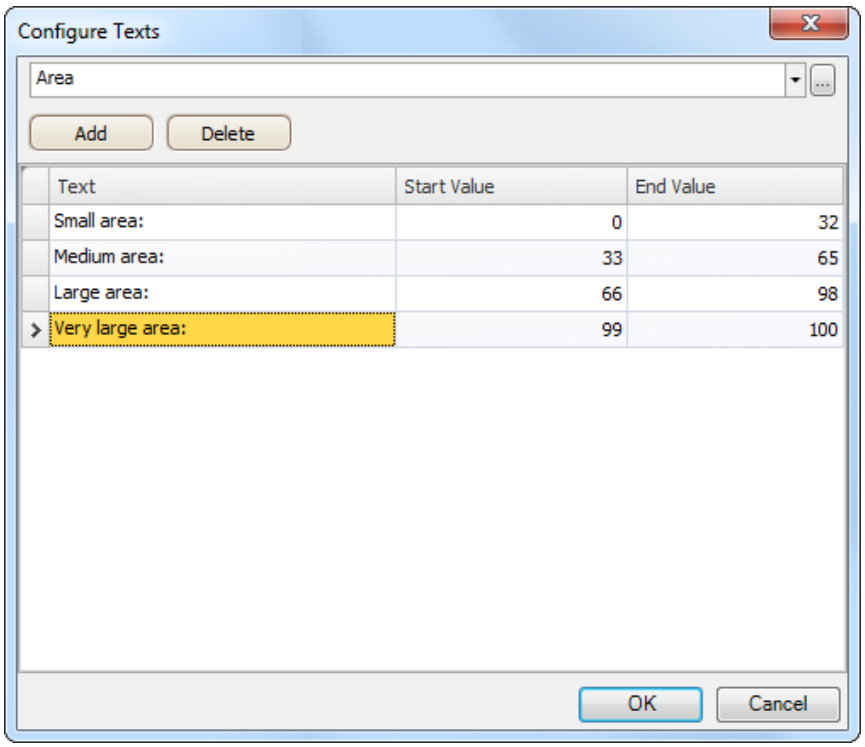
Text objects can be used to display textual information, and can also reflect changes in controller tags.

Objective:

- Presenting a variant text message that reflects the changes of the calculated area.

14.1 Configuring Texts

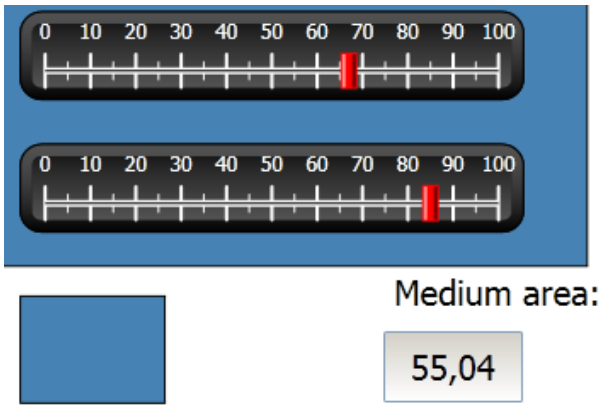
1. Open Screen4 for editing. Select the Text object labeled "Area:" and click on the **General** tab.
2. Click on **Configure Texts** in the **Text** group. Connect the text to the **Area** tag.
3. Add text strings and edit the intervals according to the example below.



With the default setting for the text object, **Autosize**, there is no need to adjust the object in order to make the longest string fit in runtime.

14.2 Running Multiple Texts Test

1. Run the project.
2. Test to set the sliders to various values. Click on the dynamic rectangle and observe the change of the analog numeric control. Check that the text is updated as well.



15 Security

Access to objects and actions in the project can be limited using security groups and user passwords.

Objective:

- Adding user names and setting passwords.
- Setting up login and logout control.
- Restricting access of recipe handling.

15.1 Security Configuration

- Click on **Security** in the **Project Explorer** to open the configuration page.

15.1.1 Security Groups

Security is handled by dividing users into security groups. These are configured on the **Groups** tab. In this example, the two default security groups, **Administrators** and **Operators**, are used.

15.1.2 Users

1. Select the **Users** tab on the Security configuration page.
2. Add a user that belongs to both security groups; Administrators and Operators.
3. Add another user that belongs to Operators group only.

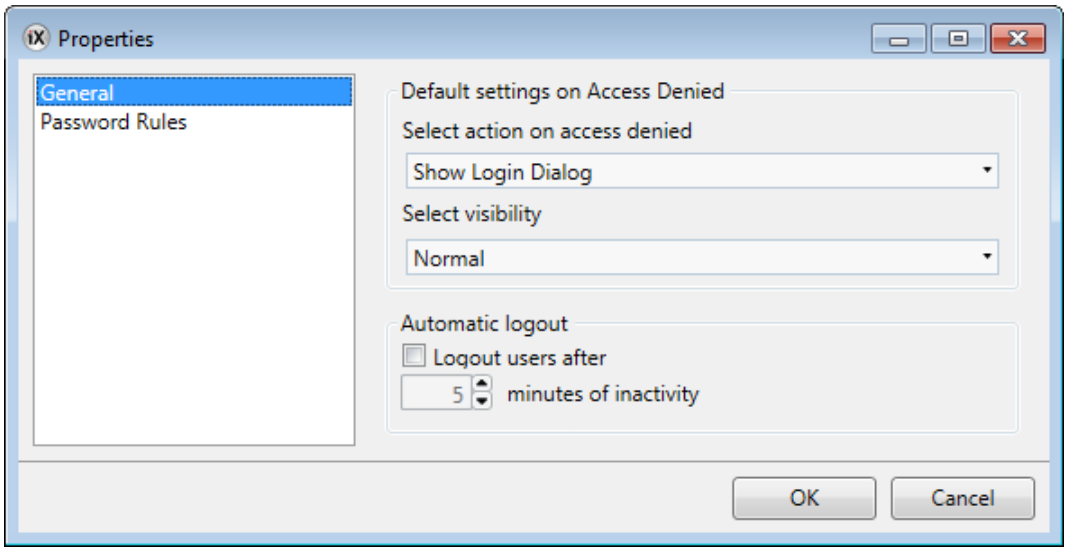
	Username	Password	Description	Groups
	Administrator	********		Administrators
	SuperUser	********		Administrators, Operators
I	User1	oxpy		Operators

Note:
The password is converted to asterisks as you leave the password input cell.

15.2 Login Behavior on Access Denied

A login dialog can be displayed whenever a user tries to access an object that is restricted for the user group that the user belongs to.

1. Select the **Users** tab, and click the **Settings** button.
2. Select **Show Login Dialog** for action on access denied, and **Normal** for visibility.



15.3 Creating a Logout Button

1. Open Screen3 (the background screen) for editing.
2. Place a button next to the "Start Screen" button. Label the button "Logout".
3. Select **Logout**, located in the **Security** group, from the drop-down list for the **Click** action.

15.4 Object Security

1. Open Screen4 for editing.

15.4.1 Administrators

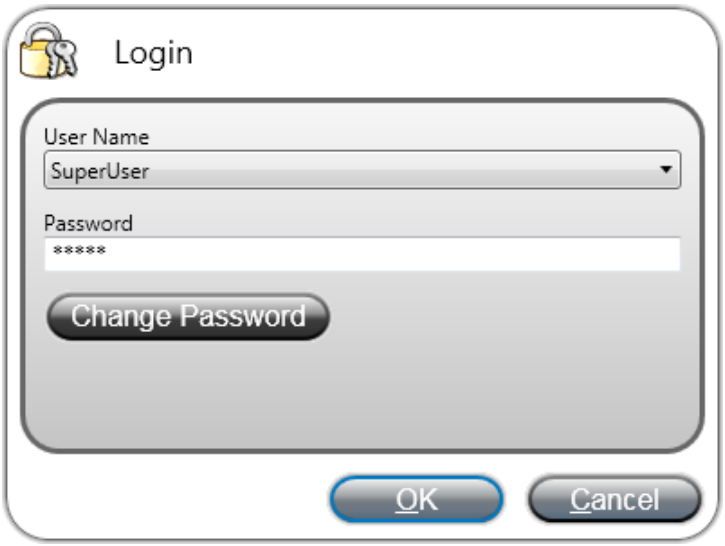
1. Right-click on the **Save Recipe** button and select **Administrators** for **Select Security Groups**.

15.4.2 Users

1. Right-click on the **Load Recipe** button and select **Operators** for **Select Security Groups**.

15.5 Running Security Test

1. Run the project.
2. Test to make sure that it is no longer possible to load or save recipes without logging in, and that the login dialog opens when any of the buttons are pressed.
3. Login as Administrator and save a recipe.
4. Test to load a recipe.
   The login dialog opens.
5. Login as User1 and load a recipe.
6. Test to save a recipe.
   The login dialog opens.
7. Login as SuperUser. Test to save and load recipes.

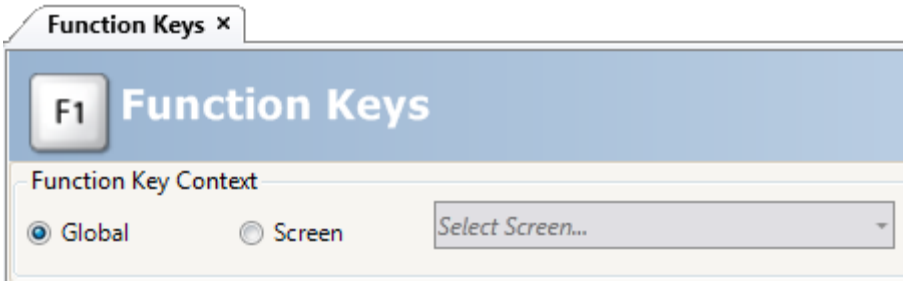


8. Log out.
9. Test that it is no longer possible to load or save recipes.

16 Function Keys

Function keys can be used to perform actions and execute scripts. This allows operator control of data and screen functionality independent of which screen is active.

Function keys can also be configured as local function keys, which means that they are applicable individual screens. In this example, global function keys are used.



Objective:

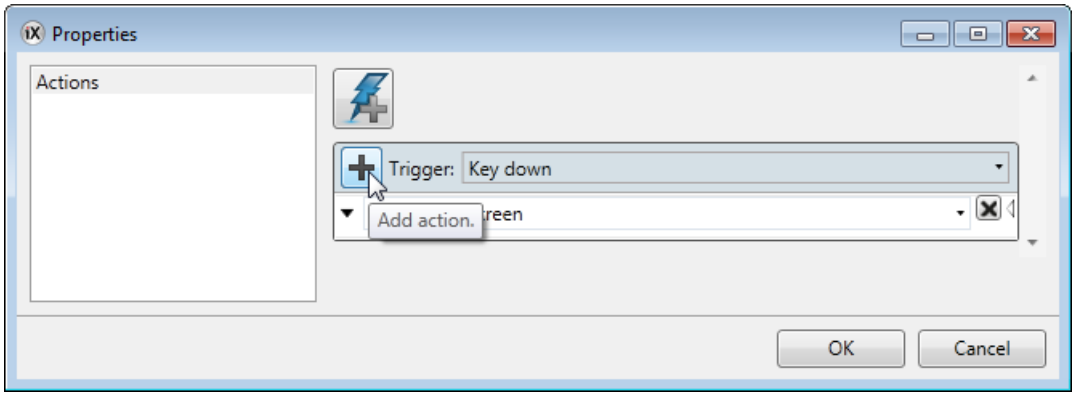
- Programming actions for function keys to change screen, set controller tag values, recipe management and display the login dialog.
- Programming function keys to execute scripts.

16.1 Defining Function Key Actions

1. Click on **Function Keys** on the **Insert** ribbon tab.

16.1.1 Show Screen

1. Click on ... under **Action** on the row for function key **F1** to open the **Properties** dialog. Click **Add** and select **Show Start Screen**, located in the **Screen** group, from the **New Action** dialog.



2. Click on the row for function key **F2**. Select **Show Screen** as action, and **Screen2** from the screen drop-down list.

3. Set up function key **F3** to show Screen4.

16.1.2 Security

- Click on the row for function key **F4**. Select **Login**, located in the **Security** group, as action.

16.1.3 Setting Controller Tag Values

1. Click on the row for function key **F5**. Select **Set Analog** as action, located in the **Controller** group, and the **Weight** tag from the **Select Tag** drop-down list. Specify the analog value **50**.
2. Set up function keys **F6** and **F7** to control the **Length** and **Width** tags.

16.1.4 Recipe

- Set up function key **F8** to load **Recipe1**, and function key **F9** to save **Recipe1**. Leave **Recipe data** empty.

16.1.5 Setting Time Zone, Region and Daylight Saving

- Set up function keys **F10** to set time zone, region and daylight saving. The action is located in the **Other** group.

Function Key	Actions
F1	Show Start Screen
F2	Show Screen
F3	Show Screen
F4	Login
F5	Set Analog
F6	Set Analog
F7	Set Analog
F8	Load Recipe
F9	Save Recipe
F10	Set Time Zone, Region and Daylight Saving ···

Show Start Screen

16.2 Defining Function Key Scripts

Function keys can also be used to trigger scripts.

16.2.1 Area Calculation

Program a function key with the area calculation for the rectangle object:

1. Click on the row for function key **F11**. Select **Script** view mode by clicking on the **Script** tab at the bottom of the screen.
2. Click on the **F11** node, and double-click on its **KeyDown** node.
3. Calculate the area on the KeyDown event with this code:

```
Globals.Tags.Area.Value =
    (double) Globals.Tags.Longitud.Value *
    (double) Globals.Tags.Ancho.Value / 100;
```

Using a function key script eliminates the need for the separate trigger tag (**Calc**).

16.2.2 Acknowledge All Alarms

Program a function key with an "Acknowledge All" for alarms:

1. Select the **KeyDown** node for function key **F12**.
2. Type the following KeyDown event code:

```
Globals.AlarmServer.Acknowledge();
```

C# code in the script tab:

```
public partial class FunctionKeys
{

    void F11_KeyDown(System.Object sender, System.EventArgs e)
    {
        Globals.Tags.Area.Value =
            (double) Globals.Tags.Longitud.Value *
            (double) Globals.Tags.Ancho.Value /100;
    }

    void F12_KeyDown(System.Object sender, System.EventArgs e)
    {
        Globals.AlarmServer.Acknowledge();
    }
}
```

16.3 Running Function Key Test

1. Run the project.
2. Test that the defined functions keys (on the PC keyboard) perform the defined actions.

17 Cross Reference

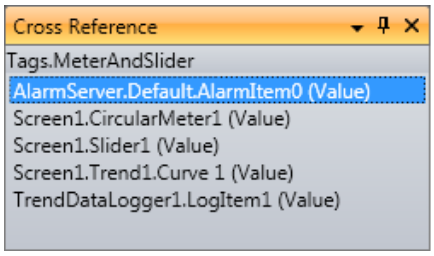
The Cross Reference tool provides an overview of where a specific tag is used in the current project.

Objective:

- Locating all occurrences of a specific tag quickly.

17.1 Using the Cross Reference Tool

1. Select the **MeterAndSlider** tag in the **Tags** configuration page, and click the **Cross Reference** button.
   The Cross Reference tool is displayed.



2. Double-click on the first item in the list.
   The Alarm Server configuration page opens on the desktop.
3. Double-click on the third item in the list.
   Screen1 opens on the desktop, and Slider1 is selected.