IL07 – Introduction to FactoryTalk Optix Visualization Platform Hands-On Lab

Lab Book

Rockwell Automation

Important User Information

This documentation, whether, illustrative, printed, "online" or electronic (hereinafter "Documentation") is intended for use only as a learning aid when using Rockwell Automation approved demonstration hardware, software and firmware. The Documentation should only be used as a learning tool by qualified professionals.

The variety of uses for the hardware, software and firmware (hereinafter "Products") described in this Documentation, mandates that those responsible for the application and use of those Products must satisfy themselves that all necessary steps have been taken to ensure that each application and actual use meets all performance and safety requirements, including any applicable laws, regulations, codes and standards in addition to any applicable technical documents.

In no event will Rockwell Automation, Inc., or any of its affiliate or subsidiary companies (hereinafter "Rockwell Automation") be responsible or liable for any indirect or consequential damages resulting from the use or application of the Products described in this Documentation. Rockwell Automation does not assume responsibility or liability for damages of any kind based on the alleged use of, or reliance on, this Documentation.

No patent liability is assumed by Rockwell Automation with respect to use of information, circuits, equipment, or software described in the Documentation.

Except as specifically agreed in writing as part of a maintenance or support contract, equipment users are responsible for:

- properly using, calibrating, operating, monitoring and maintaining all Products consistent with all Rockwell Automation or third-party provided instructions, warnings, recommendations and documentation;
- ensuring that only properly trained personnel use, operate and maintain the Products at all times;
- staying informed of all Product updates and alerts and implementing all updates and fixes; and
- all other factors affecting the Products that are outside of the direct control of Rockwell Automation.

Throughout this manual we use the following notes to make you aware of safety considerations:

WARNING: Identifies information about practices or circumstances that can cause an explosion in a hazardous environment, which may lead to personal injury or death, property damage, or economic loss.

ATTENTION: Identifies information about practices or circumstances that can lead to personal injury or death, property damage, or economic loss. Attentions help you identify a hazard, avoid a hazard, and recognize the consequence.

IMPORTANT

Identifies information that is critical for successful application and understanding of the product.

Labels may also be on or inside the equipment to provide specific precautions.

	**SHOCK HAZARD:** Labels may be on or inside the equipment, for example, a drive or motor, to alert people that dangerous voltage may be present.
	**BURN HAZARD:** Labels may be on or inside the equipment, for example, a drive or motor, to alert people that surfaces may reach dangerous temperatures.
	**ARC FLASH HAZARD:** Labels may be on or inside the equipment, for example, a motor control center, to alert people to potential Arc Flash. Arc Flash will cause severe injury or death. Wear proper Personal Protective Equipment (PPE). Follow ALL Regulatory requirements for safe work practices and for Personal Protective Equipment (PPE).

Table of Contents

IL07 - Introduction to FactoryTalk Optix Visualization Platform Hands-On Lab

Before You Begin

About this Lab

This is an introductory Lab that explores the features and functionality of FactoryTalk Optix.

In the first section of the lab, you will be creating a brand-new project to get hands-on experience with fundamental concepts and configure the framework for the rest of the lab. The remaining sections are independent of other lab sections so you may select what you wish to learn about in any order.

The lab sections contain an explanation and feature details for that particular section in paragraphs highlighted in grey color. It is not necessary to read these explanations, but a more thorough understanding of the features presented will be attained if time is taken to read through them.

In this lab, you will:

- Create your first project and configure the main window
- Configure Communication to ControlLogix and an OPC UA Server
- Configure screen types
- Add objects (data types) to screen types
- Configure navigation
- Use the Emulator to view your project in runtime.

Select from the following:

- Configure alarms and alarm history
- Configure a data logger, and display data in a Data Grid and Trend
- Create security users and groups
- Configure Language switching

Duration

- Create Your First Project – 30 minutes
- Alarming – 15 minutes
- Configure a Datalogger - 15 minutes
- Security – 15 minutes
- Language Switching - 15 minutes
- Appendix – 15 minutes

Prerequisites

There are no prerequisites required for this lab.

Materials

This training lab requires the following items.

- FactoryTalk Optix Studio version 1.5.0.617 (Release Candidate)
- FactoryTalk Logix Echo 3.01
- Studio 5000 Logix Designer version 36.11
- UaExpert OPC UA Client version 1.6.3
- UaCPPServer 1.8.1

Introduction

Objectives

- Explain "What is FactoryTalk Optix"
- Understand the FactoryTalk Optix components
- Learn about the design environment

What is FactoryTalk Optix?

FactoryTalk Optix is Rockwell Automation's new platform that enhances your HMI and data visualization experience and also augments your capabilities in the Industrial Internet of Things (IIoT), edge computing, and data management. It is an addition to the FactoryTalk suite of industrial automation software and is designed to provide an open and flexible platform for creating custom HMI applications that are tailored to user-specific needs.

With FactoryTalk Optix, users can create:

- **Real-time monitoring**: FactoryTalk Optix provides real-time information about the status of industrial systems and processes, allowing operators to make informed decisions and quickly respond to issues.
- **Data visualization**: FactoryTalk Optix provides a wide range of visualization tools and features, including graphs, charts, and histograms, to help operators better understand and analyze data.
- **User-friendly interface**: FactoryTalk Optix features a modern and intuitive interface that is designed to be easy to use, even for non-technical users.
- **Customization**: FactoryTalk Optix provides a wide range of customization options, including the ability to create custom screens, dashboards, and reports, to meet specific customer requirements.
- **Connectivity**: FactoryTalk Optix is designed to connect to a variety of industrial systems and devices, including Rockwell Automation controllers, and third-party PLCs and automation devices.
- **Scalability**: FactoryTalk Optix is designed to scale from small to large-scale applications with multiple web clients.
- **Extensibility**: OPC UA is core to the platform with support of the OPC UA Companion Specs. Native IoT connectivity and a C# engine to fulfill every customer need.

FactoryTalk Optix consists of different software components:

FactoryTalk Optix Studio:

Integrated development environment with a framework of functional modules for designing and compiling HMI or Internet of Things (IoT) applications. FactoryTalk Optix Studio includes a library of predefined objects that support the modular design of graphical interfaces, features, and logic operations of an HMI application. By using specific C# language scripts, you can automate various actions in the design phase and add customized functions to projects. FactoryTalk Optix Studio comes in either:

- FactoryTalk Optix Studio Standard
    - This is available at no cost to anyone without a license.
- FactoryTalk Optix Studio Pro
    - This is a subscription that enables cloud and collaboration functionality.

FactoryTalk Optix Application:

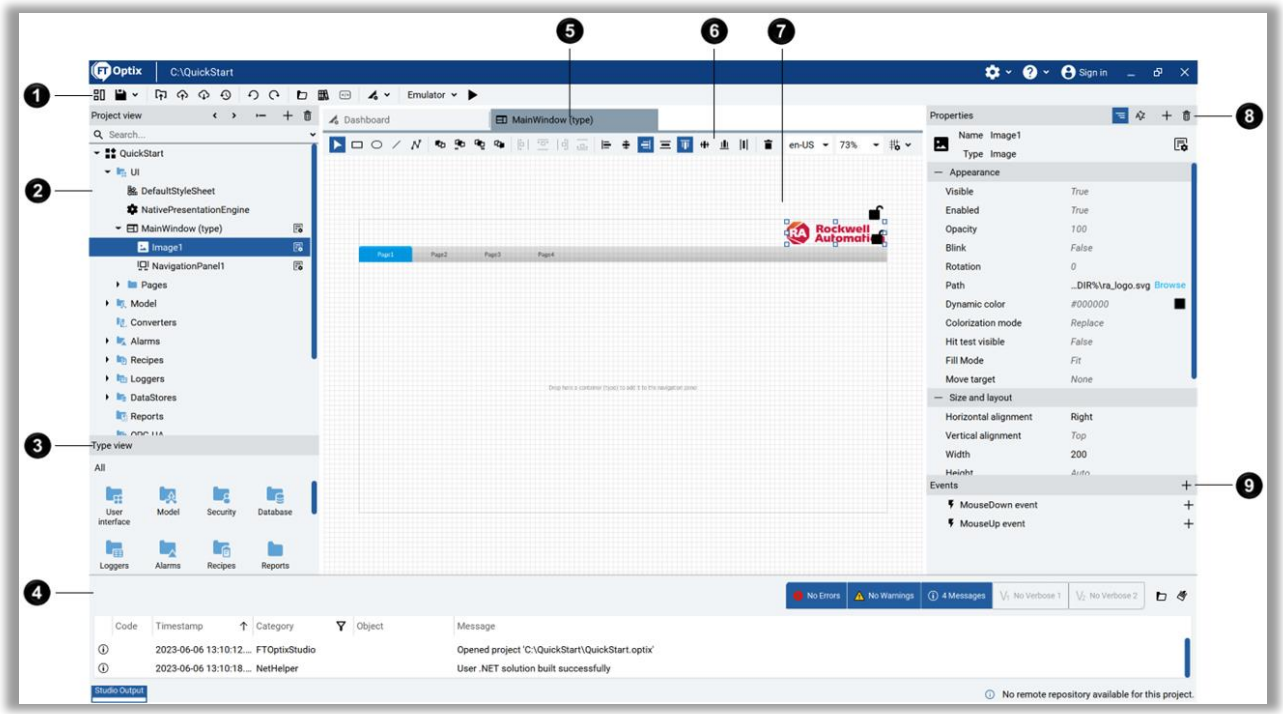
An HMI or IoT application developed and compiled in FactoryTalk Optix Studio.

FactoryTalk Optix Runtime:

Runtime software that is deployed to FactoryTalk Optix Application client systems. The FactoryTalk Optix Runtime installation package contains FactoryTalk Optix Application Update Service (software for updating and deploying FactoryTalk Optix Applications from FactoryTalk Optix Studio to the client devices) and FactoryTalk Optix License Manager.

FactoryTalk Optix Studio Interface

The Design Environment



1 Main toolbar

Commands that apply to all portions of the user interface.

2 Project view

Project information model that displays the content and structure of the project in nodes. A node can be a parent that has child nodes.

3 Type view

Object types and variable types on which instances of objects are based. Predefined types are grouped in folders according to their purpose. Custom types are grouped in folders that reflect the structure in **Project view**.

4 Output pane

Messages related to the operation of the FactoryTalk Optix Application. Applications that are running and are connected to FactoryTalk Optix Studio appear on **Emulator Output**.

5 Tabs

Tabs of open objects being modified in the editor. If you hover over a tab to select ✕ without saving the project, you are prompted to save your changes.

6 Editor toolbar

Commands that apply to the object type being modified in the editor.

7 Editor

Graphic editor for objects. Each editor opens in a new tab and is used for graphic objects and object types, such as tag importers and recipes.

8 Properties pane

Set the properties for the selected node in **Project view** or in the object editor.

9 Events pane

Associate methods with events generated by the currently selected node.

Create Your First Project

Objectives

- Create a new project
- Configure the main window
- Produce screen types and a navigation panel
- Add graphic objects and variables
- Set up Dynamic Links and Complex Dynamic Links using Key-value converters
- Establish communications
- Explore responsive graphics (Optional)

Scenario

In this section of the lab, you will be introduced to some of the core aspects of basic HMI development using FactoryTalk Optix Studio. You will be starting from scratch to create a new project, where you will be guided in:

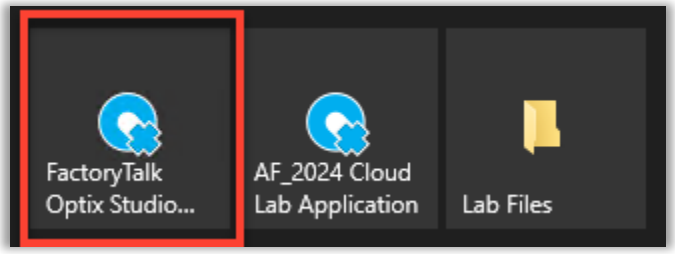
- Creating some simple displays with easily configured navigation
- Setting up communications to both a Logix controller and an OPC UA server
- Exploring the user interface (UI) responsive graphics capabilities of FactoryTalk Optix

Along the way, you will also be introduced to some more optional advanced features and concepts such as:

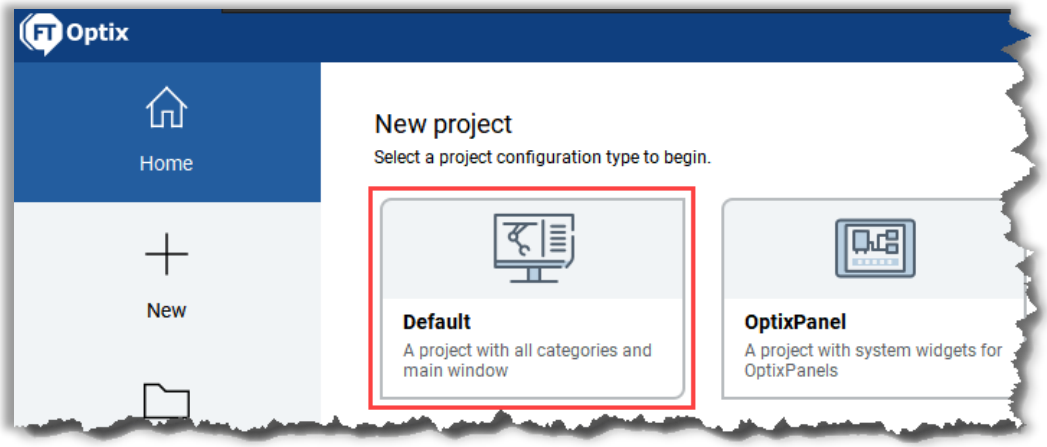
- OPC UA Information Models
- Object-oriented programming utilizing Types
- Advanced Dynamic Linking
- Key-Value Converters

Create a New Project

1. Open FactoryTalk Optix Studio by clicking on the FactoryTalk Optix Studio icon within the Windows Start menu.



2. Under **New project**, select **Default**.

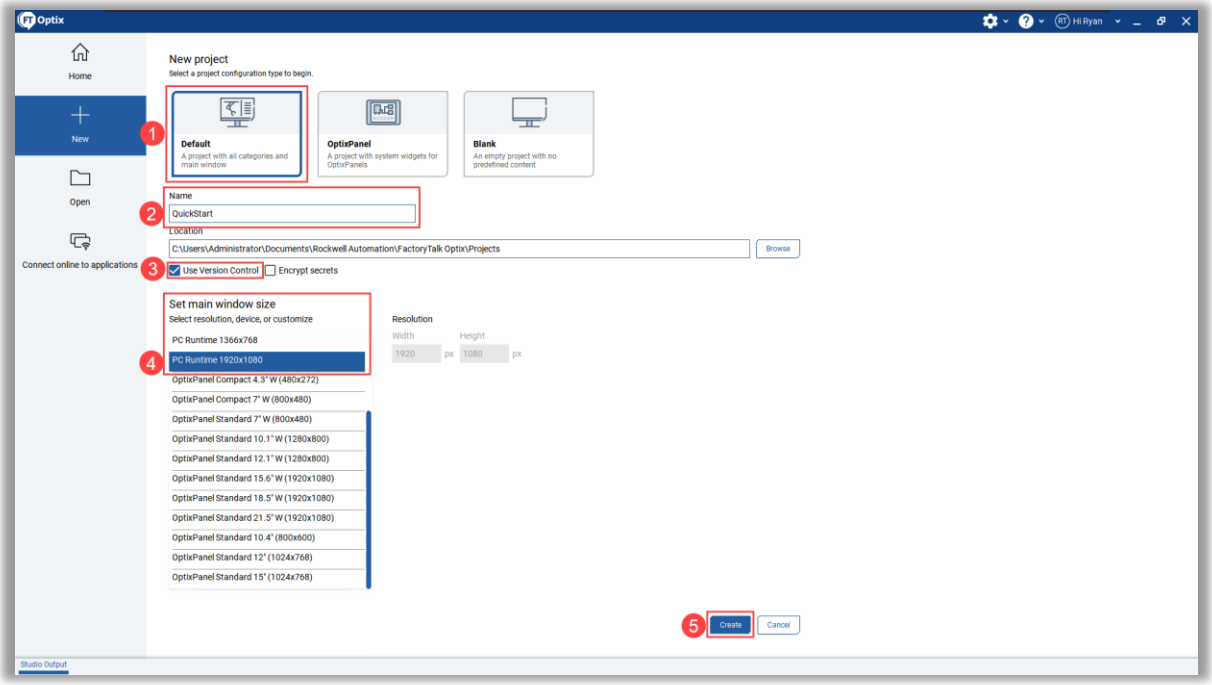


ADDITIONAL PROJECT OPTION

OptixPanel™ graphic terminals give you a PC-like user experience in a sealed HMI appliance. There is no operating system to secure and smaller applications can benefit from the superior price-performance ratio. These graphic terminals are also available in a wide range of screen sizes, bezel options, aspect ratios, and touchscreen technologies that support gestures, such as swipe and pinch, for easier integration on your factory floor.

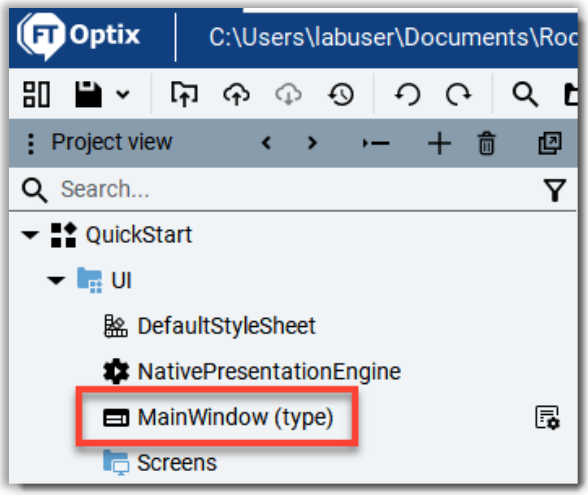
OptixPanel is the 2nd option available. Select this option if you would like to create a project with system widgets for OptixPanels.

3. Under **Name**, enter "QuickStart" as the name of the project.
4. Leave the **Use Version Control** checkbox selected.
5. Select "PC Runtime 1920x1080" under the Set main window size area.
6. Select **Create**.



Examine the Main Window

1. In the **Project view**, expand **UI** and double-click **MainWindow (type)**.



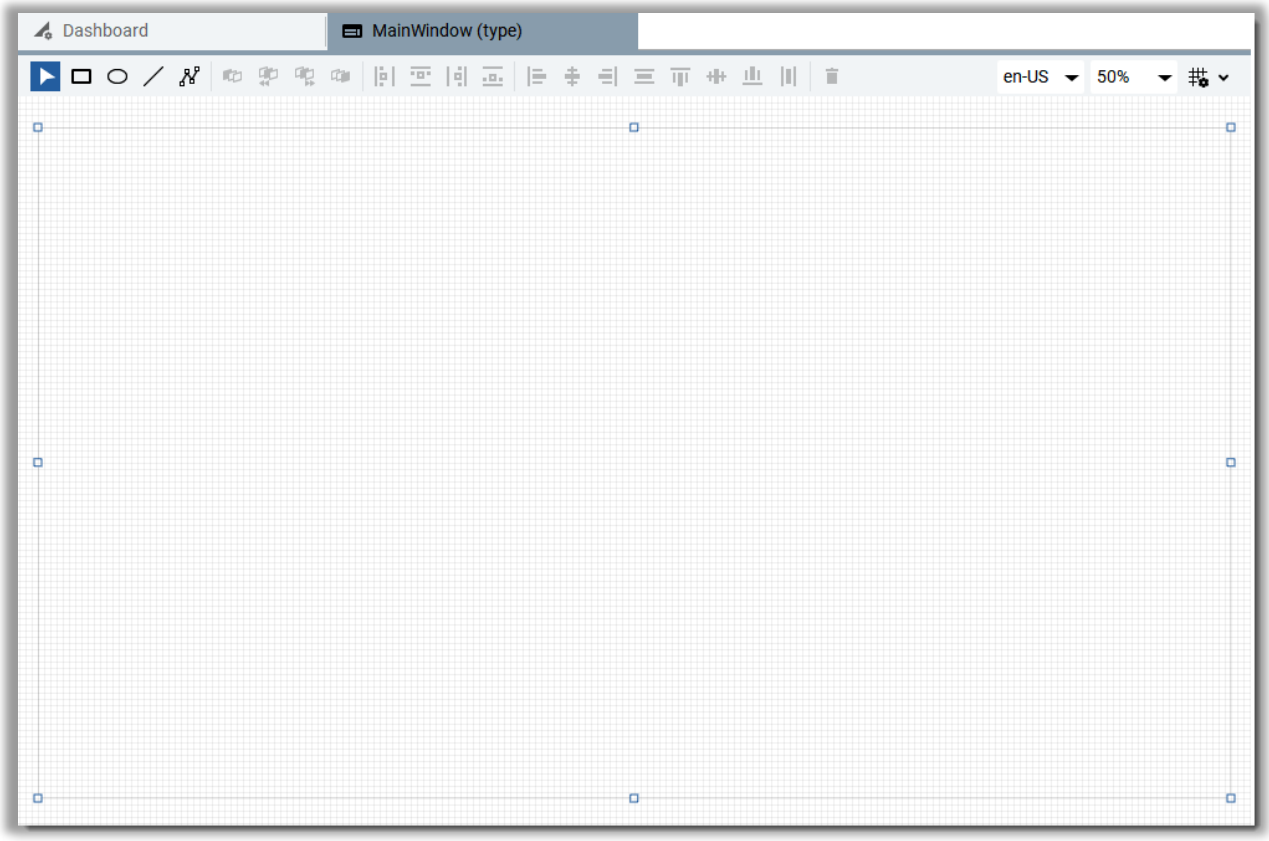
DEFAULT PROJECT COMPONENTS

A default project will provide the following four User Interface related items:

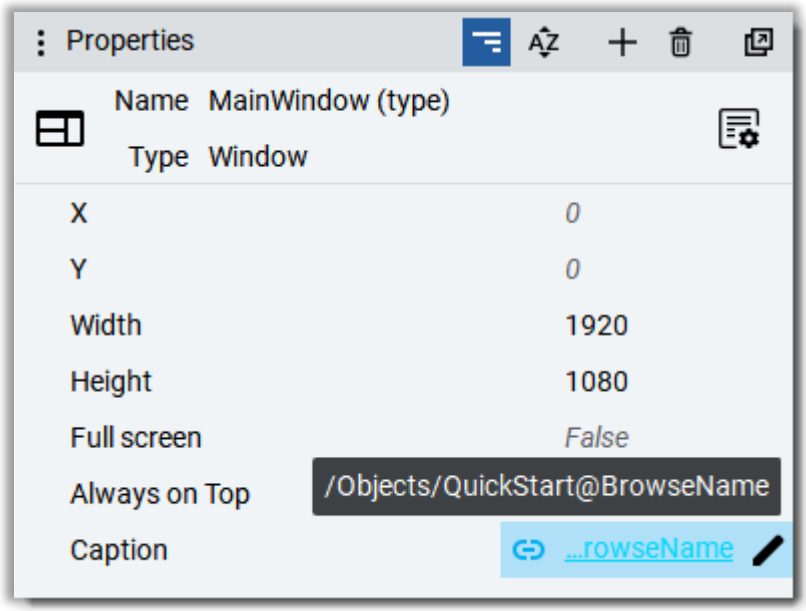
- **Default Style Sheet**: Makes it possible to globally set style properties of all graphical objects in the project
- **Native Presentation Engine**: Use the **Native Presentation Engine** for typical HMI applications that run on targets with an operator panel. There can be only one **Native Presentation Engine** in a FactoryTalk Optix Studio project.
- **Main Window**: This object is the root container of graphical objects
- **Screens**: Folder where screen types can be created

You can set **Full screen** to **True**, however, it is easier to develop and preview applications with the default **False** setting.

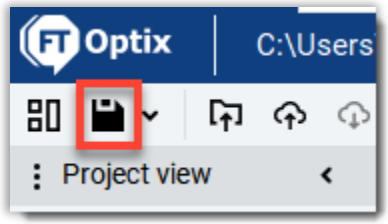
2. The **MainWindow** area should now be shown in the editor. The main window contains all the graphical elements displayed at design time in FactoryTalk Optix Studio and at runtime in your FactoryTalk Optix Application.



3. Take a look at the **Properties** pane on the right. Notice the width and height of the window reflect what you selected when creating the project. Hover over the **Caption** field value. By default, the caption on the **MainWindow** is set to the name of the project, in this case "QuickStart". You can replace the caption with any text you prefer.



4. Save the project by clicking the **Save** icon.



5. To verify that you can now test the project, click the **Play** button beside **Emulator**.

The project should deploy, but since it is currently empty, the only object shown will be the MainWindow with the caption "QuickStart".

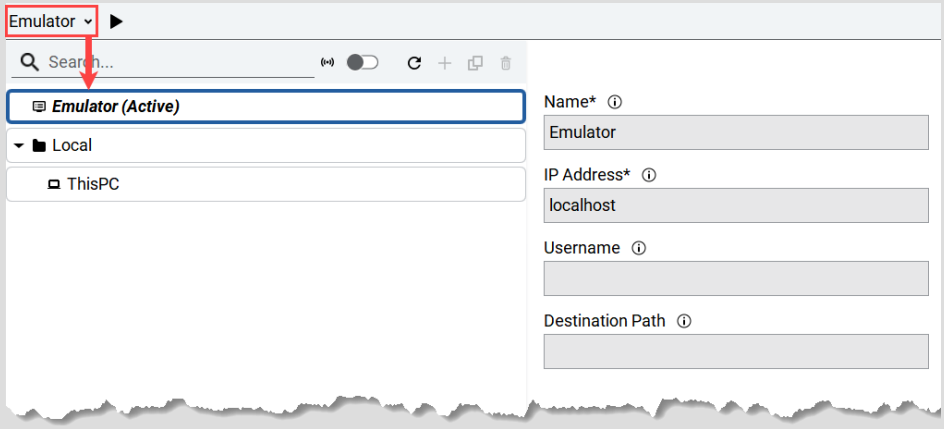
6. Close the Emulator with the **X** icon on the right side of the window.

EMULATOR

FactoryTalk Optix Studio includes an **Emulator** to make developing and testing your runtime project easy.

- The **Emulator** will run for two hours at a time, and it may be restarted as needed after the time limit expires.
- Multiple projects may be emulated at the same time.
- The **Emulator** is the default deployment option when pressing the play button within FactoryTalk Optix Studio.

Note: Click on **Emulator** to expand the runtime deployment configuration window and explore the options available there.

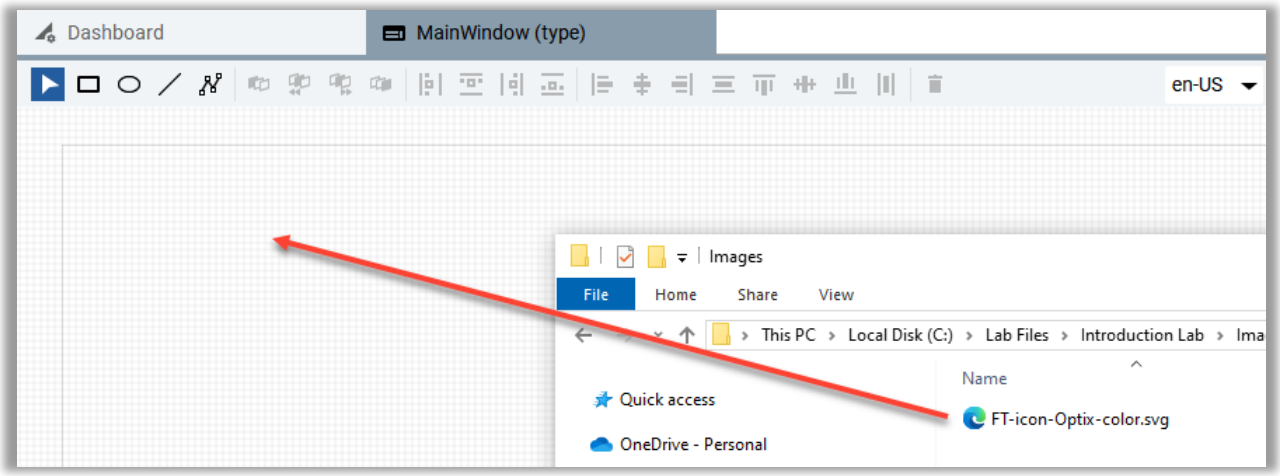


Create a Logo

Headers are a common component in HMIs. You will create a space where content is always shown.

1. In **Project view**, double-click **MainWindow (type)** to open it in the editor
2. Select **Lab Files** from the Windows Start Menu and locate the FT-icon-Optix-color.svg under C:\Lab Files\Introduction Lab\Images.

3. Drag the logo file into the main window area in the editor.

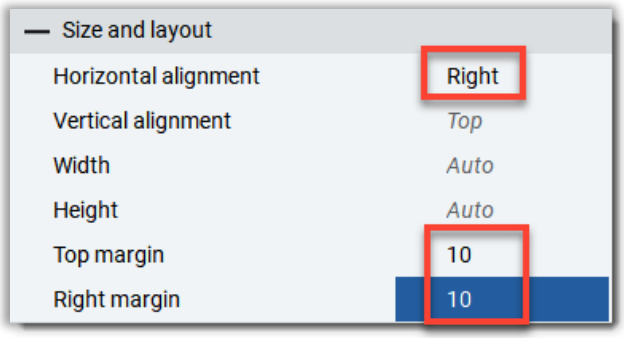


4. Close Windows Explorer.

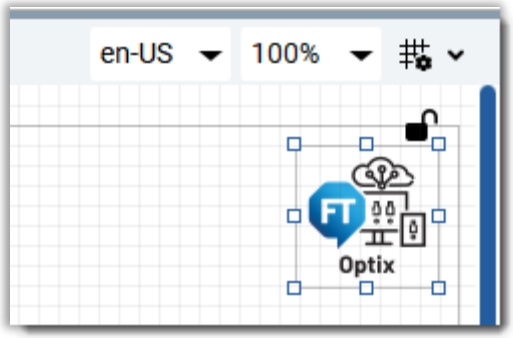
5. Depending on the image resolution and the main window resolution, the logo image might take up too much space or be positioned incorrectly. Change the position of the image so that it always displays on the top right of the window.

6. With the Image still selected, use the dropdown to change the **Horizontal alignment** to "Right" and change **Top margin** to "10" and the **Right margin** to "10".

   Note: Hover over the property and notice how the pencil icon appears . Select this icon to type in new values for a property.



7. If needed, use the scrollbar on the bottom of the editor window to see the image used for the logo.

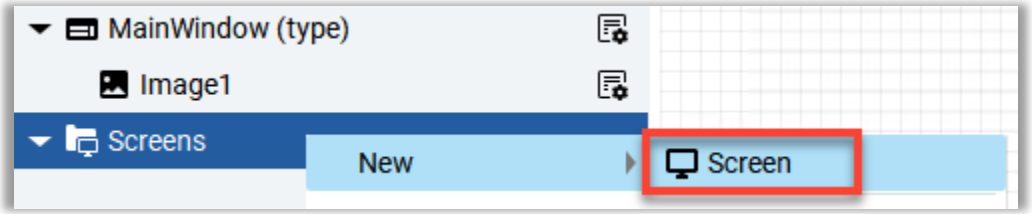


Now that you have created a logo for your project, let's explore how to create screens to structure objects and a navigation object to load the screens.

Using Containers to organize Graphic Objects

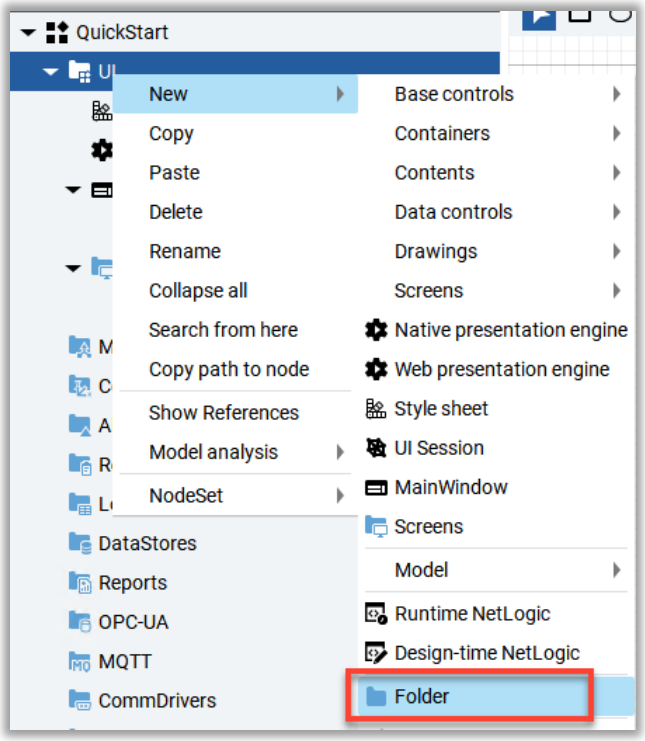
FactoryTalk Optix has several objects to help you organize objects and develop responsive graphics. In this section, you will create a Screen type and create instances of the Screen type in a folder.

1. In the **Project view**, right-click **Screens** and select **New → Screen**. **Screen1 (type)** appears under **Screens**.



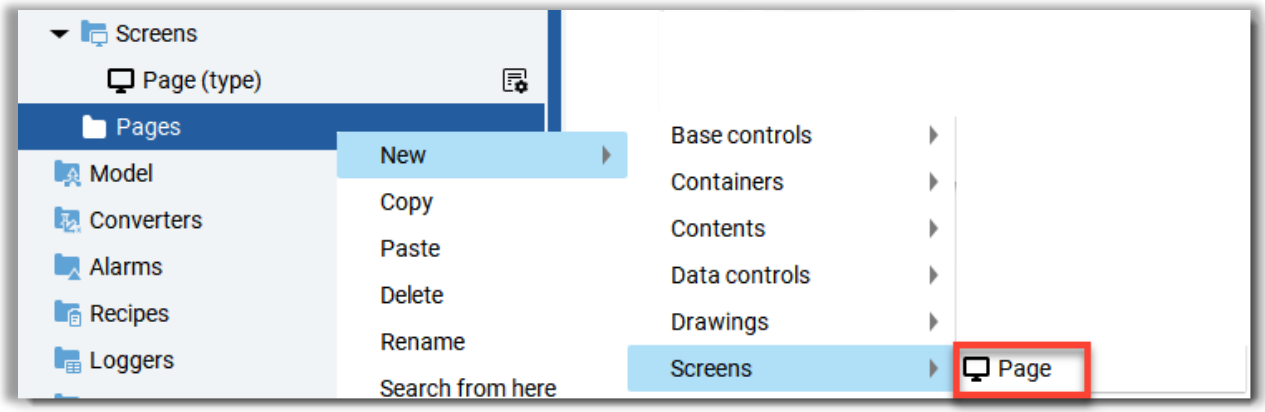
2. Hover over **Screen1 (type)**, select the edit icon , and enter "Page" as the new name.
3. In properties on the right, set **Left margin**, and **Top margin** to "10".

4. In the **Project view**, right-click **UI** and select **New > Folder**.

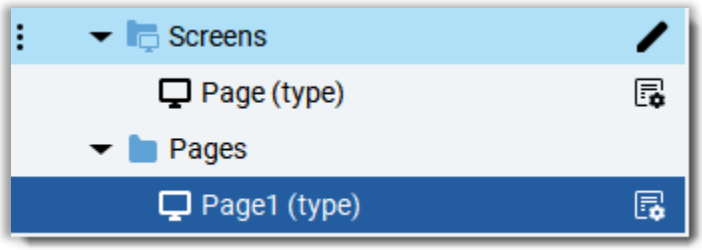


5. Hover over **Folder1**, select the edit icon , and enter "Pages" as the new name.

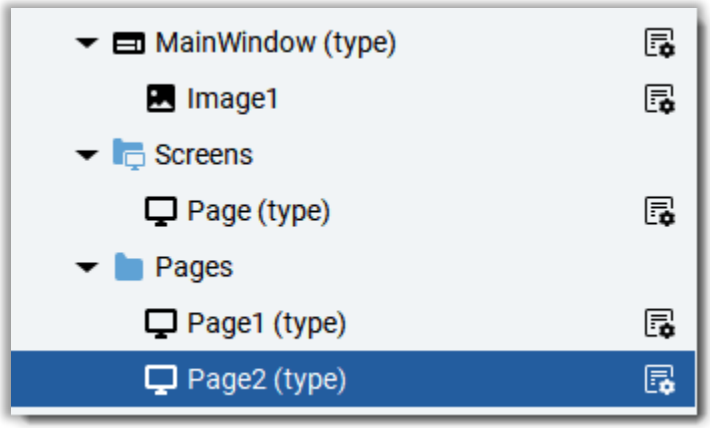
6. Right-click **Pages** and select **New > Screens > Page**.



Page1 (type) appears in the **Pages** folder.



7. Now, with **Page1 (type)** under **Pages** still highlighted, press **CTRL+C** to copy the object. Select the **Pages** folder, then press **CTRL+V** to create **Page2.** Each Page# (type) inherits its properties from **Page (type).**



Note: To properly instantiate child objects from a parent object type, only use the copy and paste method when the child type has not been modified. Otherwise, all specific changes to the particular child instance will be copied and it will not reflect the original parent type any longer. In cases where only the original type is meant to be instantiated, use the method described in step #6 above.

INSTANCES OF PARENT OBJECT TYPES

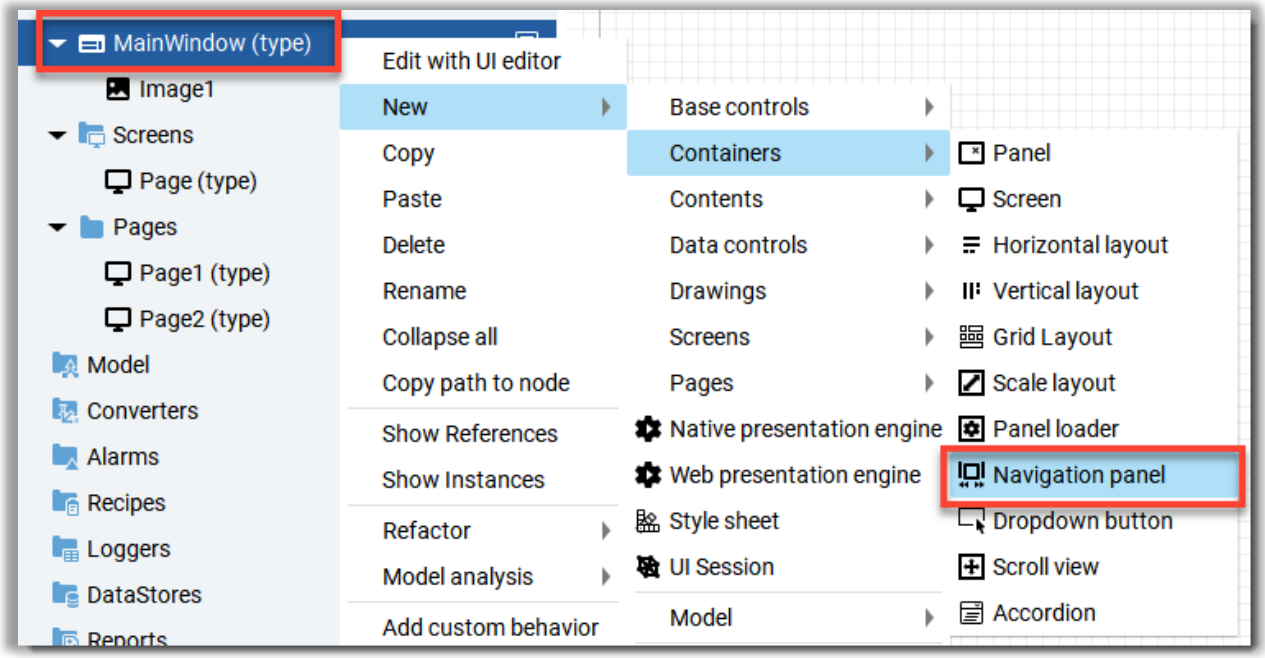
Create consistent user interfaces by inheriting properties from parent object types. For example, if you want to change the margins of both **Page1 (type)** and **Page2 (type),** edit the margin properties of **Page (type).** The modification will propagate to each **Page# (type).**

Create a Navigation Panel

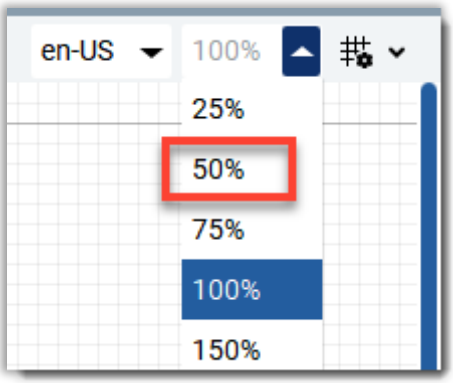
Configure a navigation panel to switch between different screens (pages) at runtime.

1. In the **Project view**, double-click **MainWindow (type)** to open it in the editor.
2. In the **Project view**, right-click **MainWindow (type)** and select **New > Containers > Navigation panel**.

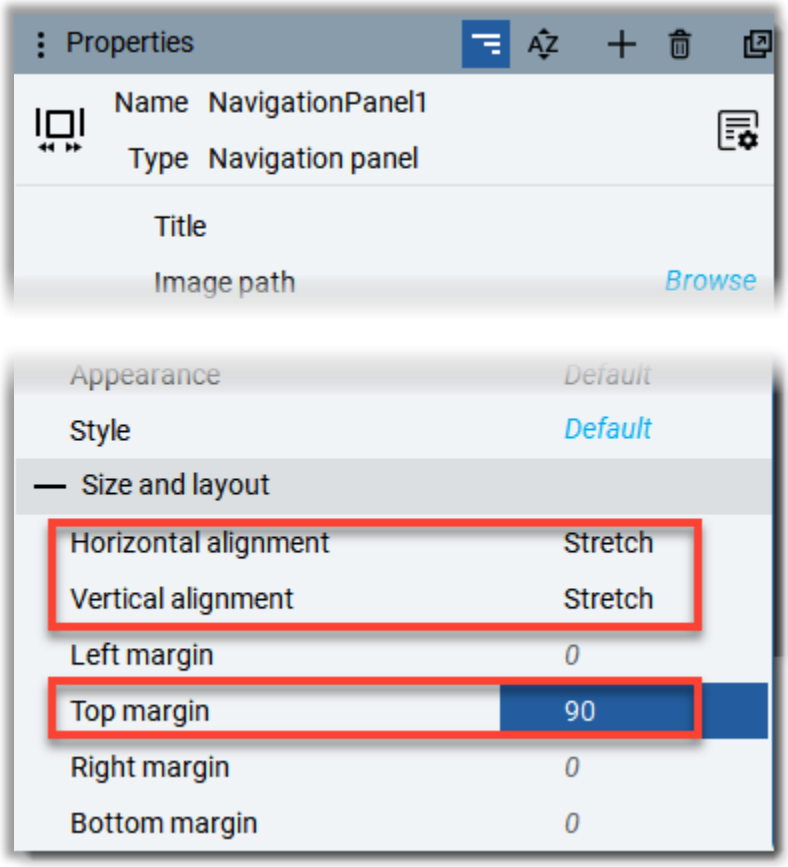
   Note: To learn more about Containers, see Appendix C – Container Objects



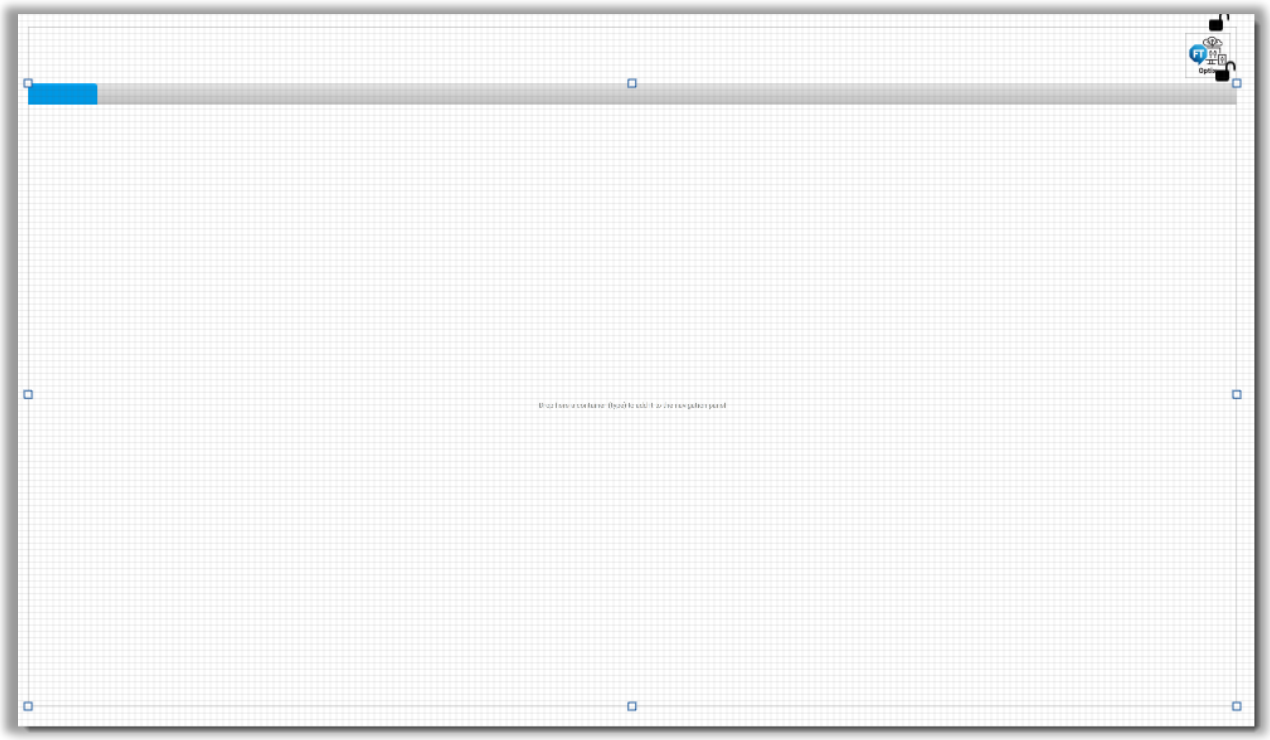
3. To view the full **MainWindow**, you can change the zoom level to 50% or use the scroll bars.



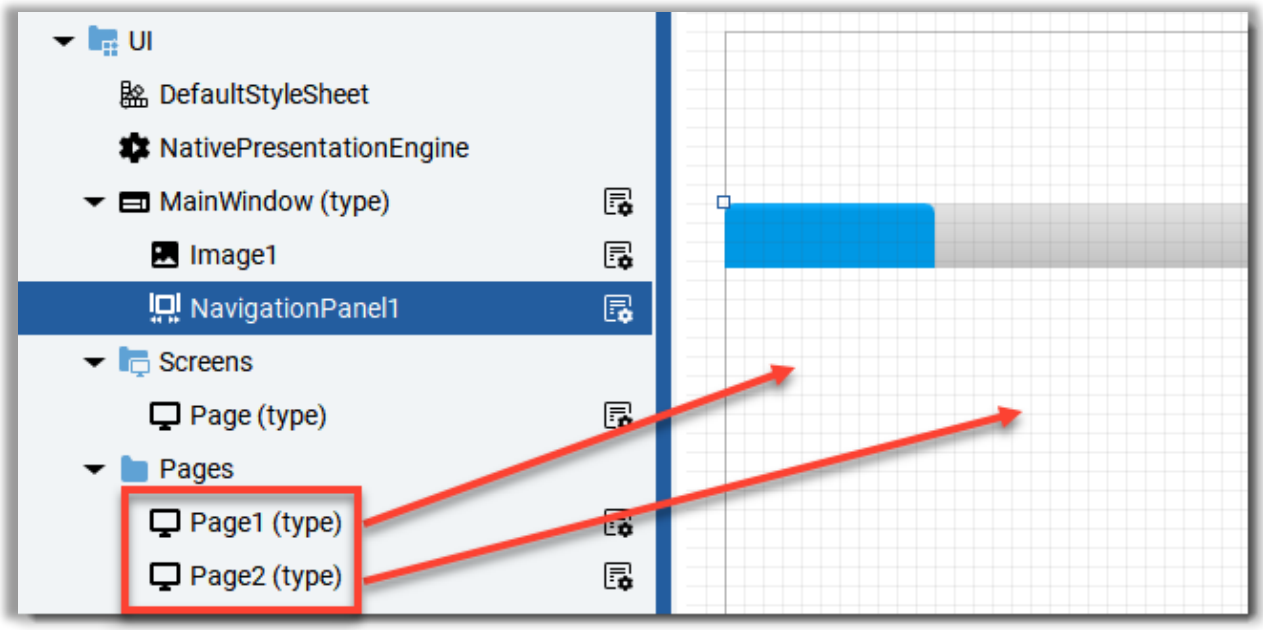
4. From the **Properties** pane, set the **Horizontal** and **Vertical alignment** to **Stretch**. Currently, the Navigation panel covers the image. Change the **Top Margin** to **90**.



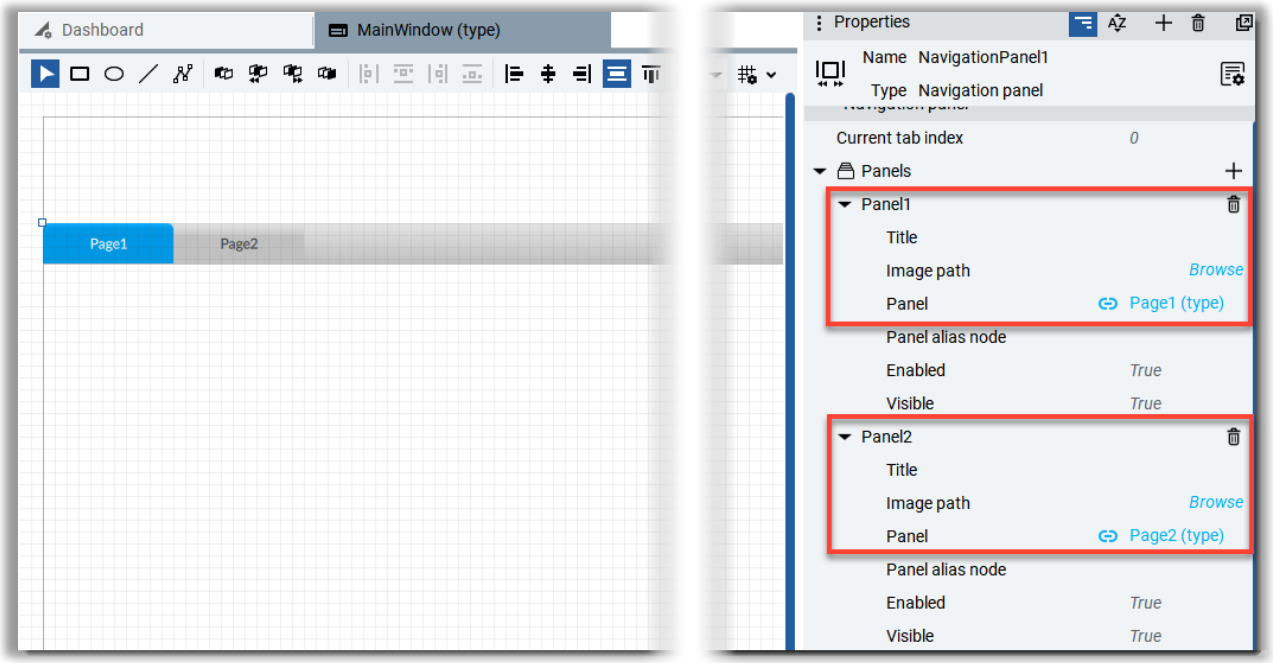
5. The **Navigation panel** now sits below the image but still stretches to fill the remainder of the **MainWindow**.



6. Drag **Page1 (type)** and then **Page2 (type)** to the **Navigation panel**.



You can change the properties of Page1 and Page2 such as the **Title** or add an image.



7. **Save** your project.

Now that you have created pages and a way to navigate at runtime, let's add some objects to the pages!

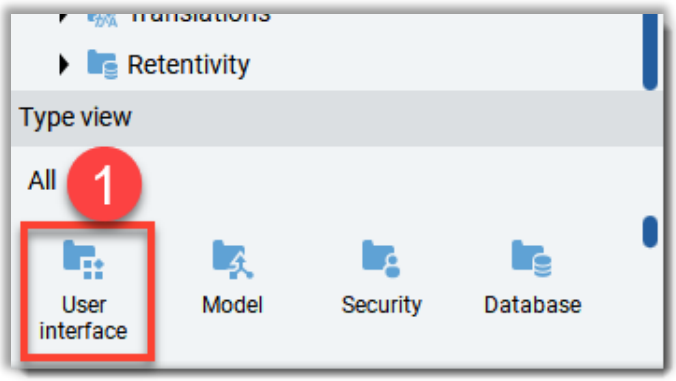
Add Objects and Variables

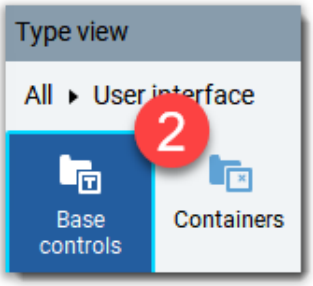
Add a Switch, Spin box, and LED objects to Page1 of your application.

1. In the **Project view**, double-click **Page1 (type)** to open it in the editor.
2. Rename **Page1** to "MyDashboard**"** by hovering over it and selecting the edit icon**.**

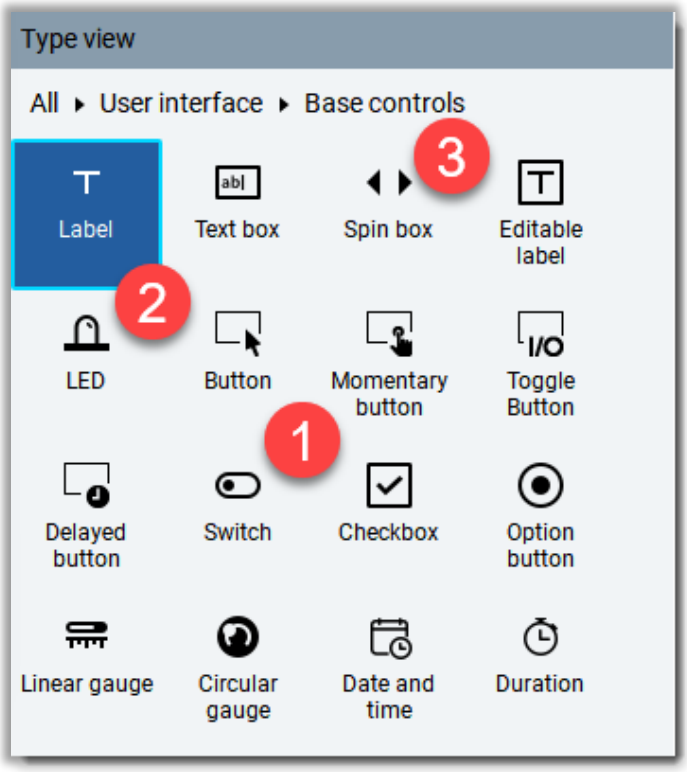
   Note: Now, you are beginning to modify a particular instance of the base Page (Type) parent object we created earlier. This instance will become a simple dashboard as you proceed through the following steps.

3. In **Type View,** double-click **User Interface**, then **Base controls** to view the content.

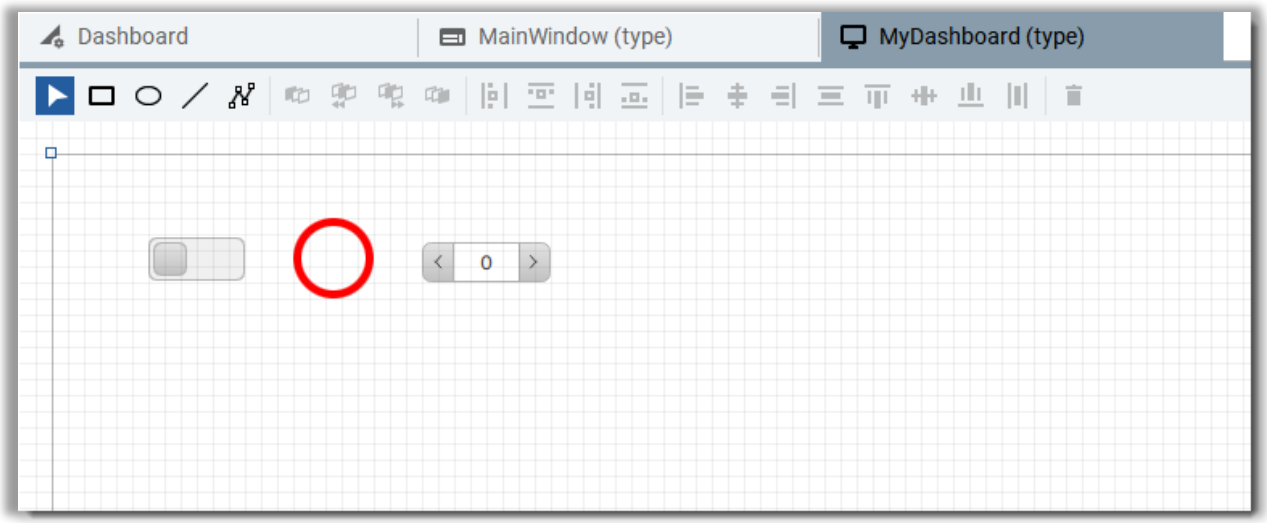




4. Drag a Switch, LED, and Spin Box to **MyDashboard**.



5. Arrange the objects according to your preferences.

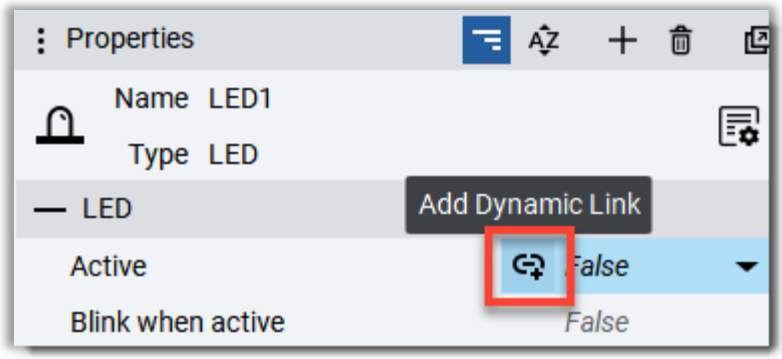


Associate the LED status with the switch

Control the color of the LED object by changing spin box values at runtime.

1. In the **Project view**, select **LED1**.

2. In **Properties**, next to the property value called **Active**, select the **Add Dynamic Link** icon



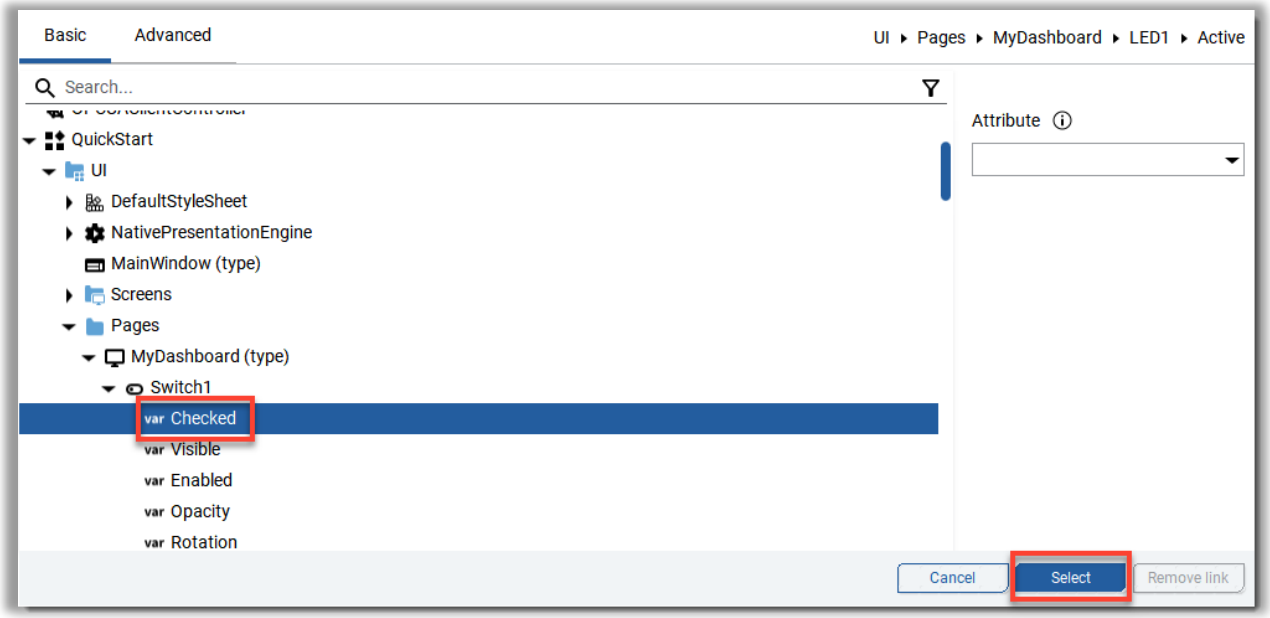
DYNAMIC LINKS

Use a dynamic link to set a variable value based on:

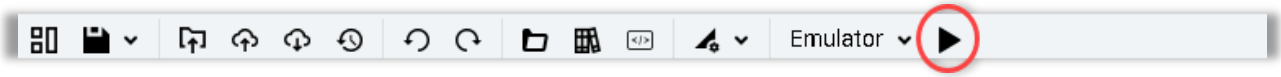
- The value of another variable.
- A property of an object.

For example, you can set the text property of a label object based on the value of a temperature variable.

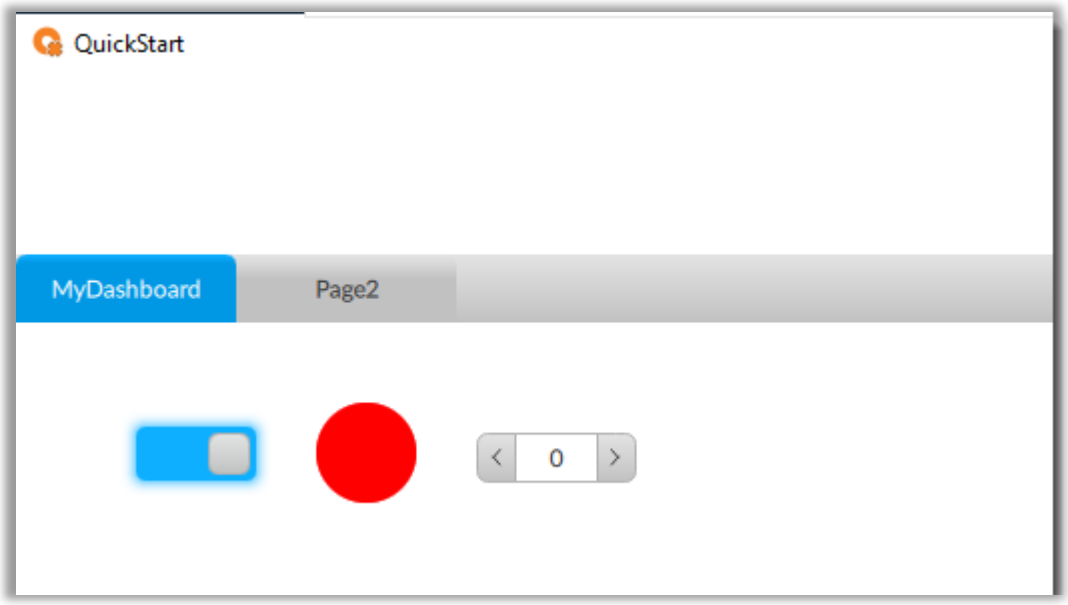
3. In the Dynamic Links browser, browse to **QuickStart > UI > Pages > MyDashboard (type) > Switch1 > Checked**.



4. Click **Select**.
5. **Save** your project.
6. From the toolbar, click the **Run Emulator** icon ▶.



7. Once the **Emulator** opens, toggle the switch and observe the LED becoming active (on).

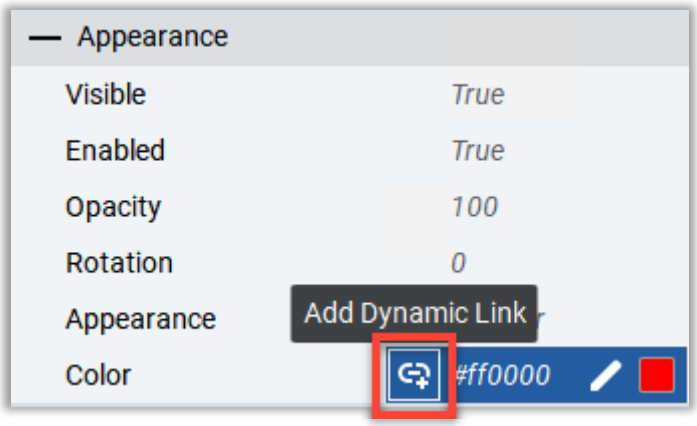


8. Close the emulator.

Change the color of the LED with a Spin box

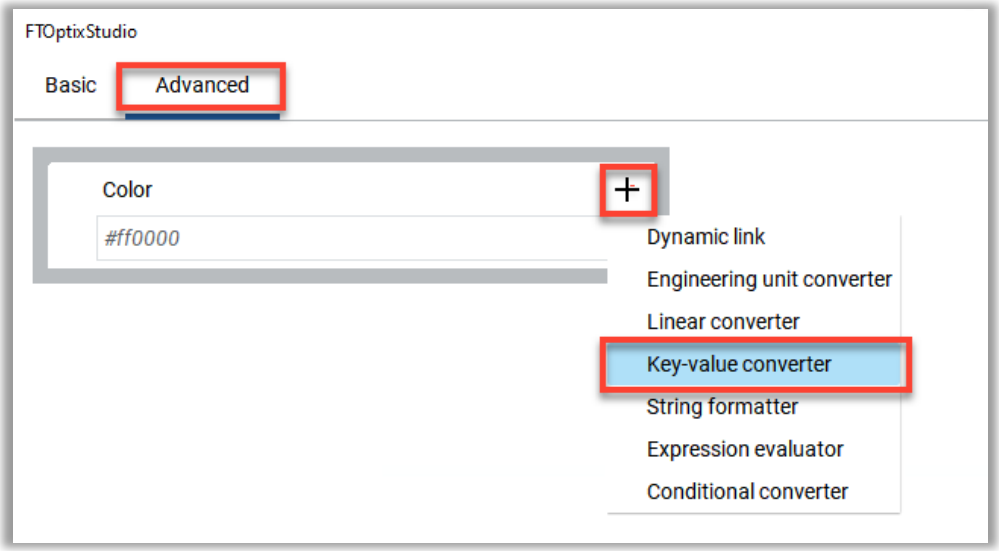
Control the color of the LED object by changing spin box values at runtime.

1. In the **Project view**, select LED1.
2. In **Properties**, next to the **Color** property, select the **Add Dynamic Link** icon .



3. Select the **Advanced** tab to get to the Complex Dynamic Link editor.

4. In the Complex Dynamic Link Editor, select the **Add new** icon + and select **Key-value converter.**



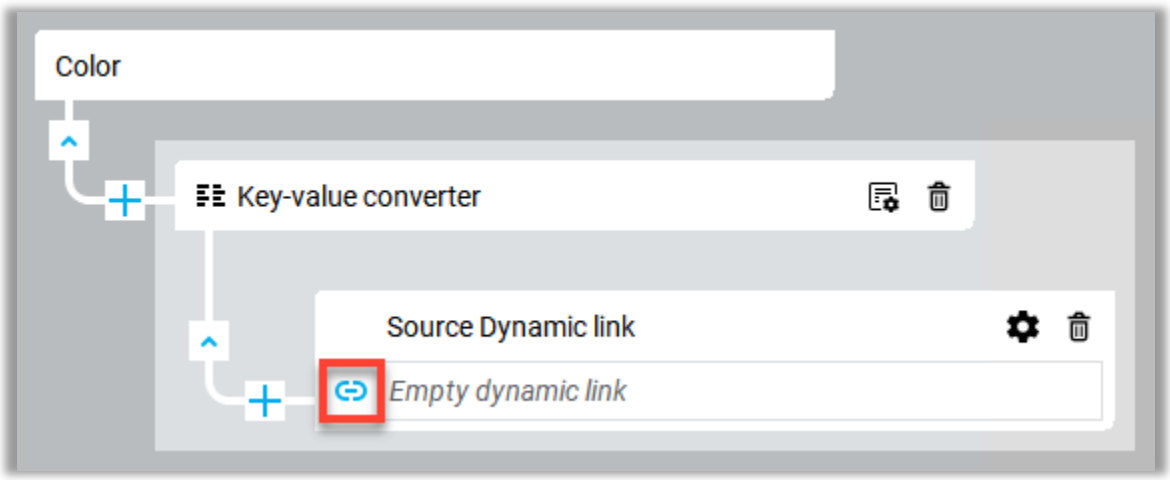
COMPLEX DYNAMIC LINKS & CONVERTERS

The complex dynamic link editor allows you to add advanced logic to the source node value. The complex dynamic link editor is displayed in the Advanced tab of the dynamic link browser.
Use converters to transform the source value and then assign the value to the parent node.
Converters include:

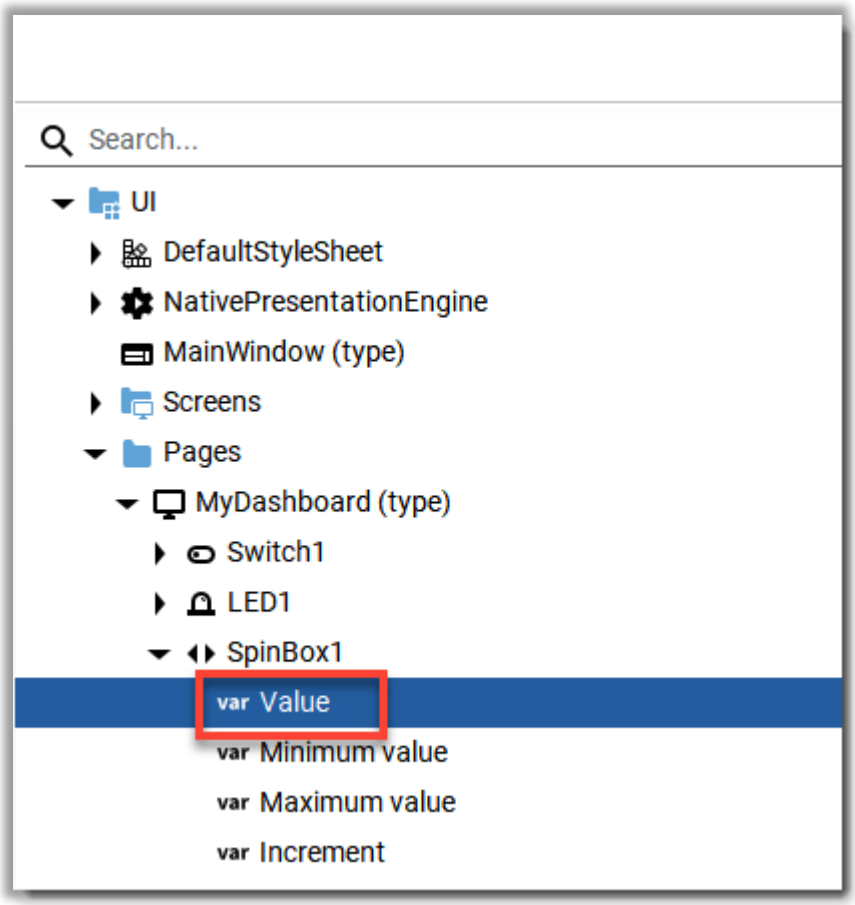
- **Engineering unit converter** – convert raw data values into scaled values.
- **Expression evaluator** - Calculate the result of an expression with integers, decimals, numeric variables, string variables.
- **Key-value converter** – Convert the source value based on a table of key-value pairs.
- **Linear converter** - Convert raw data values into scaled values based on the linear relationship.
- **String Formatter** – Modify the formatting of values.

Specify the source and converter logic to transform the value.

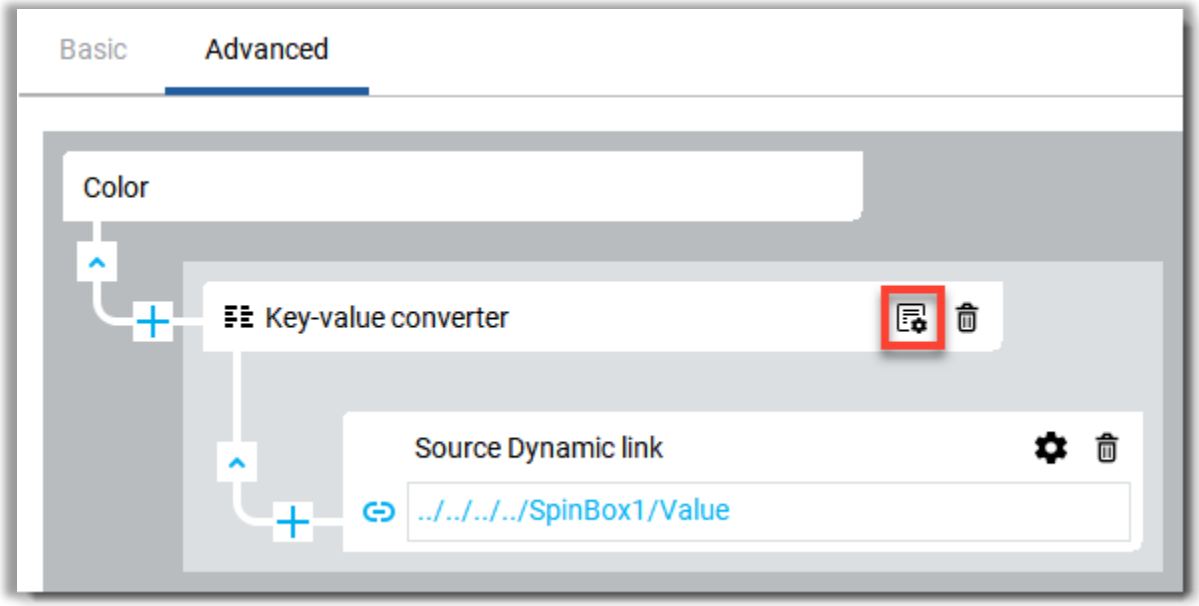
5. Select the **Change Dynamic Link** icon .



6. Browse and select **QuickStart > UI > Pages > MyDashboard (type) > SpinBox1 > Value**. Click **Select**.

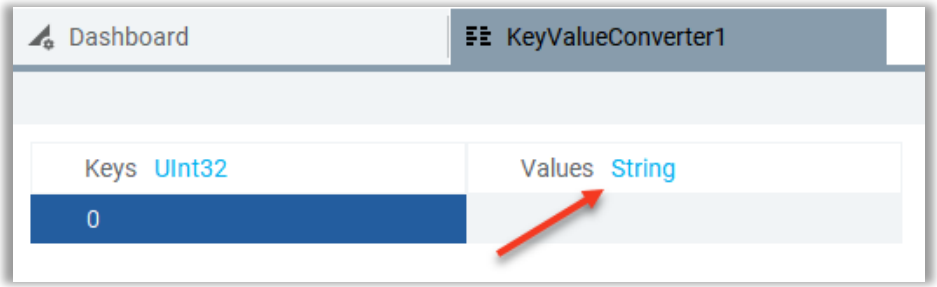


7. Next to **Key-value converter**, select the **Configure** icon .

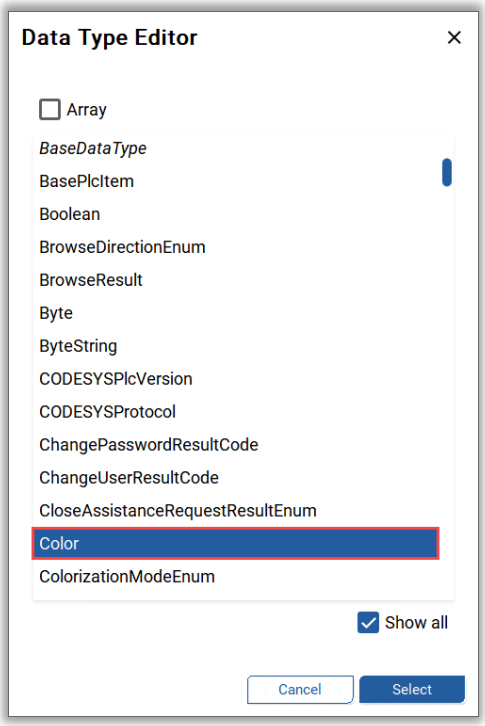


8. Now that the **KeyValueConverter1** editor has appeared on the main screen, close the Complex Dynamic Link editor by selecting **Close**.

9. In the central pane of the editor, the Value Map editor is displayed. Next to **Values** (header of the second column), click on **String** to open the **Data Type Editor**.

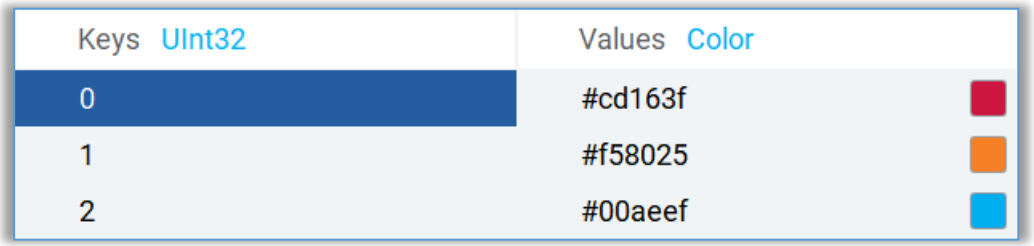


10. Select the **Show all** checkbox and change the data type to **Color**.



11. Choose **Select**.

12. Add two rows by selecting the **Add new** icon + two times and set the values as follows:

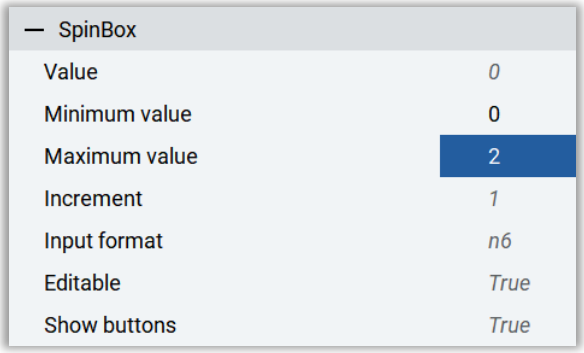


13. Close the **KeyValueConverter1** tab.

14. In the **Project view**, select **SpinBox1**.

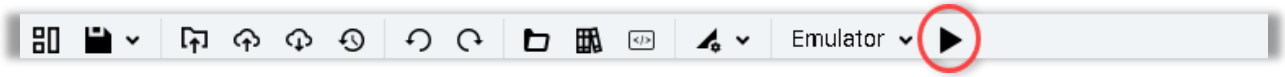
15. In Properties, set the **Minimum value** to "0", and the **Maximum value** to "2".

    Note: Setting **Minimum** and **Maximum** values prevents users from providing values that are not handled by the Key-value converter at runtime.

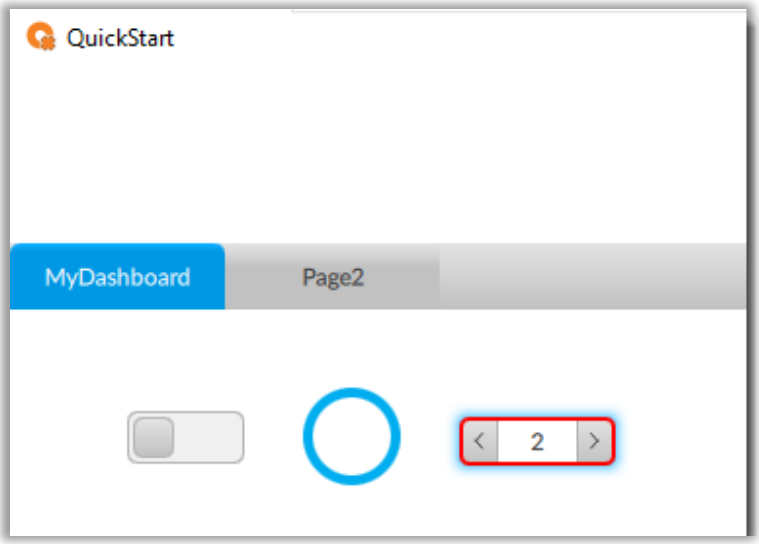


16. **Save** the project.

17. From the toolbar, click the **Run Emulator** icon ▶.



18. At runtime, in the Emulator, change the LED color using the Spin box. Click past the "2" value and notice the red indicator that alerts the operator they have reached the maximum value.



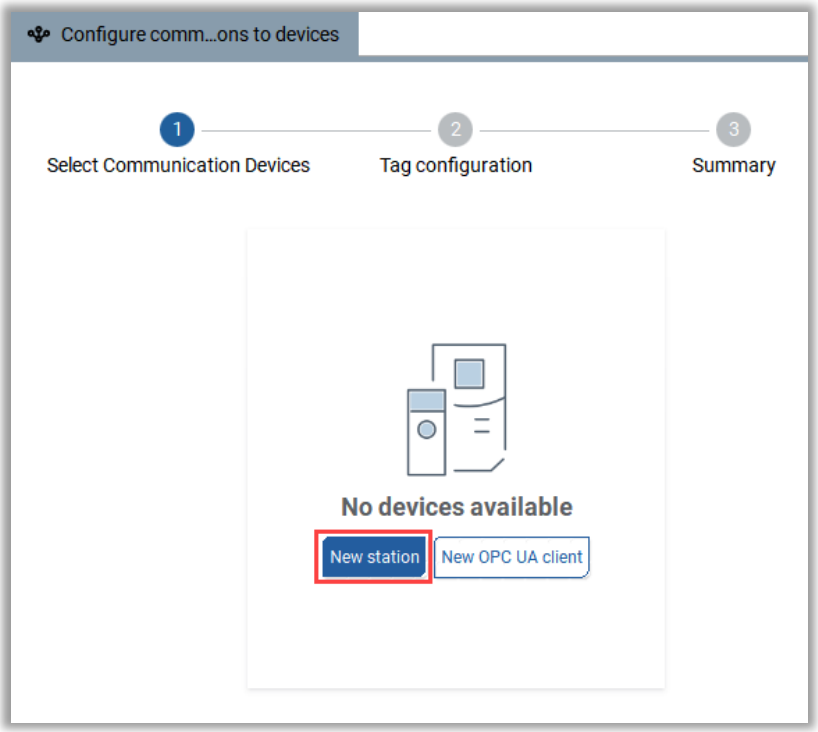
    Note: Turn the switch on to see the entire LED change colors with the Spin box and not just the border!

19. **Close** the Emulator.

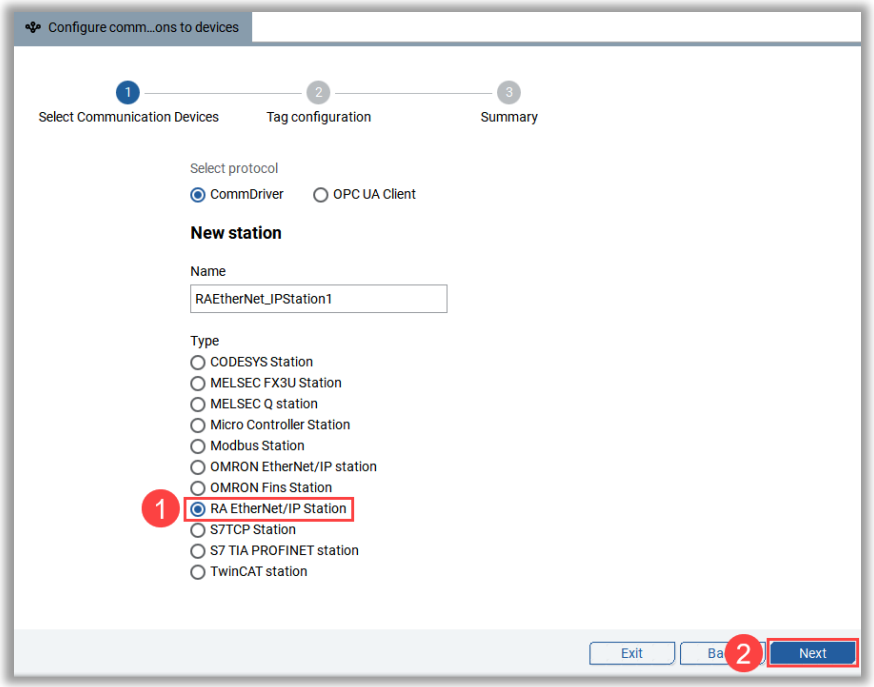
Configure Communications: Logix Station

1. Select the **Dashboard** tab or click on the **Dashboard** icon to access the wizards. For this section, you will select the **Configure communications to devices** icon.

2. Select **New station**.

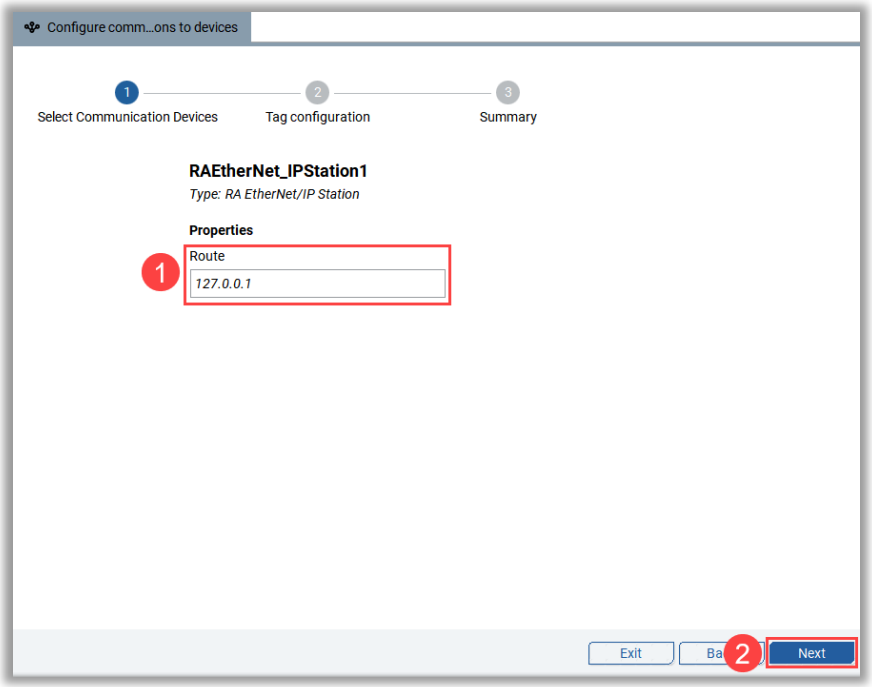


3. Now pick the radio button for **RA EtherNet/IP Station**. Leave the **Name** at the default value and click **Next**.



Note: Notice the list of native drivers that can be added to FactoryTalk Optix as well as OPC UA Client option.

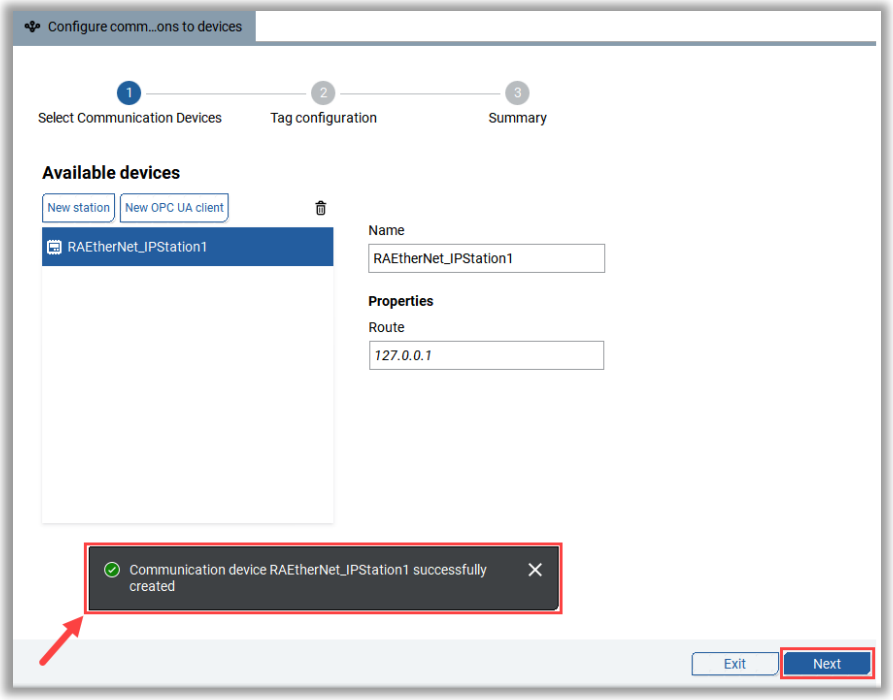
4. Enter "127.0.0.1" as the IP address in the **Route** field, and then click **Next**.



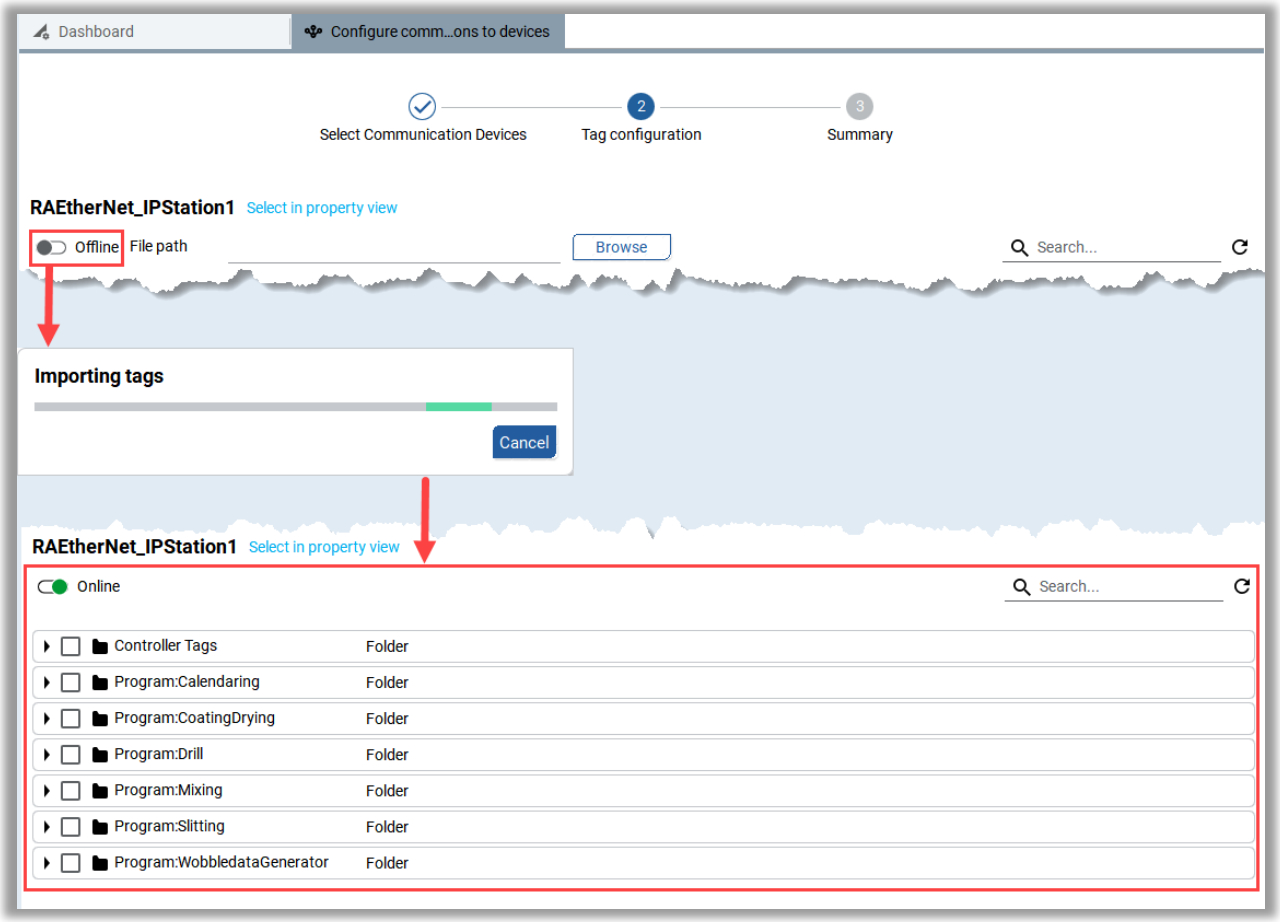
ROUTE SYNTAX

Using the IP address, the driver is looking at the controller in slot 0.
If you want to use a controller in a different slot you need the following syntax:
IP address\Backplane\slot. For example:
192.168.1.20\Backplane\2 would be the controller in slot 2

5. Once the RA Ethernet/IP station has been successfully created, choose **Next** to continue to **Tag configuration**.

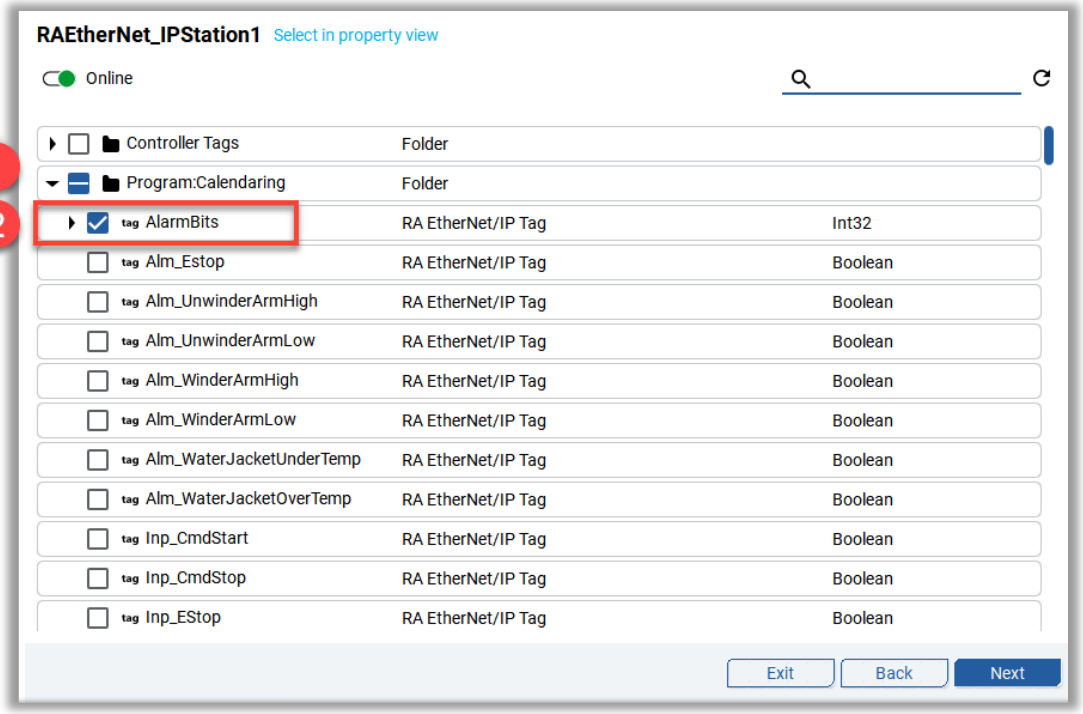


6. Toggle the switch next to **Offline** to execute an **Online** tag import. You should see *Importing tags* in the lower right-hand corner of the screen.

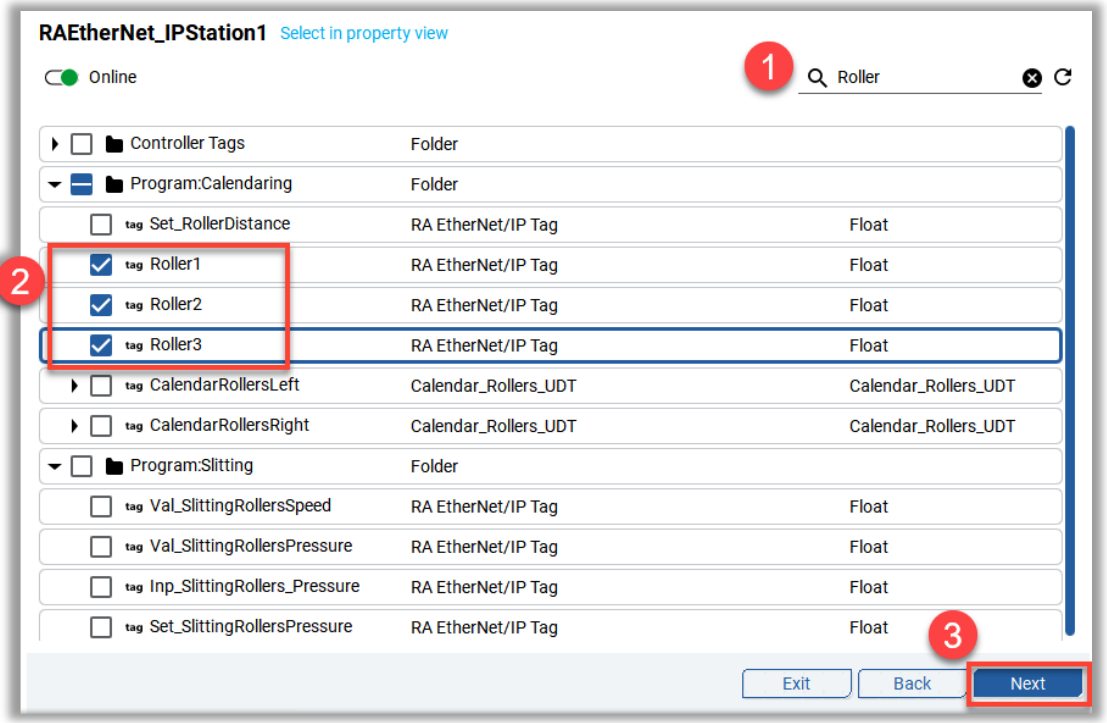


Note: If you don't have a controller, you can develop in Offline mode with an L5K, L5X or ACD file.

7. Now, expand **Program:Calendaring** and check the box next to **AlarmBits**.

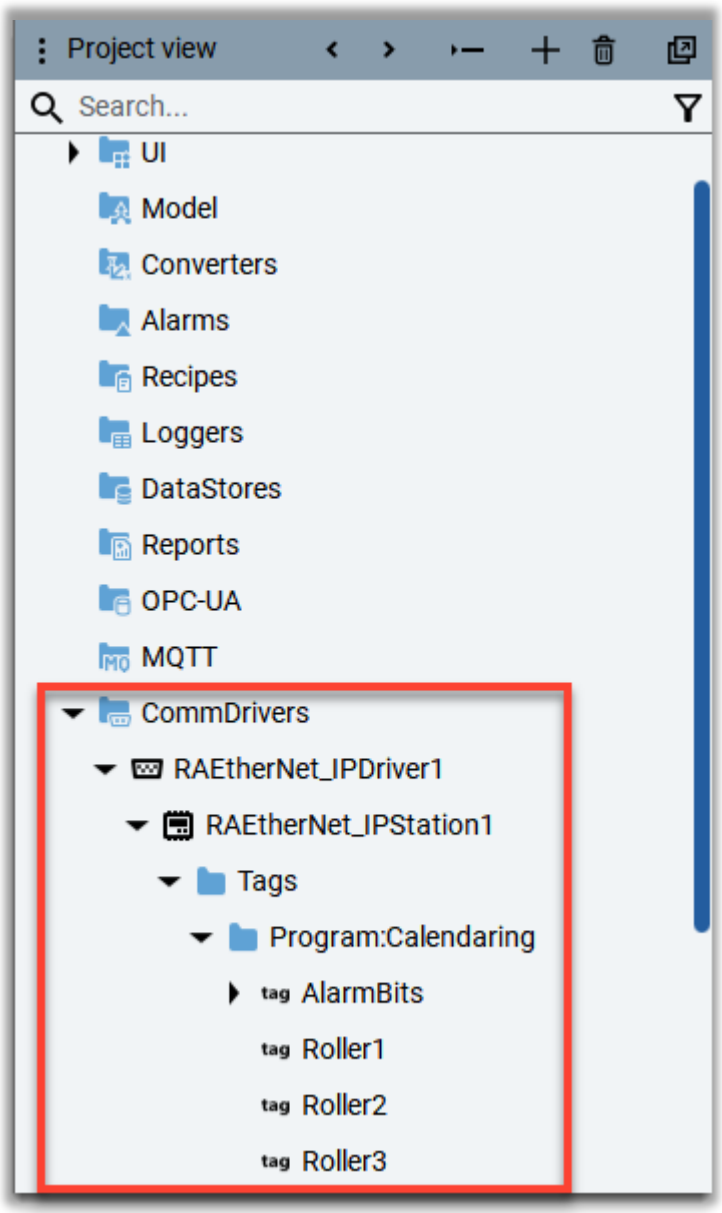


8. Enter "Roller" in the search box and check **Roller1, Roller2** & **Roller3**. Select **Next** to add the tags.



9. After seeing the Summary tab, click on **Exit** to close the communications wizard.

10. Now, in the **Project view** under **QuickStart > CommDrivers > RAEtherNet_IPDriver1 > RAEtherNet_IPStation1 > Tags > Program:Calendaring** you will find the communication driver and station created plus the tags you imported from the Logix controller into the project, **AlarmBits, Roller1, Roller2, Roller3**.



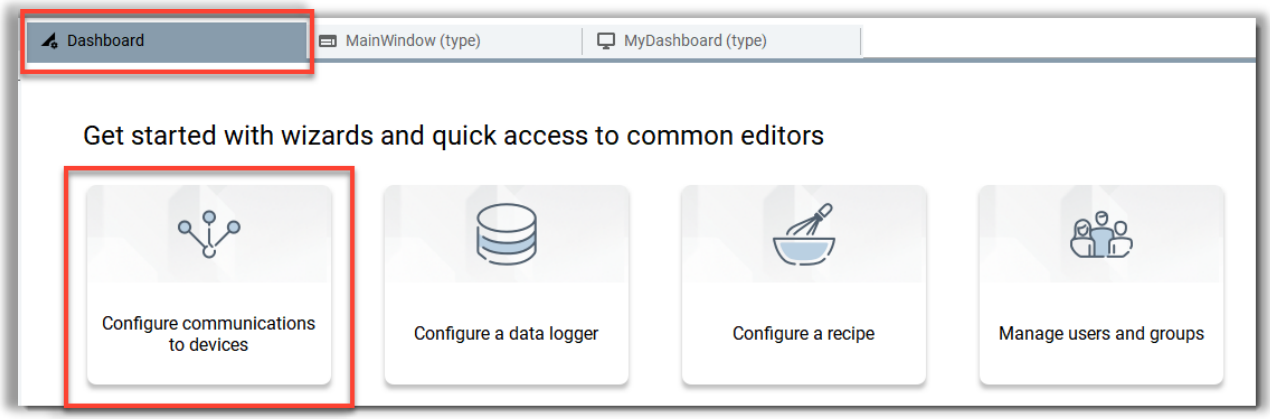
Configure Communications: OPC UA

Besides all the different communication stations/drivers that can be added to your project, FactoryTalk Optix also has full OPC UA connectivity capabilities. FactoryTalk Optix is compliant with the OPC UA (OPC Unified Architecture) standard and can communicate with any OPC UA client or server.

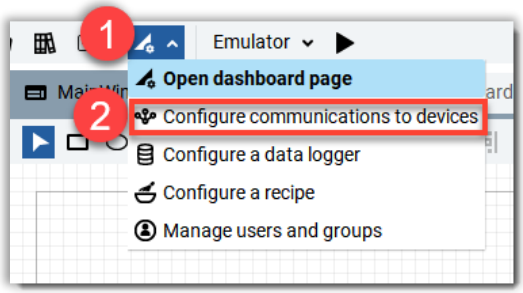
Note: Devices running properly configured FactoryTalk Optix applications can operate as an OPC UA client or server. For more information on OPC UA connectivity, see Appendix B.

OPC UA Client Configuration

1. Click on the **Dashboard** tab to access the wizards.

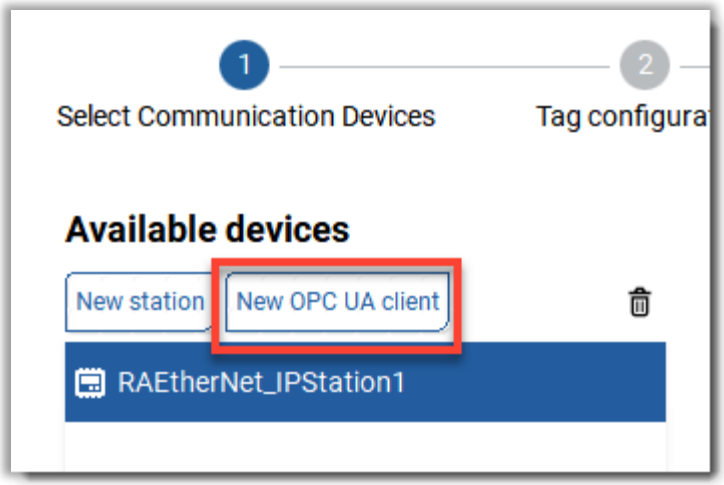


If the Dashboard tab isn't showing, you can access the wizards from the dashboard icon on the top menu.

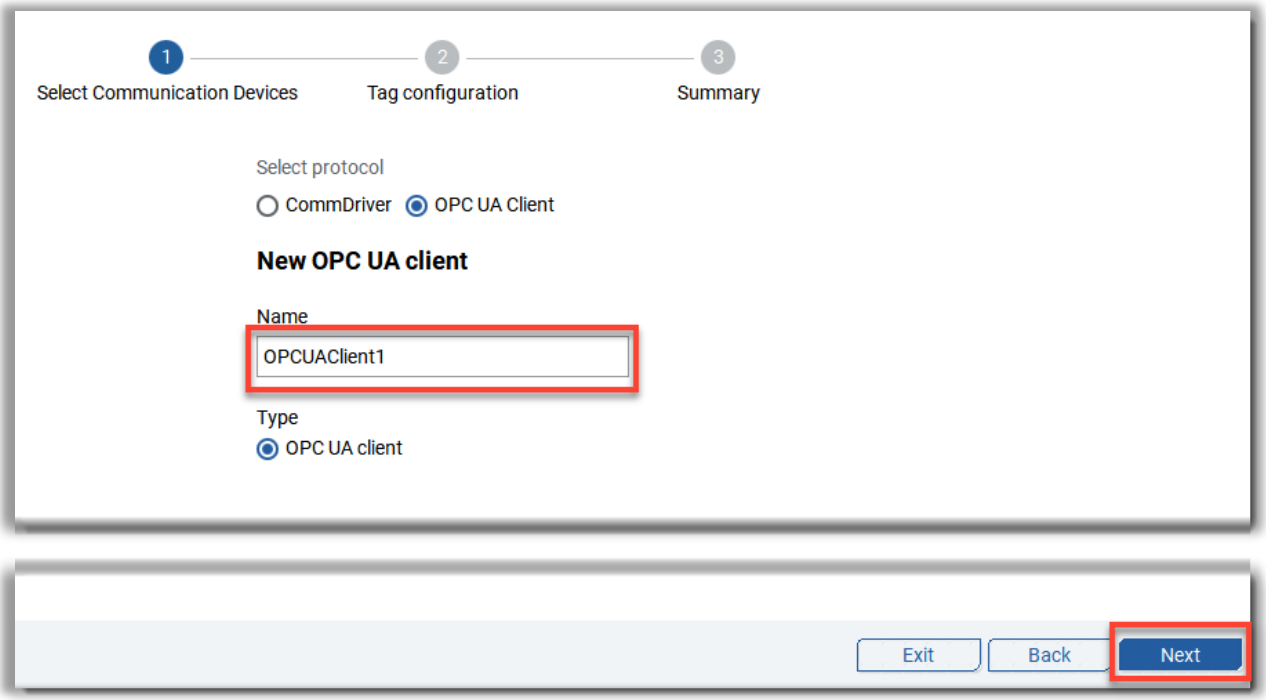


2. Click **Configure communications to devices**.

3. Under **Available devices** choose **New OPC UA client**.

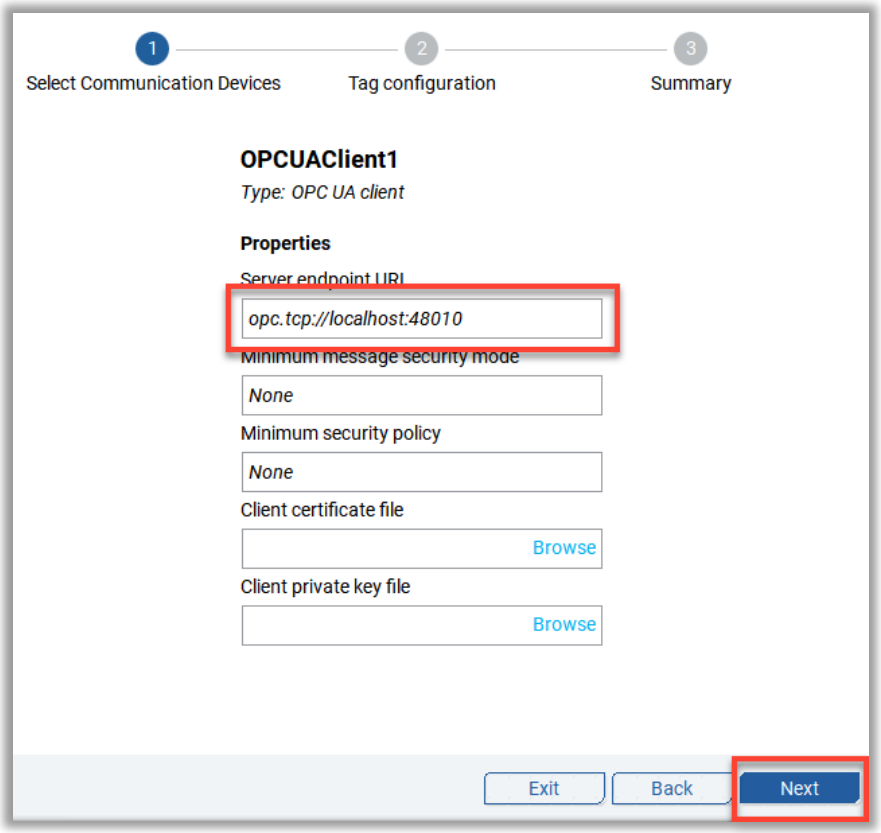


4. Keep the **Name** default as **OPCUAClient1** and click on **Next.**

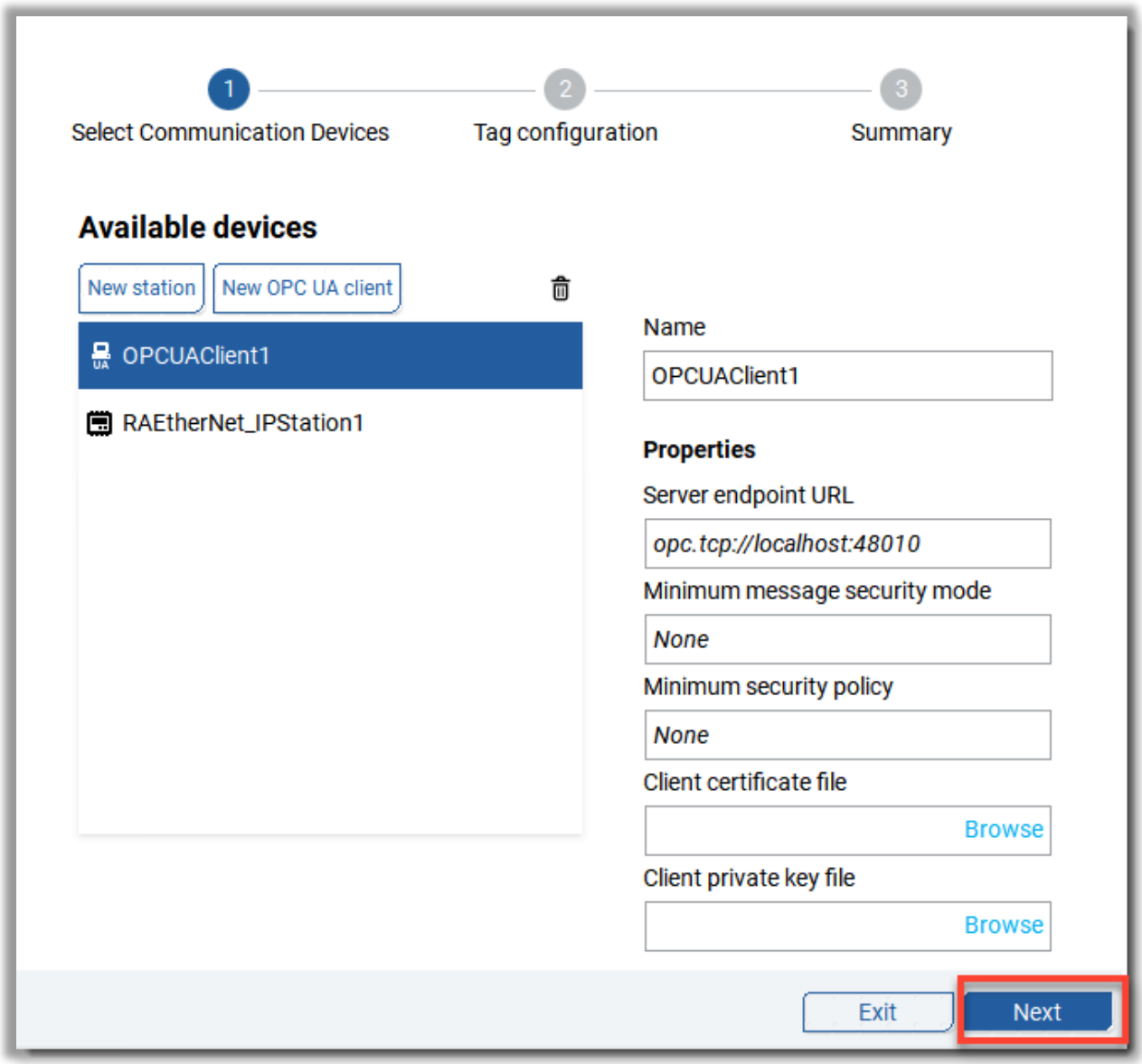


5. Under the **Server Endpoint URL** you need to fill in "opc.tcp://localhost:48010". Leave the rest of the settings at default, then click **Next**.

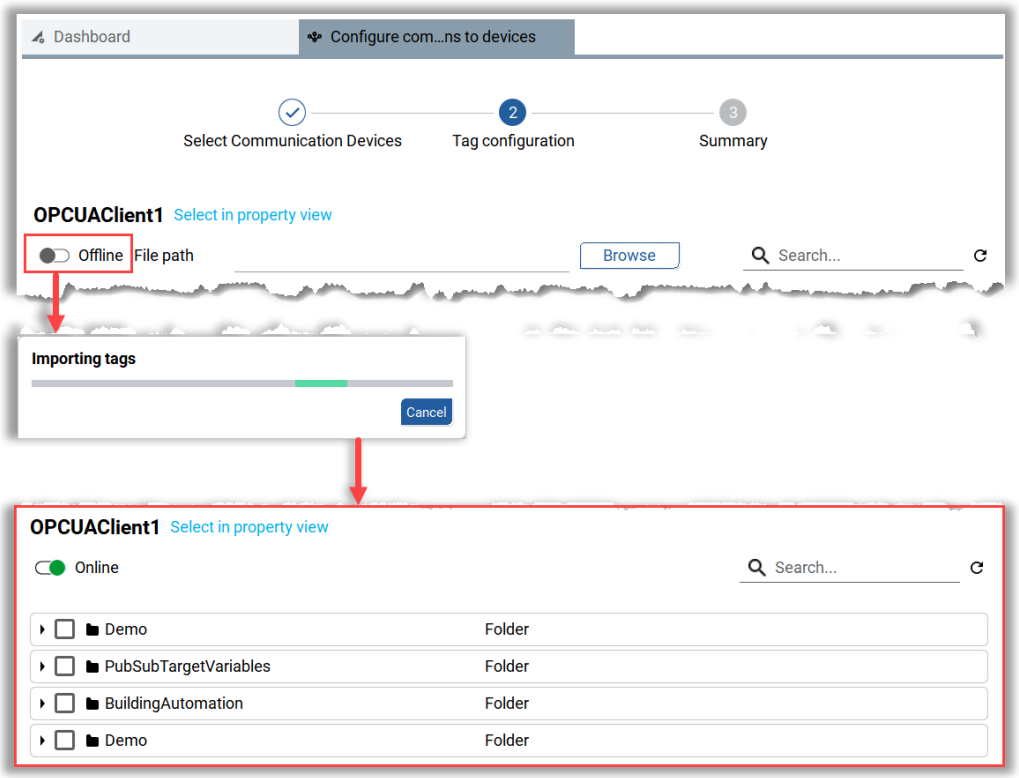
   Note: In the interest of time, we will not configure security to avoid needing to go through the additional steps required to manage certificates.



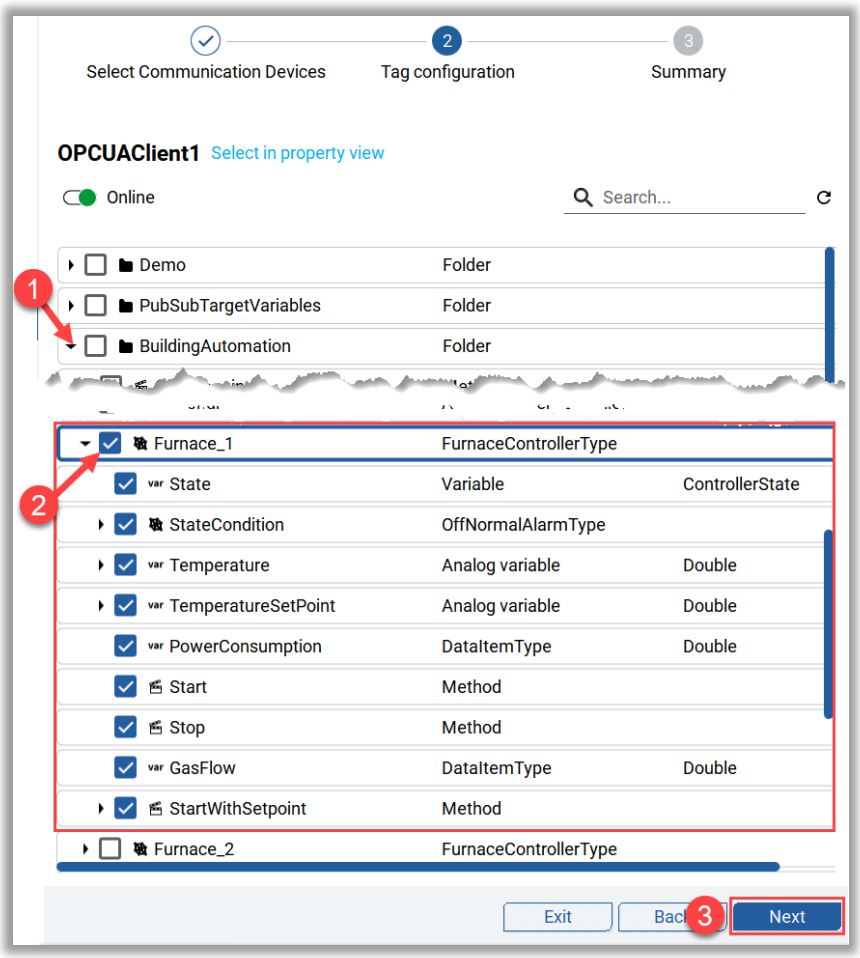
6. **OPCUAClient1** is now shown in **Available devices.** Click **Next** to complete the first step.



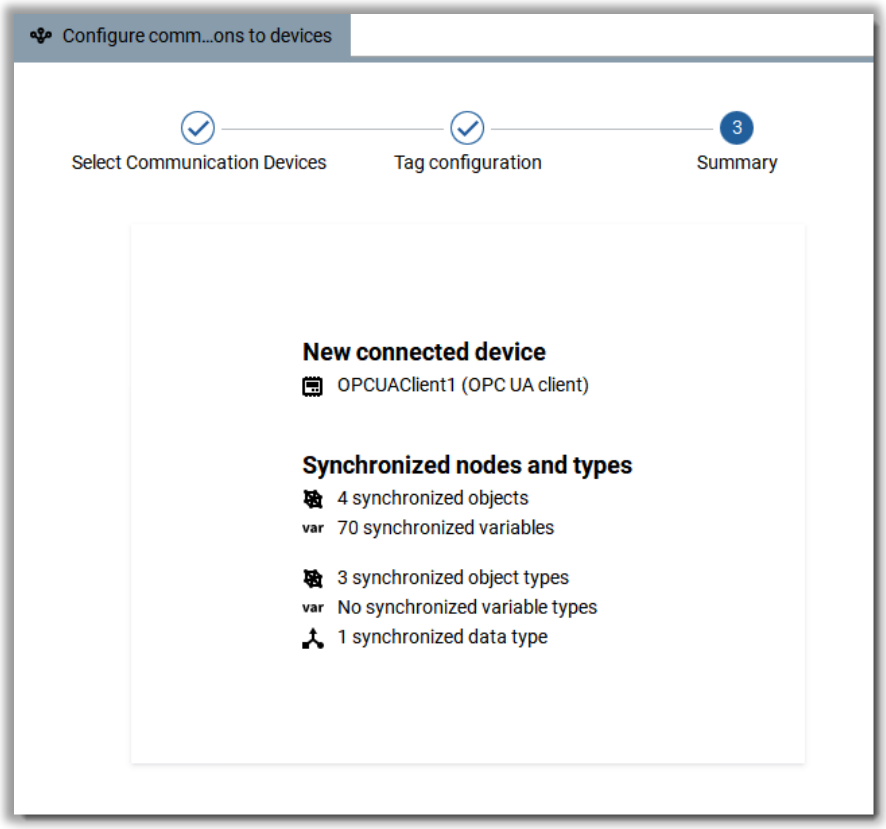
7. In the **Tag configuration** window, toggle the switch next to **Offline** to execute an Online tag import. Wait for tags to import from the OPC UA server.



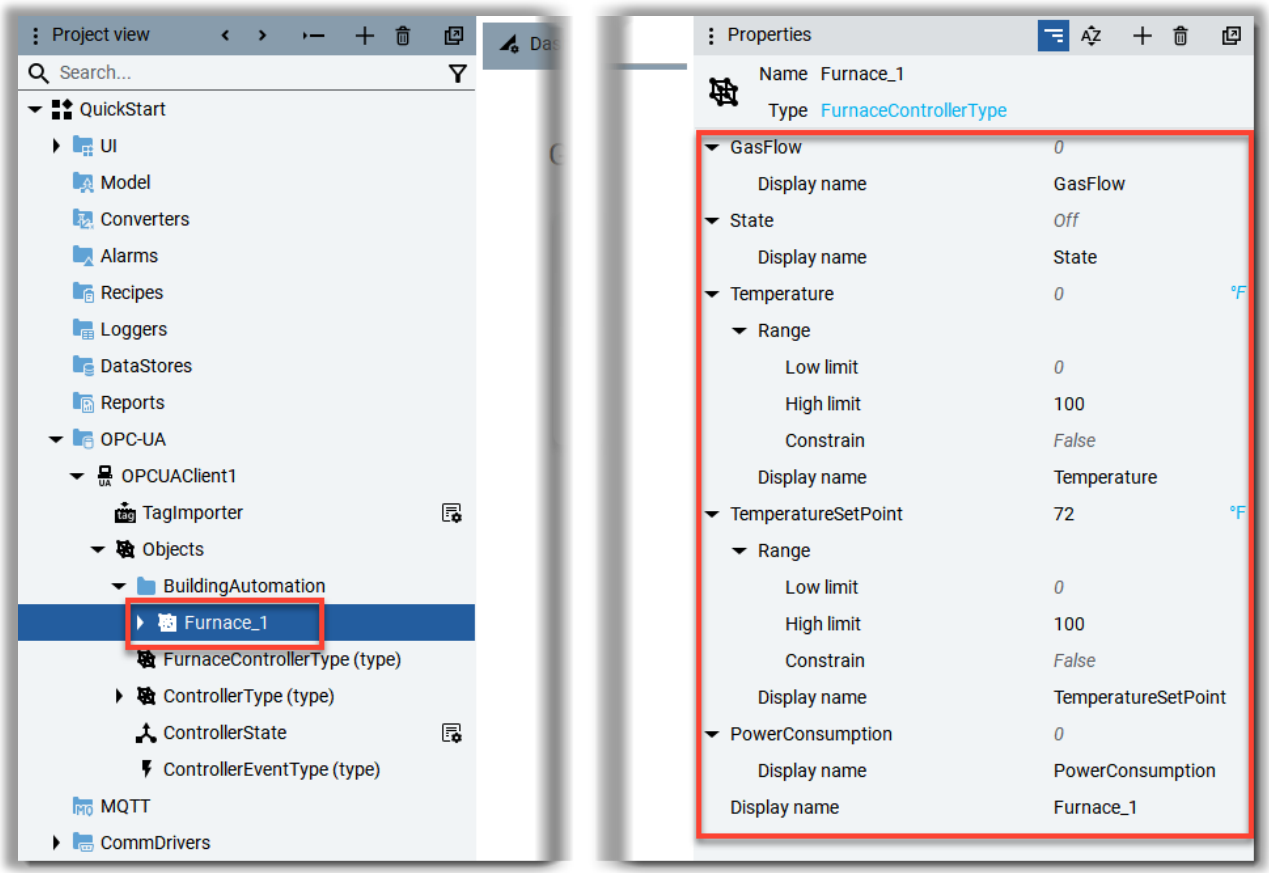
8. Now, expand **Building Automation** and check the box next to **Furnace_1** to import the OPC UA information model object and all its members. Then choose **Next** to add the object to the application.



9. Click on **Exit** to close the communications wizard.



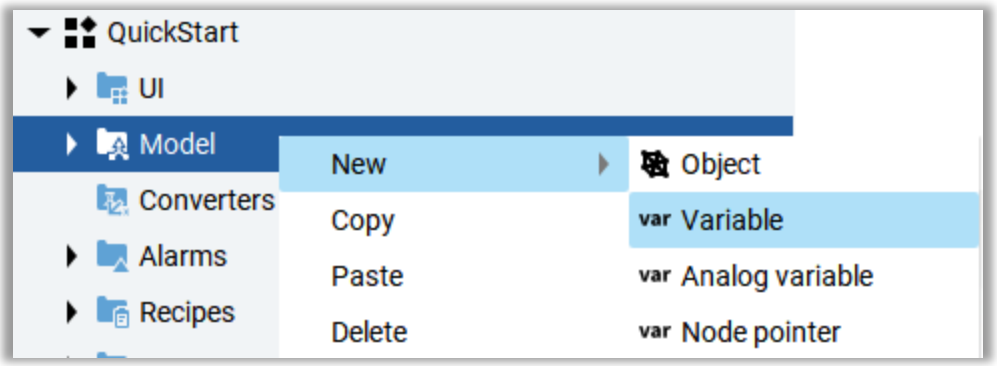
10. From the **Project view**, expand **QuickStart > OPC-UA > OPCUAClient1 > Objects > Building Automation** to see the OPC UA Client and the object you imported from the OPC UA server. Select **Furnace_1** and note the variables of the object.



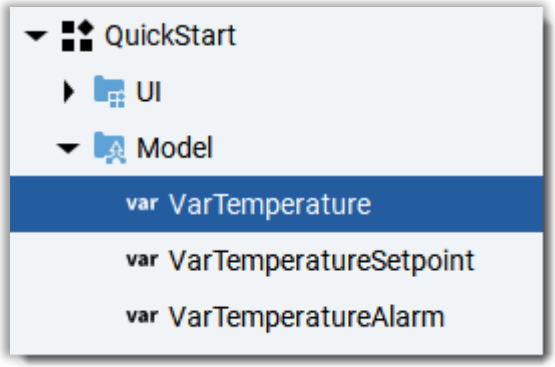
Creating Variables

The **Model** folder is used to create and hold objects and variables used in the project. The variables can be used in the project as they are, or reference tags that have been imported from controllers or OPC UA servers.

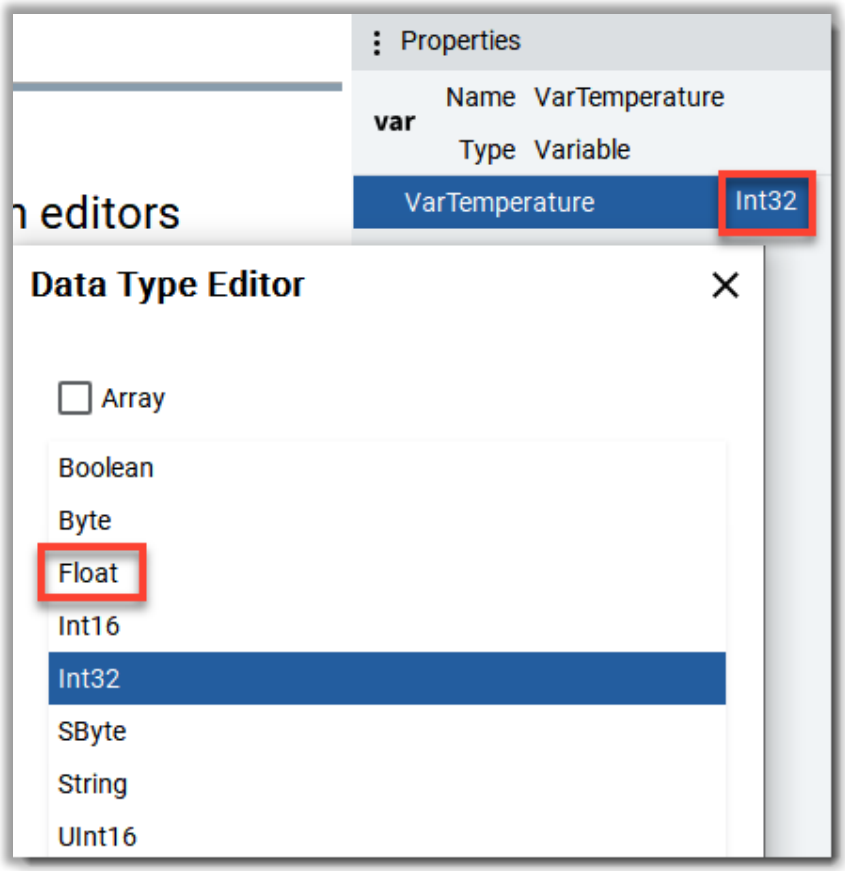
1. In the **Project view**, create a variable by right-clicking **Model** and selecting **New > Variable**.



2. Create three more variables by repeating the step above or by using copy & paste.
3. Rename the variables by hovering over the name and modifying them as shown:

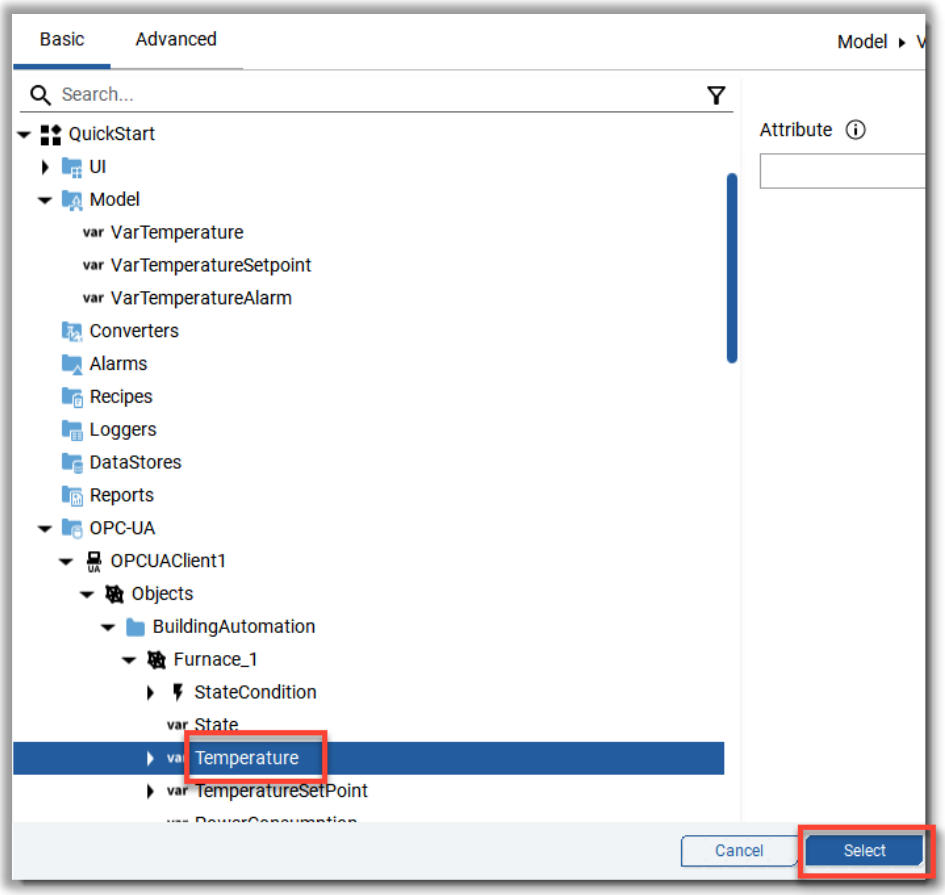


4. In the **Project view**, select **VarTemperature**, and in **Properties** select **Int32** and change to **Float**.

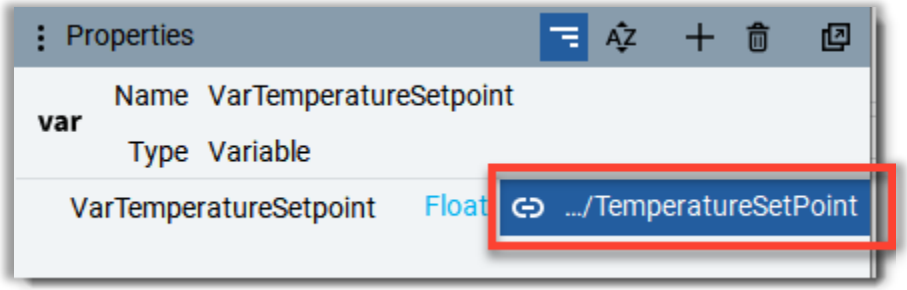


5. Repeat this for **VarTemperatureSetpoint** and **VarTemperatureAlarm**.
6. In the **Project view**, select **VarTemperature**.
7. In **Properties**, click on the **Add Dynamic Link** icon .

8. Browse and select **QuickStart > OPC-UA > OPCUAClient1 > Objects > BuildingAutomation > Furnace_1 > Temperature**.



9. Repeat the previous steps for **VarTemperatureSetpoint**. Browse **QuickStart > OPC-UA > OPCUAClient1 > Objects > BuildingAutomation > Furnace_1 > TemperatureSetPoint**.

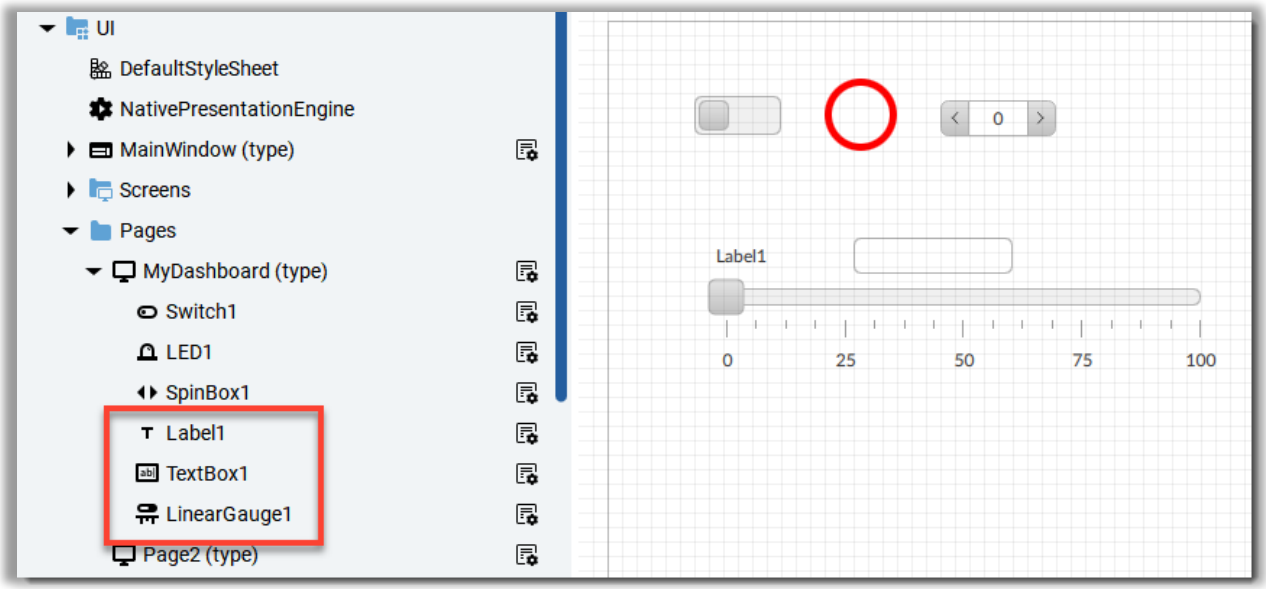


10. Save your project.

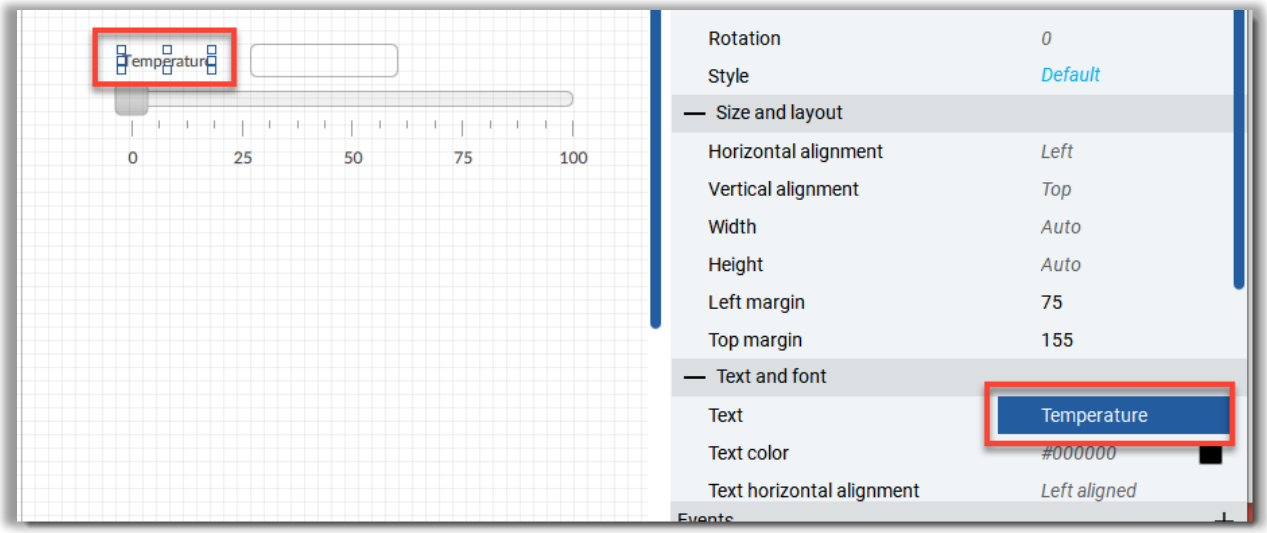
Now that you've created variables, you will associate them with an object to view and control the temperature.

Create Temperature Control objects

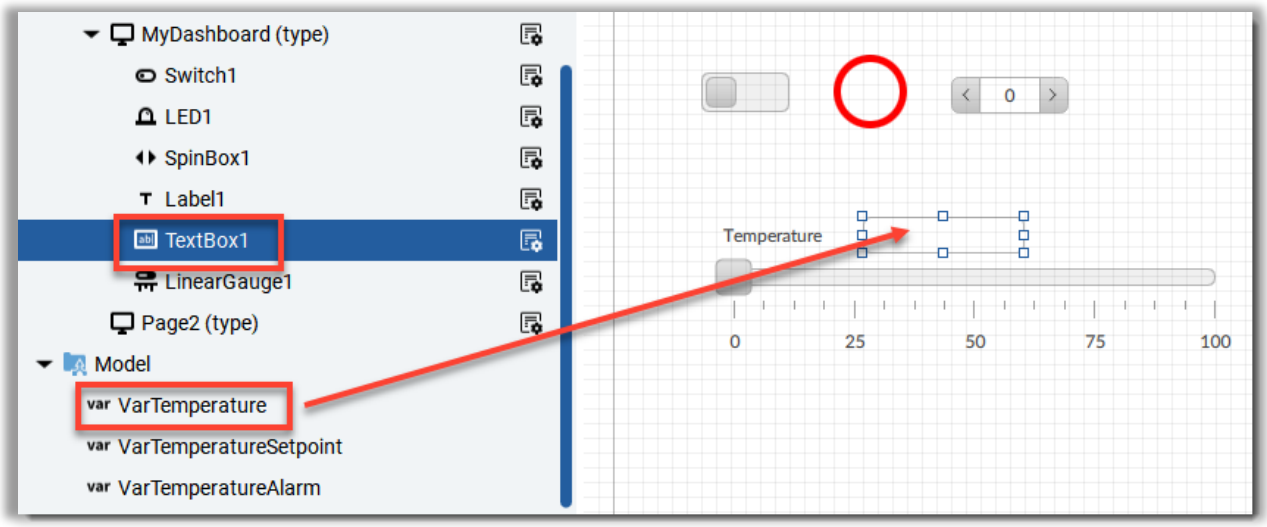
1. From the **Project view** expand **QuickStart > UI > Pages** and double-click **MyDashboard (type)**.
2. Right-click on **MyDashboard (type)** and select **New > Base controls > Label**.
3. Following the step above, add a **Text box** and a **Linear gauge**.
4. Move the three new objects so that they look something like the following:



5. Select **Label1** from the **Project view** or the editor and change the **Text** property to "Temperature".

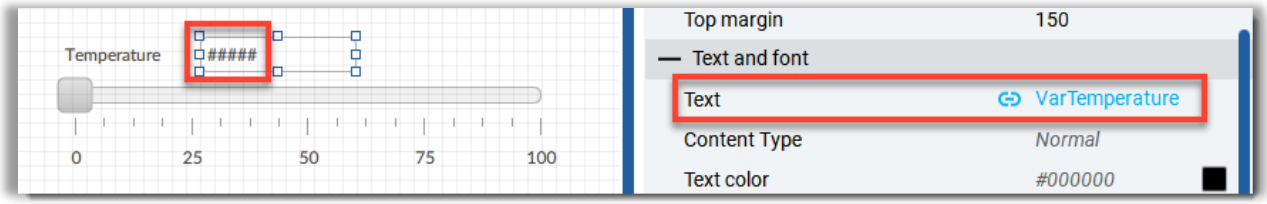


6. Select **TextBox1**. Drag and drop **VarTemperature** to TextBox1.

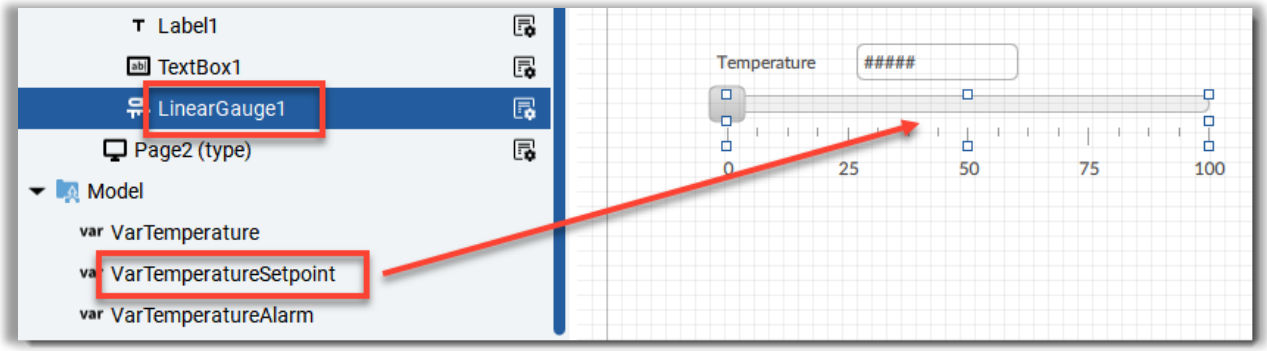


TIP

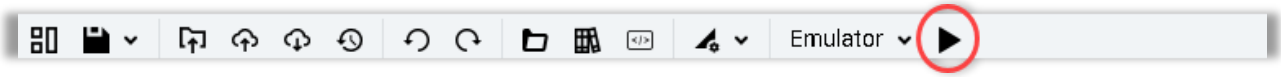
Objects often have a primary property to allow the drag and drop method to populate the primary property of the object – in the case of a textbox, the primary property is **Text.**



7. Use the same drag and drop method to associate **VarTemperatureSetpoint** with **LinearGauge1**.



8. Save the project.
9. Click the **Play** button beside **Emulator** to run the project.



10. Change the **TemperatureSetPoint** value by moving the **Linear gauge**, and watch the **Temperature** respond accordingly. The **Temperature** will gradually move toward the selected setpoint.

    Note: This is a very large furnace and it can take a long time for the temperature to reach the setpoint 😊.

11. Close the Emulator when you are done!

Alarming

Objectives

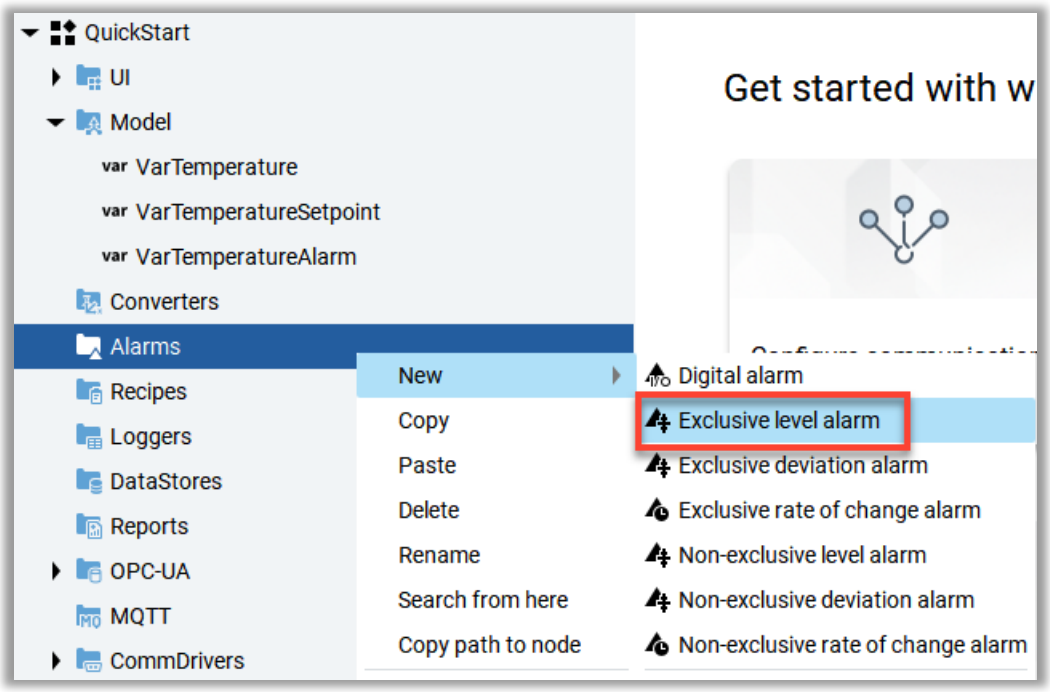
- Configure Optix Digital and Analog Alarms
- Enable Logix Alarms
- Add and configure Alarm Summary and Alarm Manager

Scenario

In this section of the lab, you will experience how easy it is to configure alarms in FactoryTalk Optix and how to subscribe to Logix Tag-based alarms. You will observe and interact with these alarms at runtime utilizing ready-to-use alarm objects from the extensive FactoryTalk Optix Template Libraries.

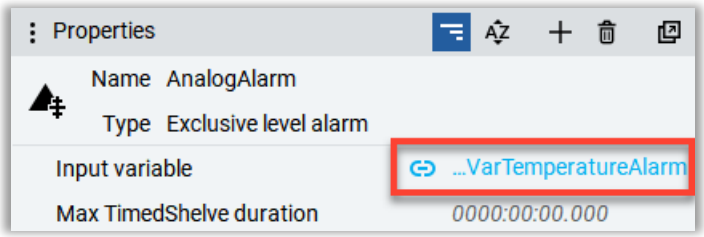
Configure Optix Alarms

11. In the **Project view**, right-click the **Alarms** folder and select **New > Exclusive level alarm**.



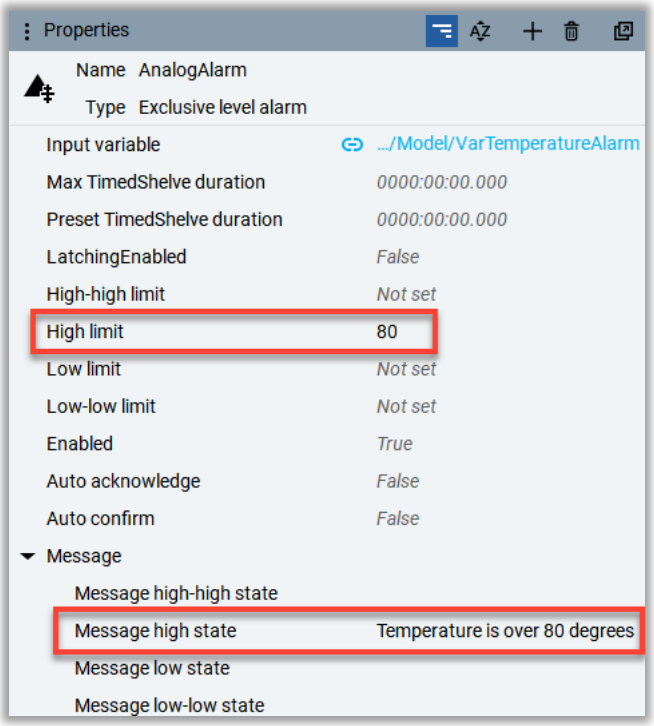
12. Rename **ExlusiveLevelAlarm1** to "AnalogAlarm" by clicking on the pencil symbol on the right of the object.

13. Expand the **Model** folder and drag and drop the **VarTemperatureAlarm** variable to the **Input variable** field in the **AnalogAlarm** object's **Properties** pane (make sure AnalogAlarm is selected in the **Project view**).



14. With the alarm's **Properties** pane still open, set the **High limit** to "80".

15. Under **Message**, in the **Message high-state** field, type the message: "Temperature is over 80 degrees".
Your configured alarm properties should look like this:

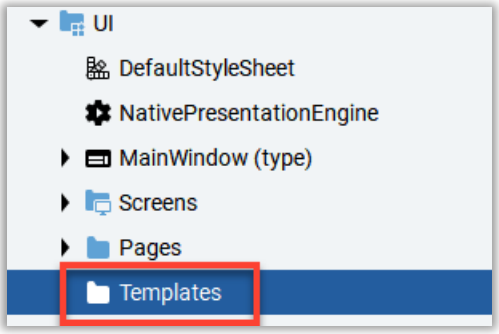


Adding Library objects

FactoryTalk Optix Studio has a set of libraries that make development fast and efficient. Libraries contain object definitions or templates that can be added to your projects. For example, libraries contain graphic elements such as values and tanks that can be used in your project. Library templates also exist for miscellaneous objects, pre-defined code called scripts, style sheets or themes, and widgets to interact with at runtime.

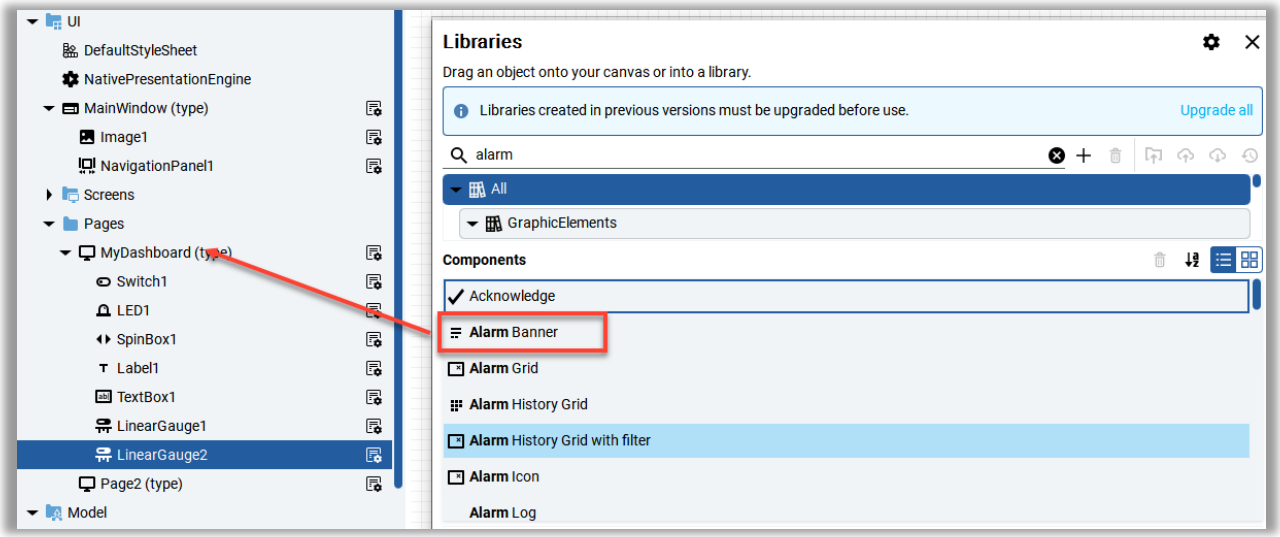
The steps below will show you how to create objects to trigger an Optix alarm and use Alarm objects from the library to view and manage the alarms.

1. Open **MyDashboard** under the **Pages** folder if it isn't already open.
2. In the same window move the gauge to place **LinearGauge2** below **LinearGauge1**.
3. Expand **Model** to view the variables.
4. Select **LinearGauge2** so its Properties are shown.
5. Drag **VarTemperatureAlarm** to **LinearGauge2**.
6. In the **Project view**, right-click **UI** and select **New > Folder**. Hover over **Folder1**, select the edit icon , and enter "Templates" as the new name.

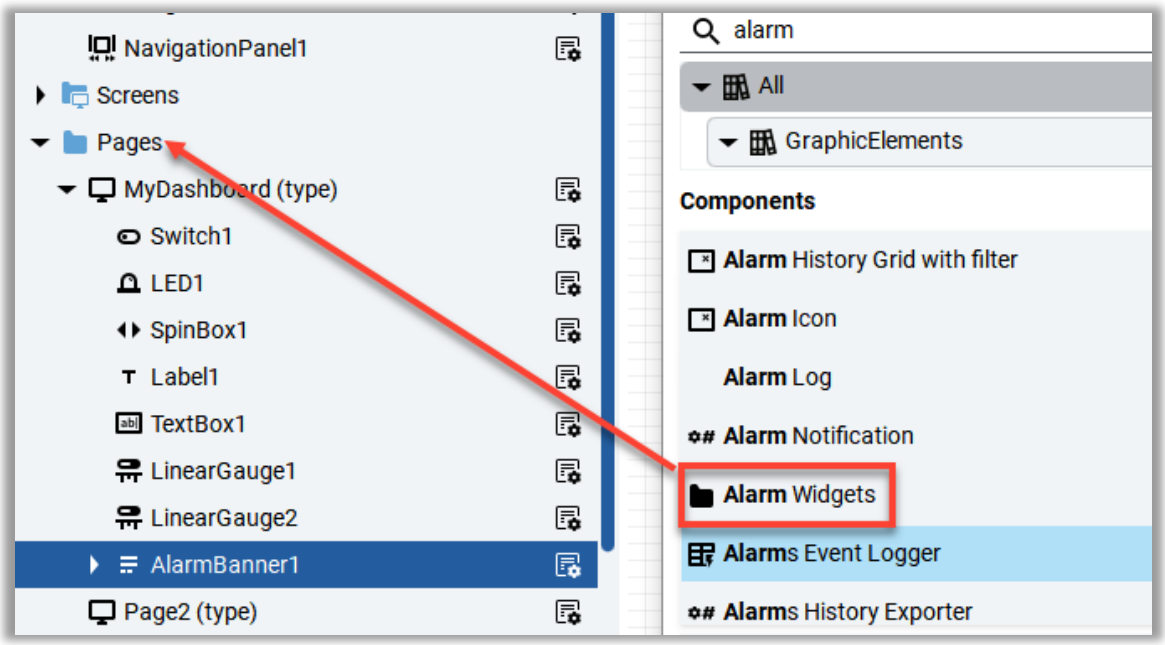


7. Open **Libraries** from the top menu.

8. In the search bar, type “alarm”. Select the **Alarm Banner** and drag it to the **MyDashboard**.



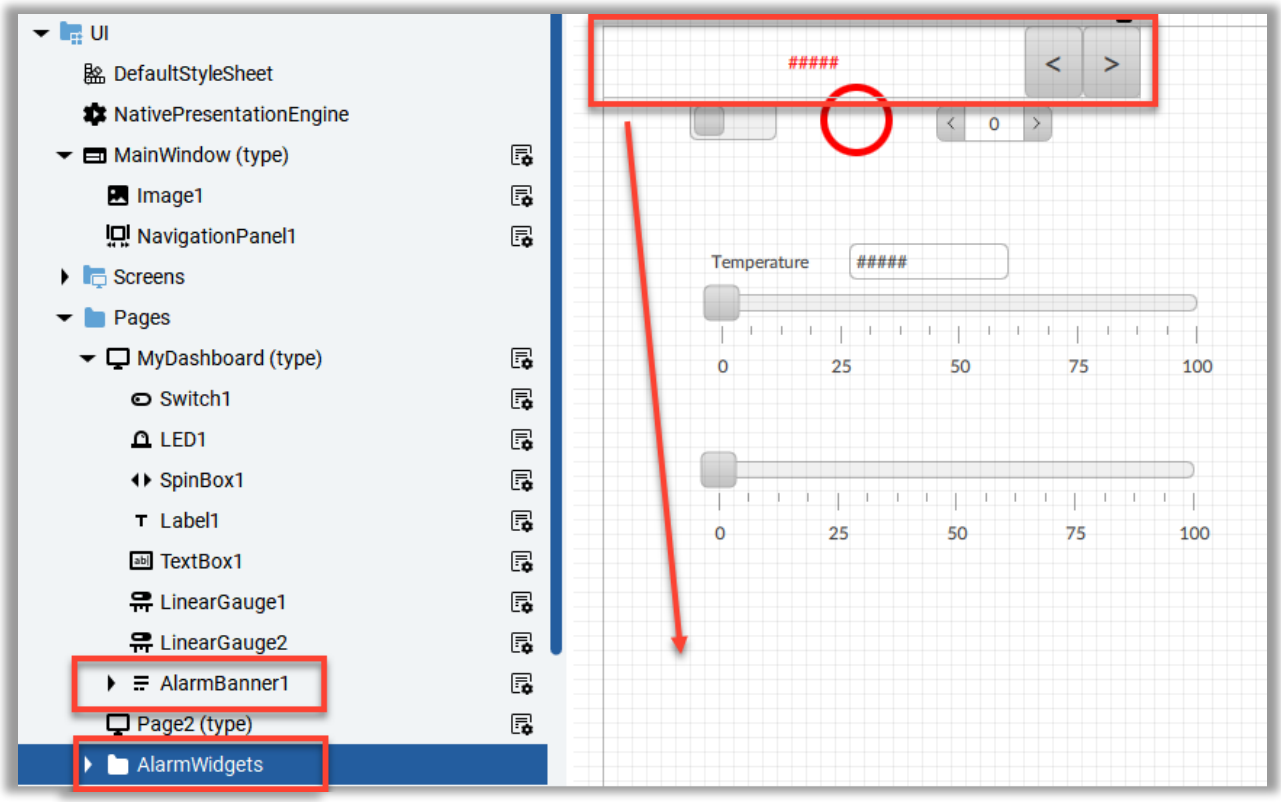
9. Scroll down in **Libraries** and select the **Alarm Widgets** folder and drag it to the **Pages** folder.



LIBRARY OBJECTS

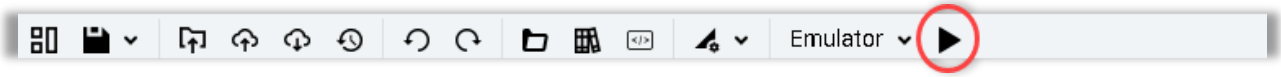
The Alarm banner widget is an object that can be placed directly on a screen or other container. The Alarm Widgets folder contains several objects that are contained in panels, so they can be added to the Pages folder and then added to the Navigation Panel.

10. Close **Libraries**. Your **Project View** and **MyDashboard** page should look like the following screenshot.

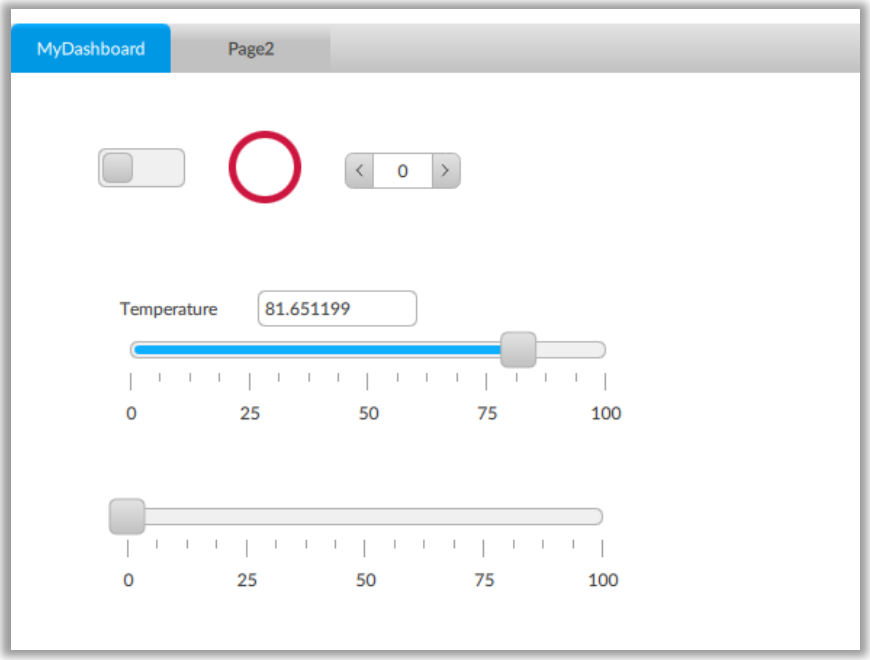


11. Select **AlarmBanner1** and move it below the second linear gauge.

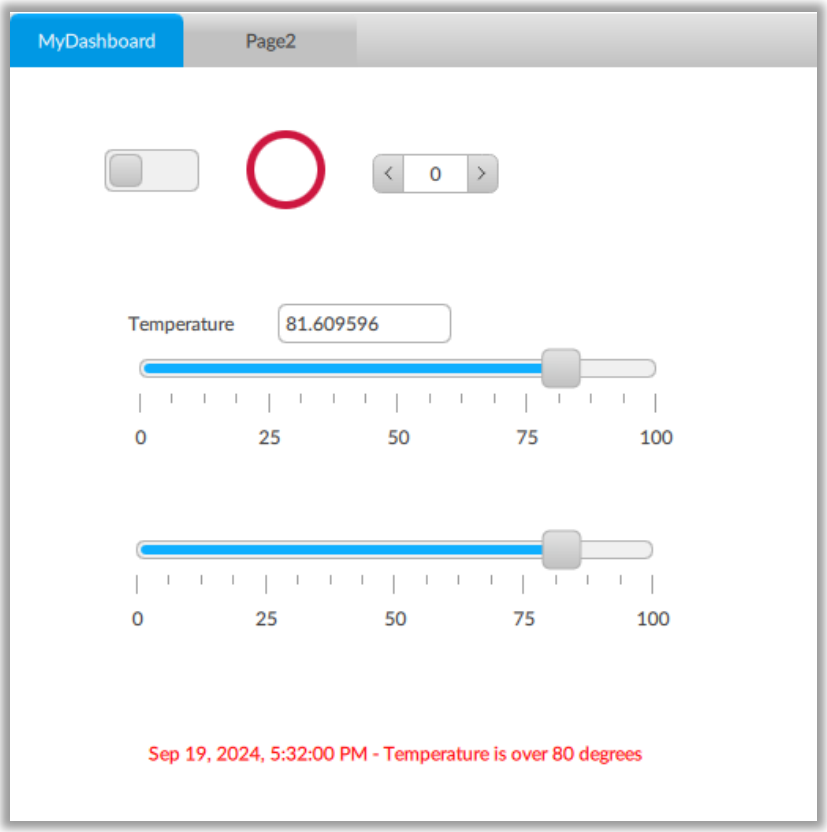
12. Run the **Emulator** to test the Alarm banner.



13. Notice the Alarm banner isn't visible on the MyDashboard screen.



14. Use the slider on the bottom linear gauge so that it is over 80 to trigger the alarm.



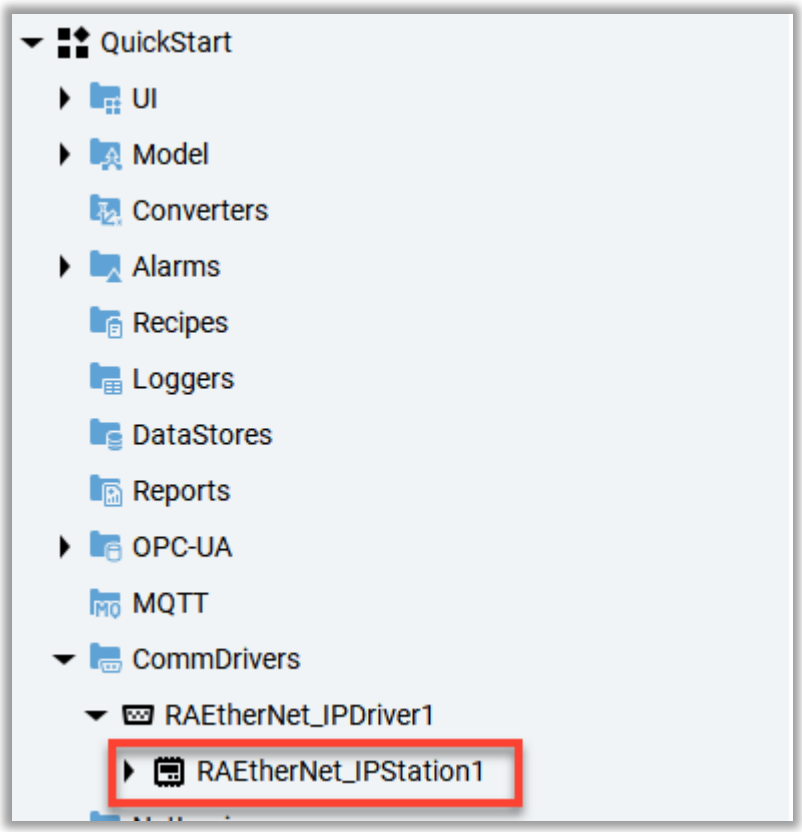
The Alarm banner has a property that sets the visibility of the alarm banner when an alarm is triggered!

15. Close the Emulator and return to FactoryTalk Optix Studio.

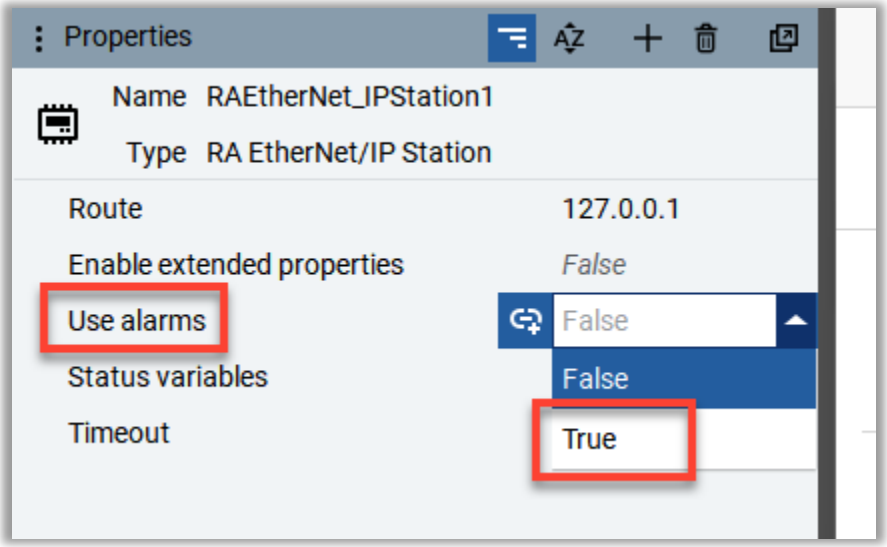
Enabling Logix Alarms

Logix controllers support two types of controller-based alarms: instruction-based (ALMA & ALMD) and Logix Tag-based alarms. FactoryTalk Optix version 1.5.x supports both Optix and Logix alarms!

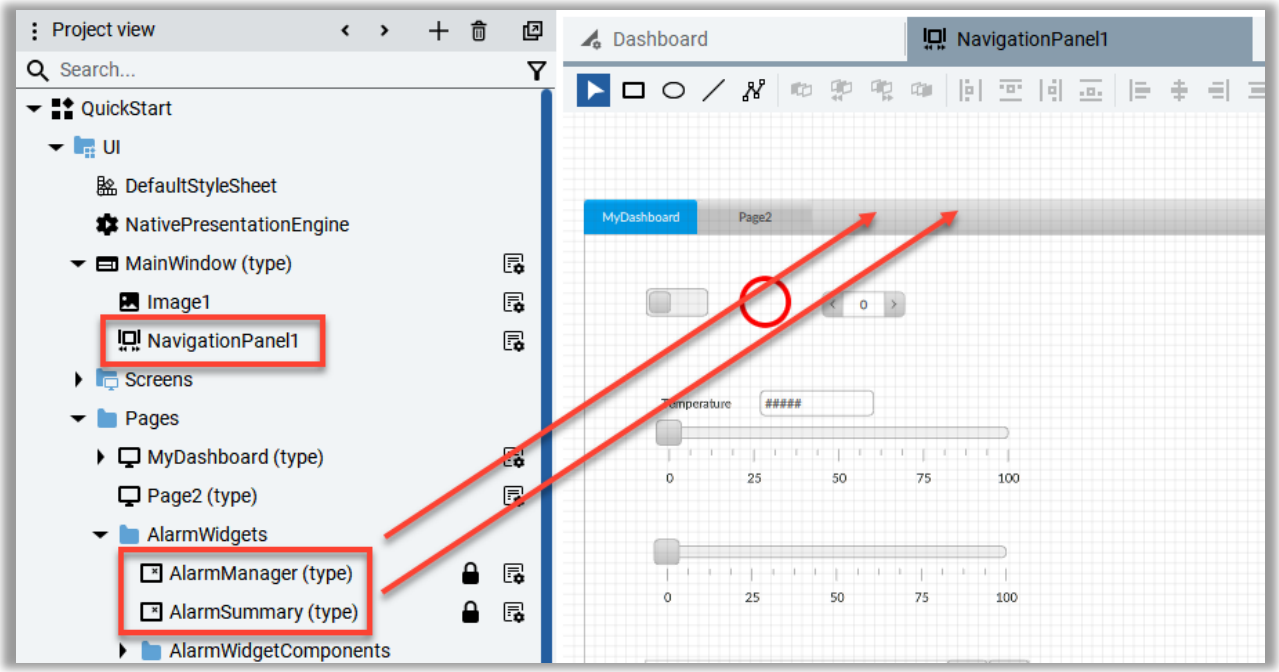
1. Expand **CommDrivers > RAEthernet_IPDriver1** and select **RAEthernet_IPStation1**.



2. In the **Properties** pane use the dropdown to set **Use Alarms** to **True.**



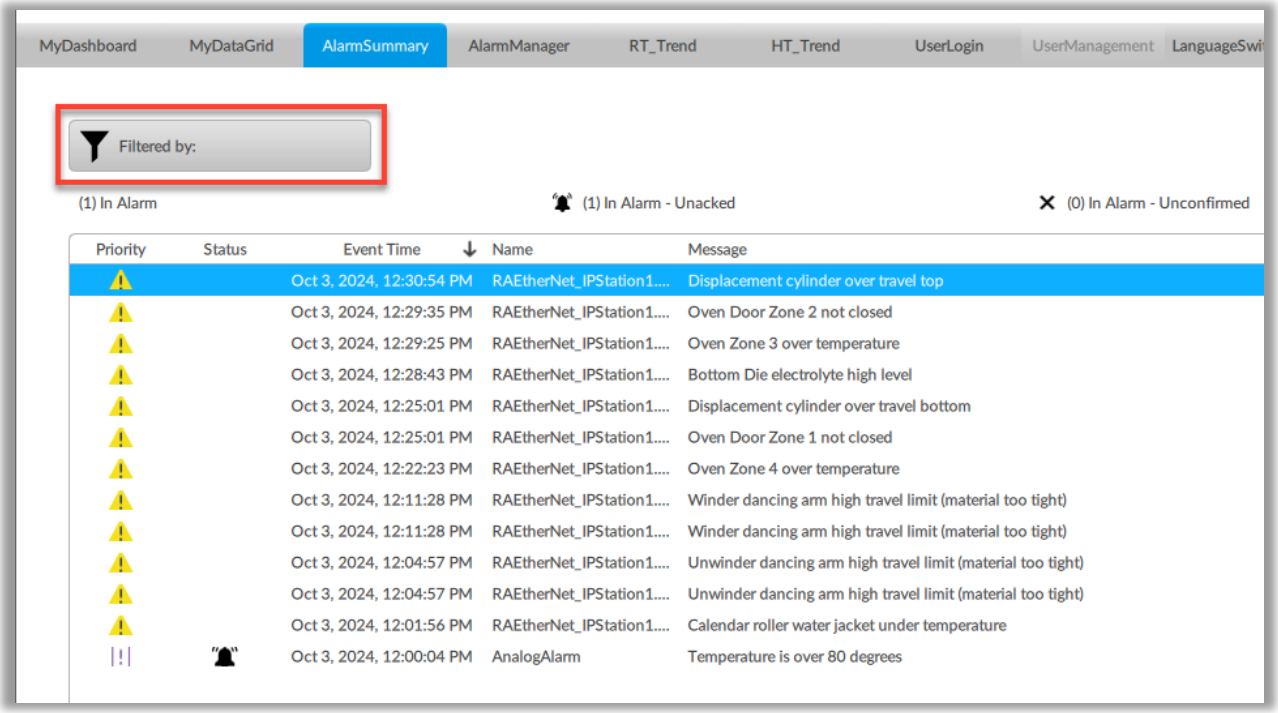
3. Add the Alarm Widget screens to the **NavigationPanel1** by expanding the **AlarmWidgets** folder and dragging the **AlarmSummary** and **AlarmManager** panels to **NavigationPanel1.**



4. Run the Emulator to view the Logix alarms and the **AlarmSummary** and **AlarmManager** screens.

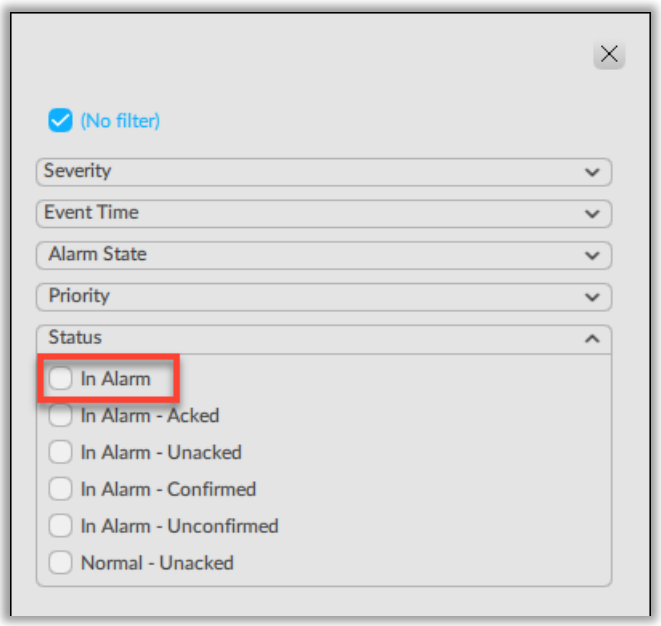
Now that there is more than one alarm, the Alarm Banner displays arrow keys to allow you to scroll through the alarms on the **MyDashboard** screen.

5. Select the **AlarmSummary** tab.



There are several alarms coming directly from the Logix controller, as well as the Optix alarm **AnalogAlarm.**

6. Select **Filtered by:** button and select **In Alarm** and then **Apply**.



Configure a Data logger

Objectives

- Use a wizard to create a datalogger
- Use a datagrid to show the data on the screen
- Create a trend and configure it as Realtime Data
- Create a trend and configure it as Historical Data

Scenario

In this section, you will learn how easy it is to create a datalog using a wizard.

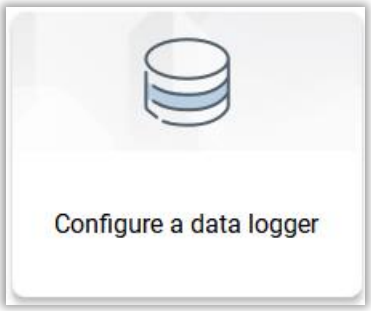
When the data is logged, you want to display the data on a screen by using the standard datagrid object that is included in FactoryTalk Optix.

After that, we will add two trends that can show real-time and historical data.

Note: Screenshots may differ depending on what optional sections have been previously completed.

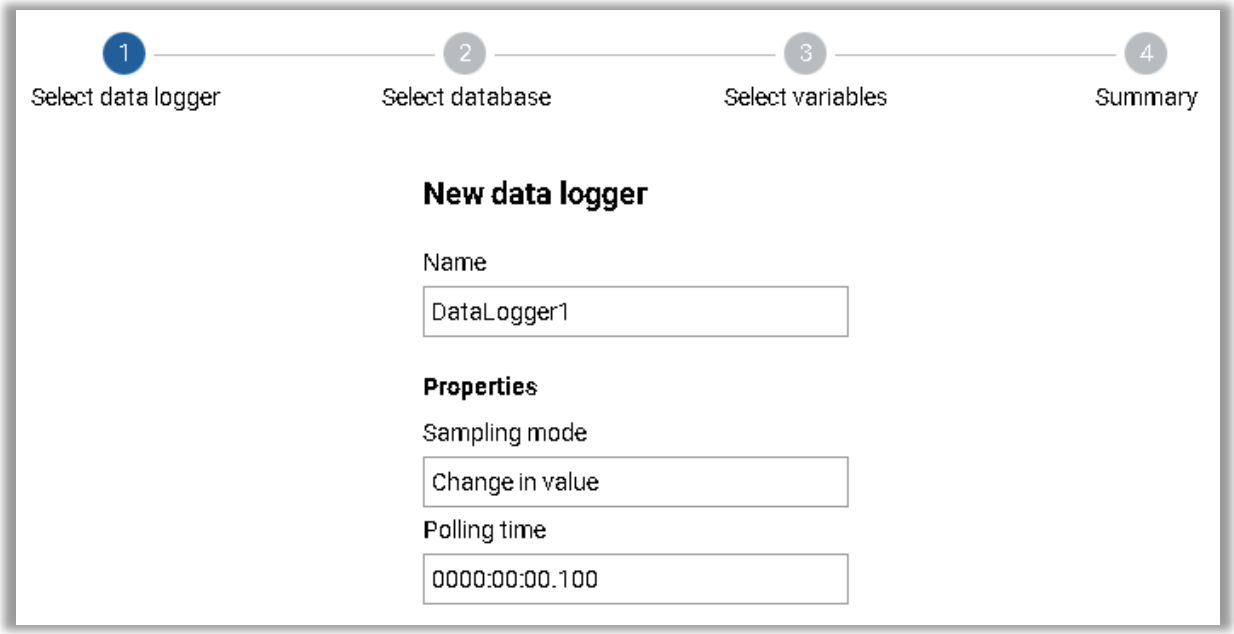
Configure a Data logger:

1. From the FactoryTalk Optix Studio toolbar, select the **Open dashboard page** icon .
2. Select **Configure a data logger**.



3. Select **New data logger** to create a data logger.
4. Select a **Sampling mode** of **Change in value**. Sample all monitored variables with values that changed from the previous sampling

5. Select **Polling time**. Set the sampling properties using the format: **0000:00:00.100**.



6. Select **Next** and **Next** again.
7. In **Available databases**, select **New database**.
8. Edit **Name** and set it to "DataLogDB".
9. Select **Local InfluxDB** connection for the **Type** and select **Next** and then select **Next** again.

NOTE

FactoryTalk Optix™ v1.4 provided support for connecting to a remote InfuxDB database. FactoryTalk Optix™ v1.5 now also provides support for a local InfluxDB database option.

10. Click **Next** and select the **VarTemperature** variable under the **Model** folder to log.
11. Select **Next**.
12. Wait for the successful message and **Exit**.

INFORMATION ABOUT DATA STORES

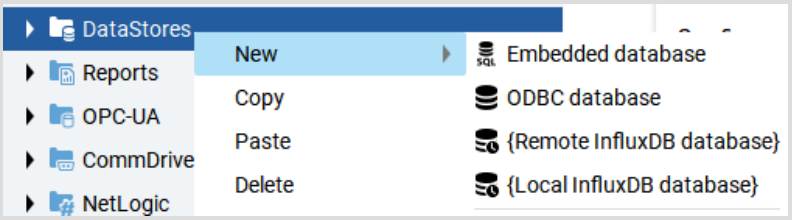
FactoryTalk Optix™ provides support for three kinds of data stores:

1. **Embedded database**: An object that represents a SQLite database internal to the project at runtime. You typically use an Embedded database object to manage data supporting other objects within the FactoryTalk Optix™ Application. An Embedded database is not accessible from an external application while the Runtime is running.

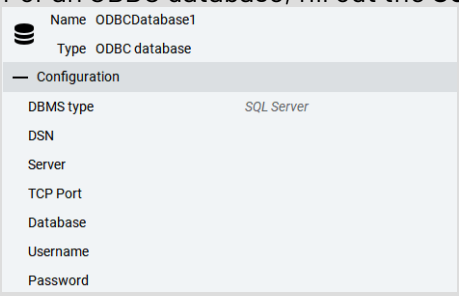
2. **ODBC database**: An object that represents a local or remote database supported by the Open Database Connectivity (ODBC) protocol. You can use ODBC to query or populate a remote database maintained with other systems that contains data for the project. Depending on the access management policy of an external database, the external database can have read/write limits.

3. **InfluxDB database**: A high-performance, time series database capable of storing any form of time series data. It is well-suited for applications that involve tracking and analyzing data points over time. It excels in scenarios where data is being written continuously at high volumes while users also require the ability to query that data quickly.

To create a database connector, in **Project view**, right-click **DataStores,** select **New**, and select the desired database:

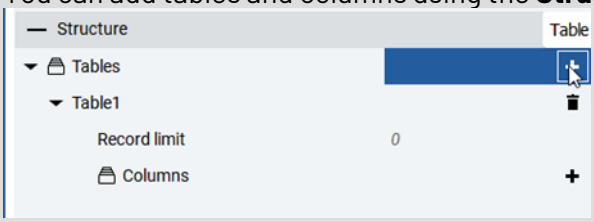


For an ODBC database, fill out the **Configuration** properties:



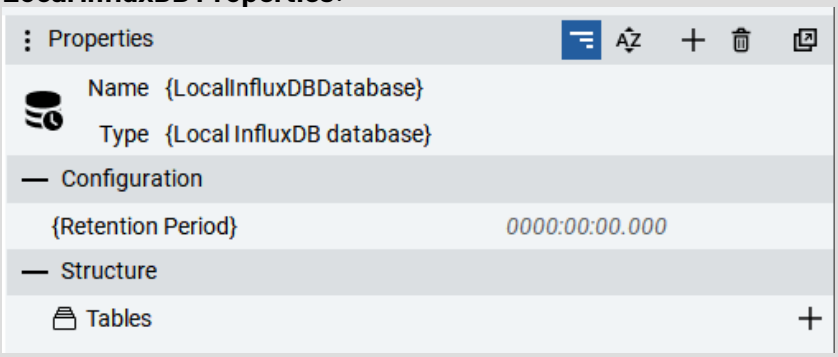
If using a Windows device to connect to the ODBC database, you can use a **DSN** (Data Source Name – which comes from the ODBC driver configuration). Otherwise, manually input all the properties needed for the connection.

You can add tables and columns using the **Structure** section in the **Properties** panel:



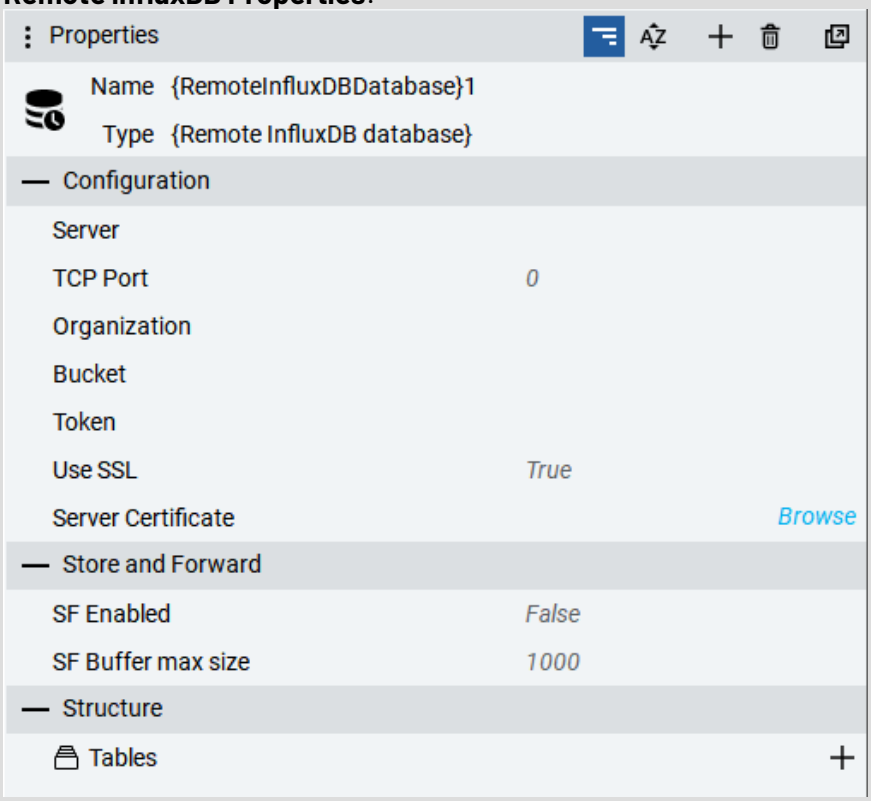
Also note, if configuring a data logger, once the database is specified, the table and columns are generated for the database.

Local InfluxDB Properties:



Retention Period: the duration in seconds for how long the data will be kept in the database. The default value is 0, which represents infinite retention.

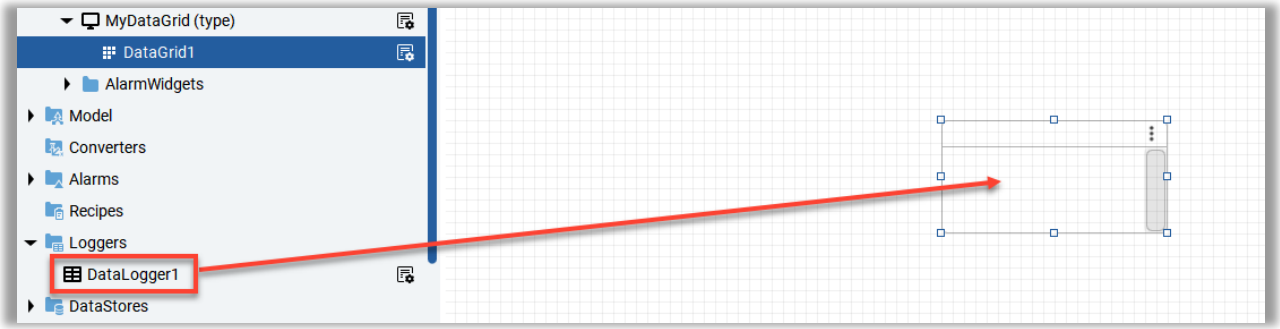
Remote InfluxDB Properties:



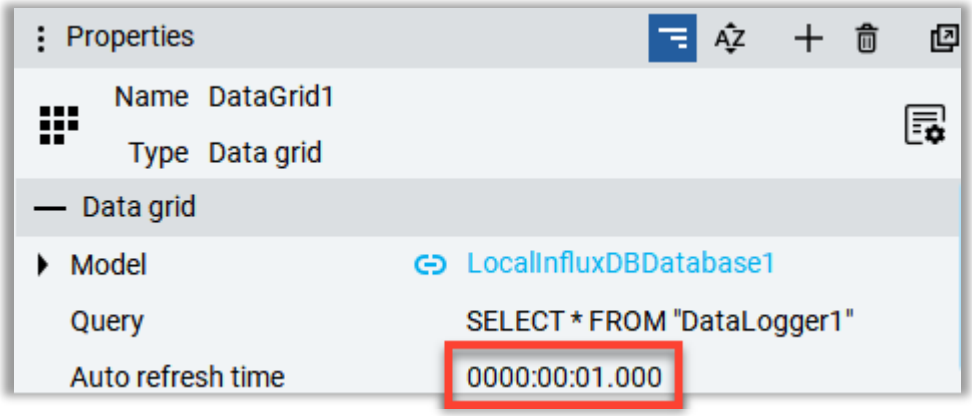
Server: IP address or hostname of the database server in the network.
TCP Port: TCP port of the database server.
Organization: inherited from the InfulxDB data store. It is a workspace for a group of users. All dashboards, tasks, buckets, and members belong to an organization.
Bucket: inherited from the InfulxDB data store. It is a named location where time series data is stored for a set duration of time. A bucket belongs to an organization.
Token: inherited from the InfulxDB data store. It provides secure interaction between InfluxDB and clients. The token is specific to a user and identifies InfuxDB permissions within a user's organization.
Use SSL: enables the use of SSL encryption.
Server Certificate: server certificate used during SSL encryption.
SF Enabled: enables Store and Forward.
SF Buffer max size: maximum number of elements in the buffer ranging from 1 to 100,000.

You can display the data logger and event logger data in a table. The Data grid object shows the data stored by the logger in the database.

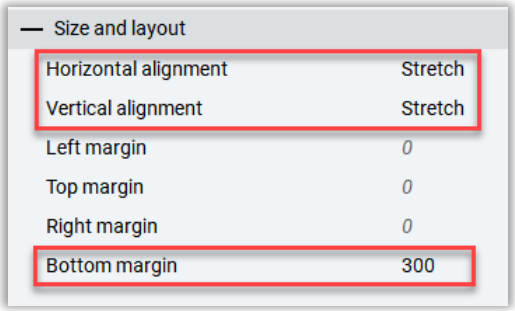
13. In the **Project view** pane, expand **UI > Pages** and rename **Page2 (type)** to "MyDataGrid".
14. Right-click on **MyDataGrid** page and select **New > Data controls > Data grid**.
15. Double-click the **DataGrid1** object to open it in the editor.
16. From the **Project view** pane, drag **DataLogger1** from the **Loggers** folder to the Data grid object in the editor called **Datagrid1**.



17. To display the data updated in real time at runtime, configure the **Auto refresh time** property of **DataGrid1** using the format**: 0000:00:01.000**.



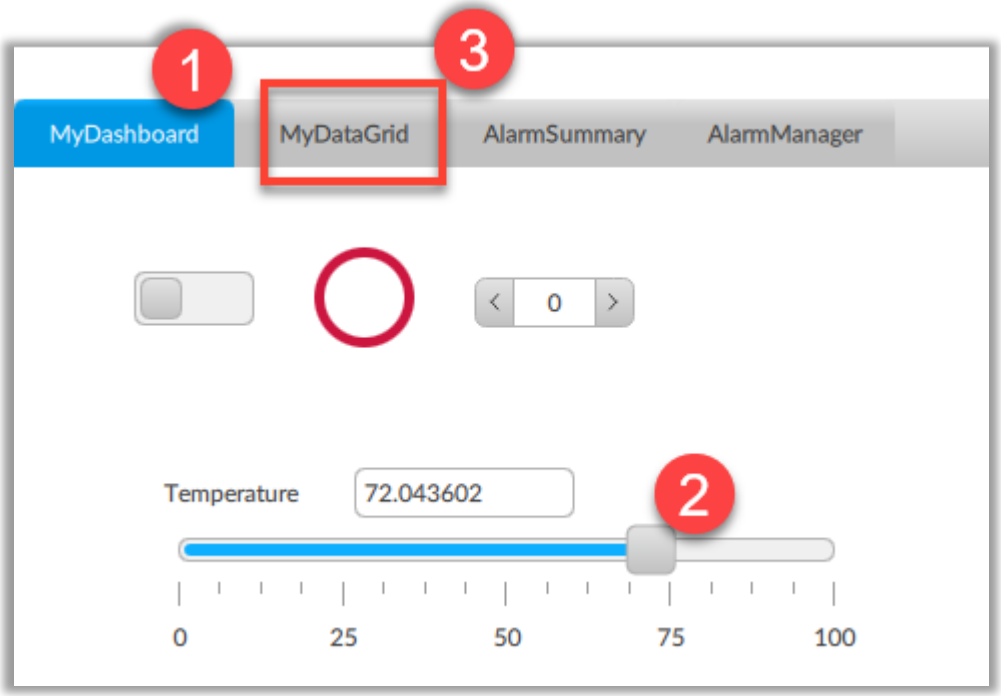
18. In **Properties**, scroll down to the **Size and layout** section and set the **Horizontal alignment** and **Vertical alignment** to **Stretch** and the **Bottom margin** to "300".



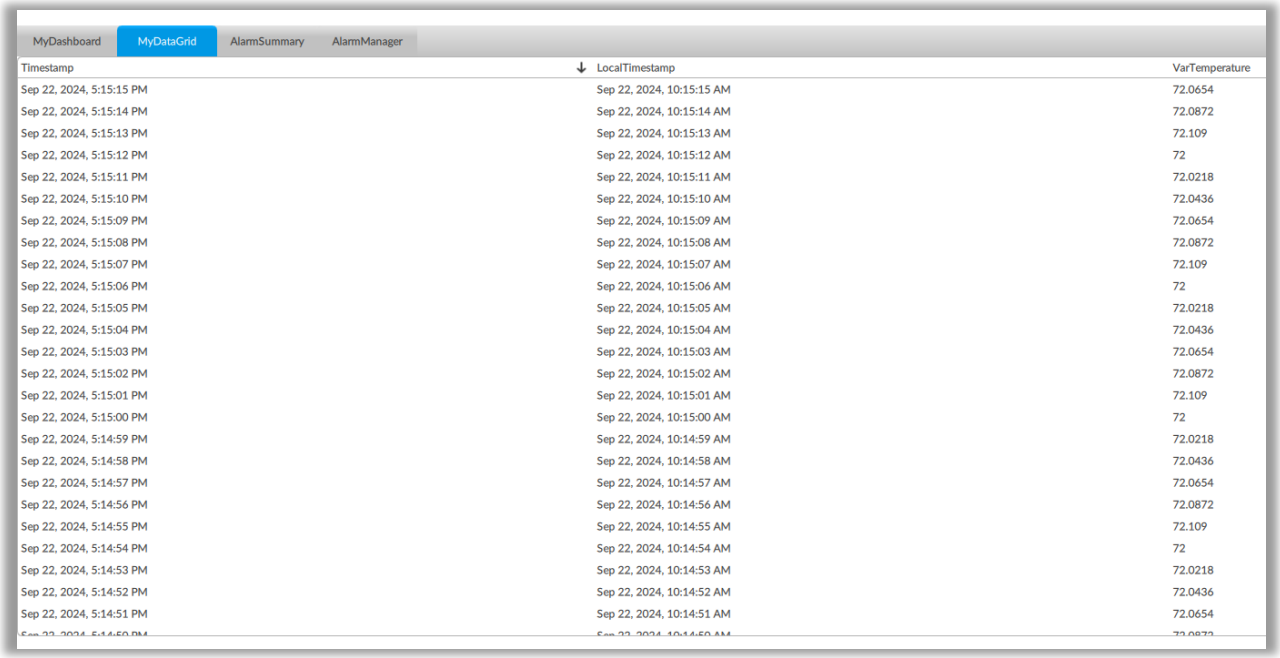
19. Save the project.

20. From the toolbar with Emulator still displayed, click the **Emulator** play icon ▶.

21. At runtime, under **MyDashboard**, change the temperature setpoint value and you will see the temperature change automatically.



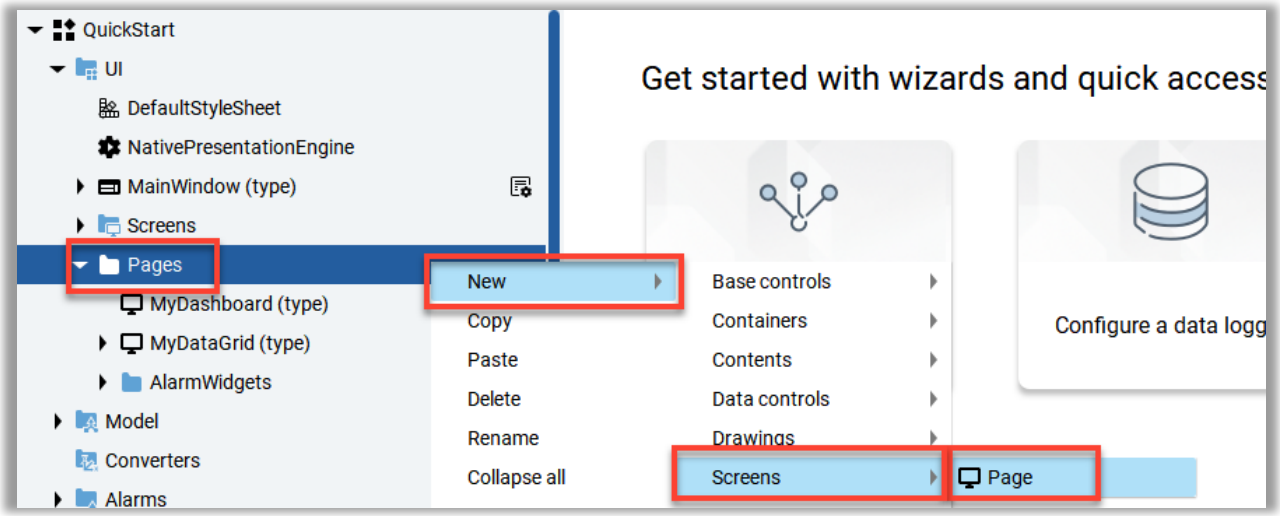
22. Navigate to **MyDataGrid** and observe the value changes in the Data Grid.



23. **Close** the Emulator.

Realtime Trending

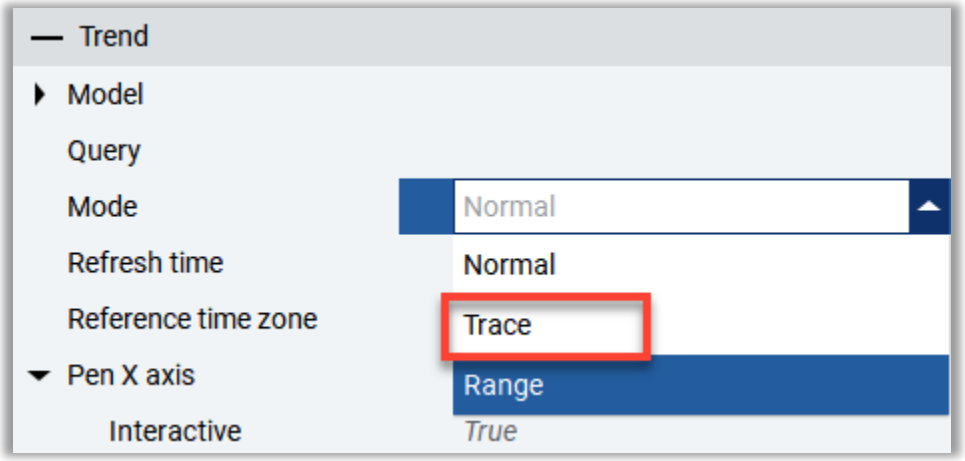
1. In the **Project view** pane, right-click **UI > Pages > New > Screens > Page**.



This will create a **Page1 (type)** object.

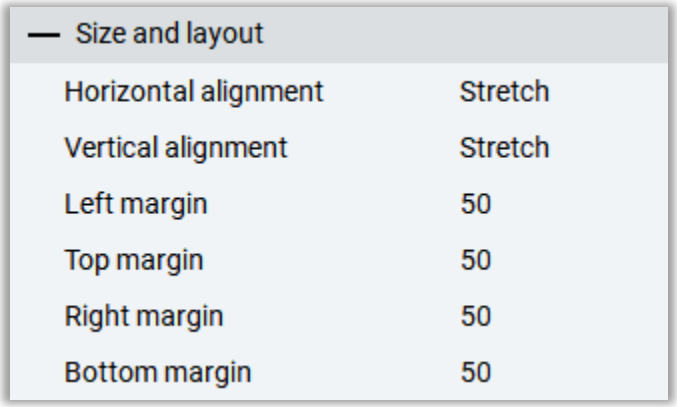
2. Right-click on **Page1 (type)** and rename the page to "RT_Trend".

3. Copy **RT_Trend (type)** to use for Historical Trending in the next section. Right-click and select **Copy**, then right-click **Pages** and select **Paste**.
4. Rename **RT_Trend1 (type)** to "HT_Trend".
5. Right-click on **RT_Trend** page and select **New > Data controls > Trend**.
6. Double-click the **Trend1** object to open it in the editor.
7. In **Properties**, under **Model**, change the **Mode** from **Normal** to **Trace**.

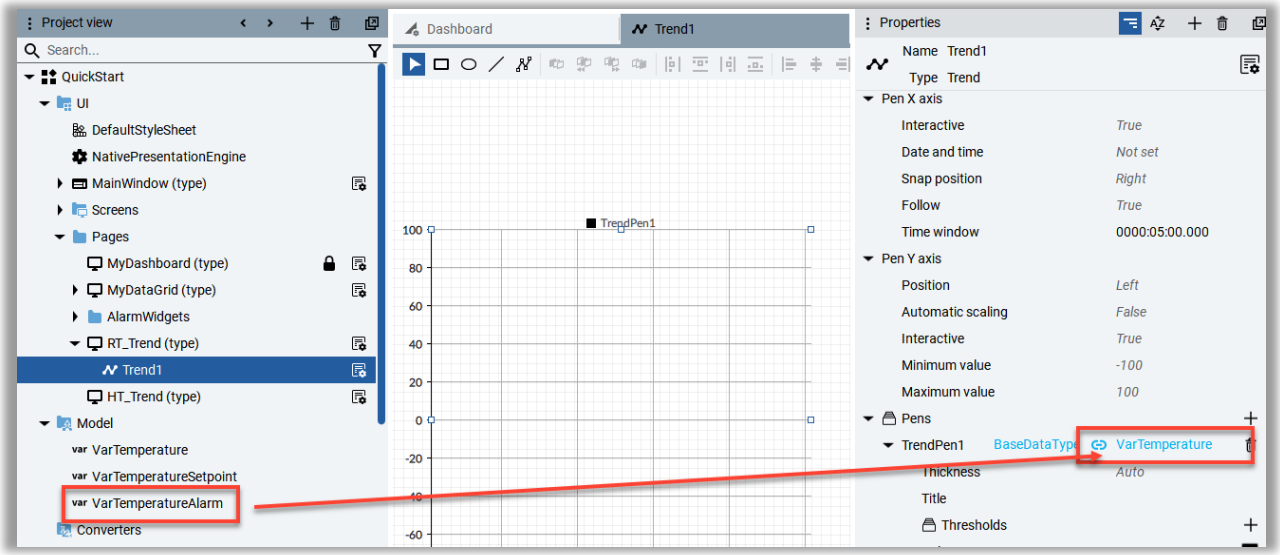


8. Set the **Time window** to 5 minutes using the following format: "0000:05:00:000".

9. Set the **Horizontal alignment** and **Vertical alignment** to **Stretch**.
10. Set **Left**, **Top**, **Right,** and **Bottom margins** to "50".



11. Under **Pens**, drag and drop to create a dynamic link between **TrendPen1** and the pen data source **VarTemperature** variable.



12. To change the Color of the pen, select the edit icon, enter “RED”, and hit enter on the keyboard.

13. Save the project.

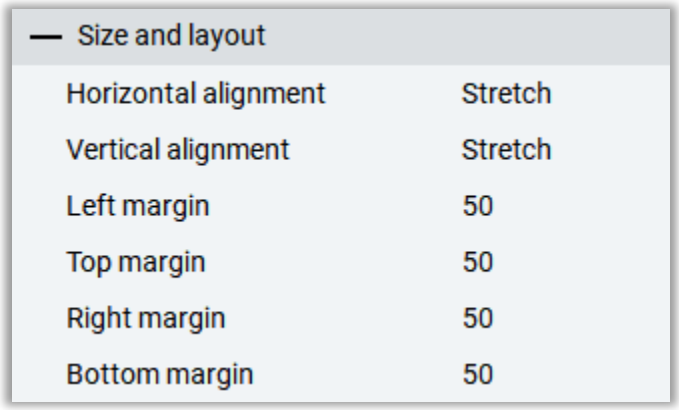
Historical Trending:

1. Right-click on the **HT_Trend** page and select **New > Data controls > Trend**.
2. In **Properties** set the **Time window** to 5 minutes using the following format: “0000:05:00:000”

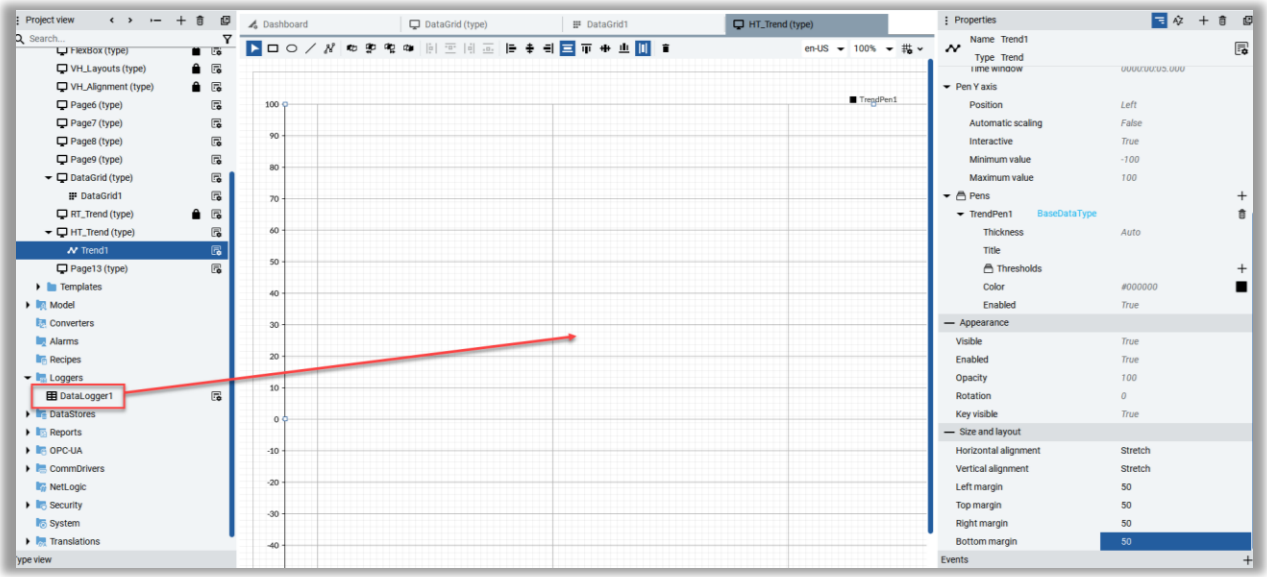
Time window	0000:05:00.000

3. Set **Horizontal alignment** and **Vertical alignment** to **Stretch**.

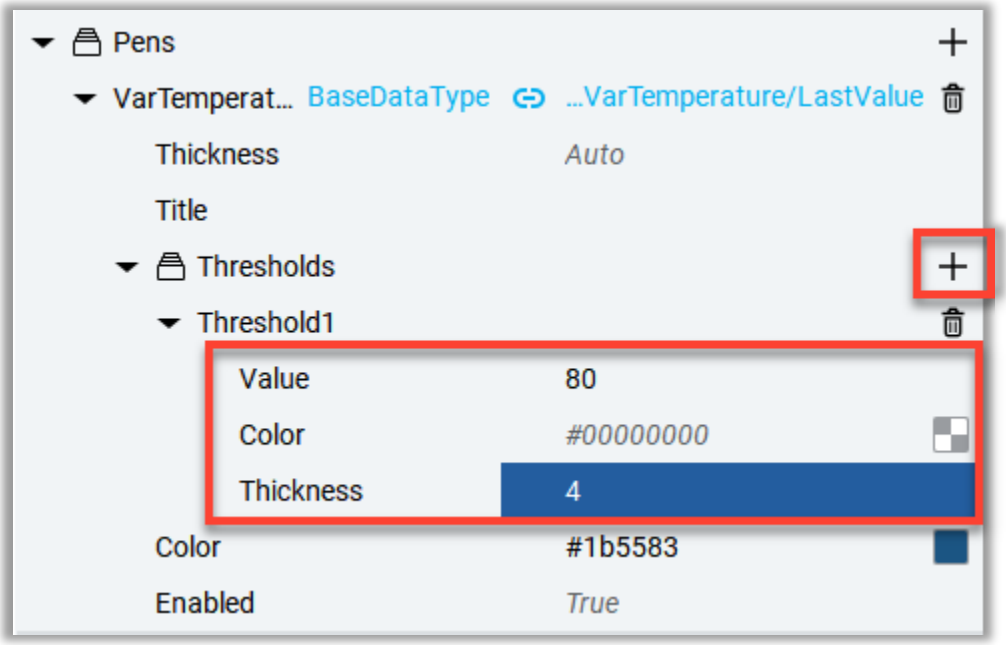
4. Set **Left**, **Top**, **Right** and **Bottom** margins to “50”.



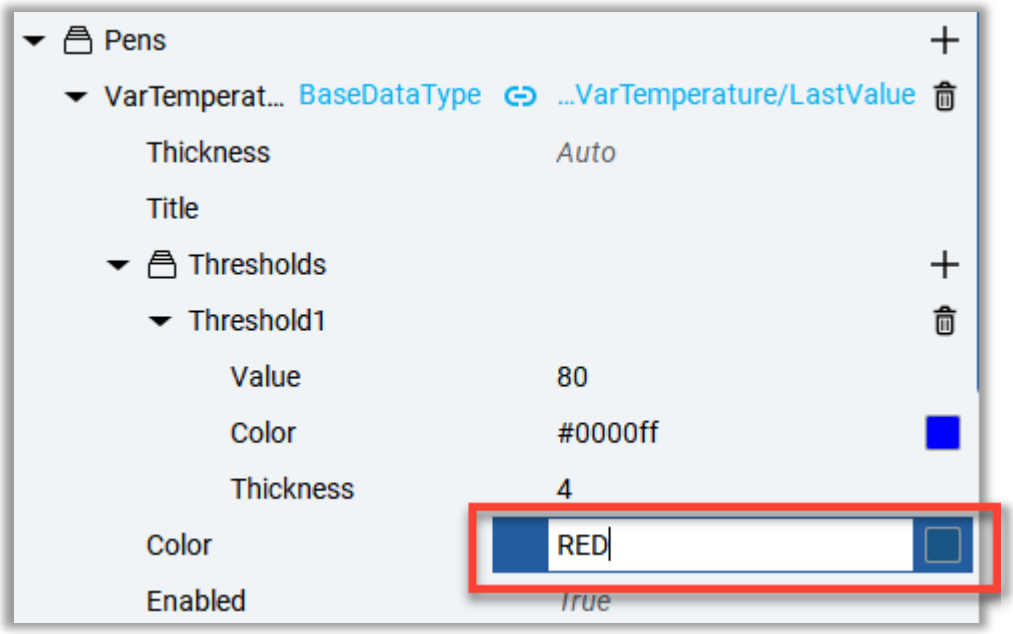
For historical trending, we need to provide the trend with a model. Use drag and drop to drag the **DataLogger1** we created earlier under **Loggers** and drop it on the **trend** chart to automatically update the **Model** and **Pens** properties of the trend chart.

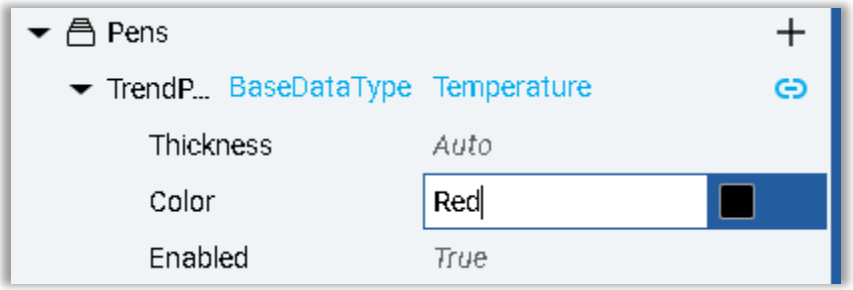


5. Under **Pens**, click on the **+** next to **Thresholds** and change the properties of **Threshold1** to match what is shown below.

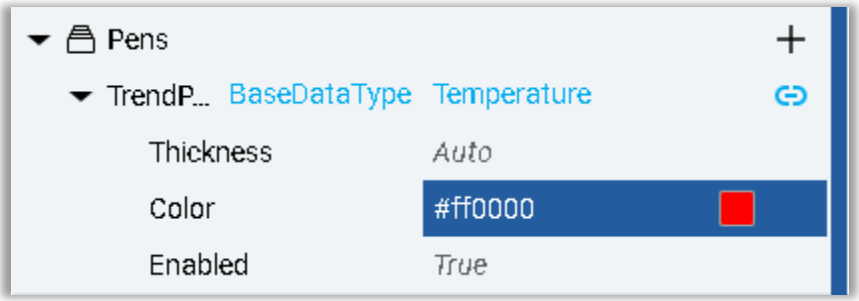


6. To change the **Color** of the pen, select the edit icon ✎, enter "RED", and hit **enter** on the keyboard.

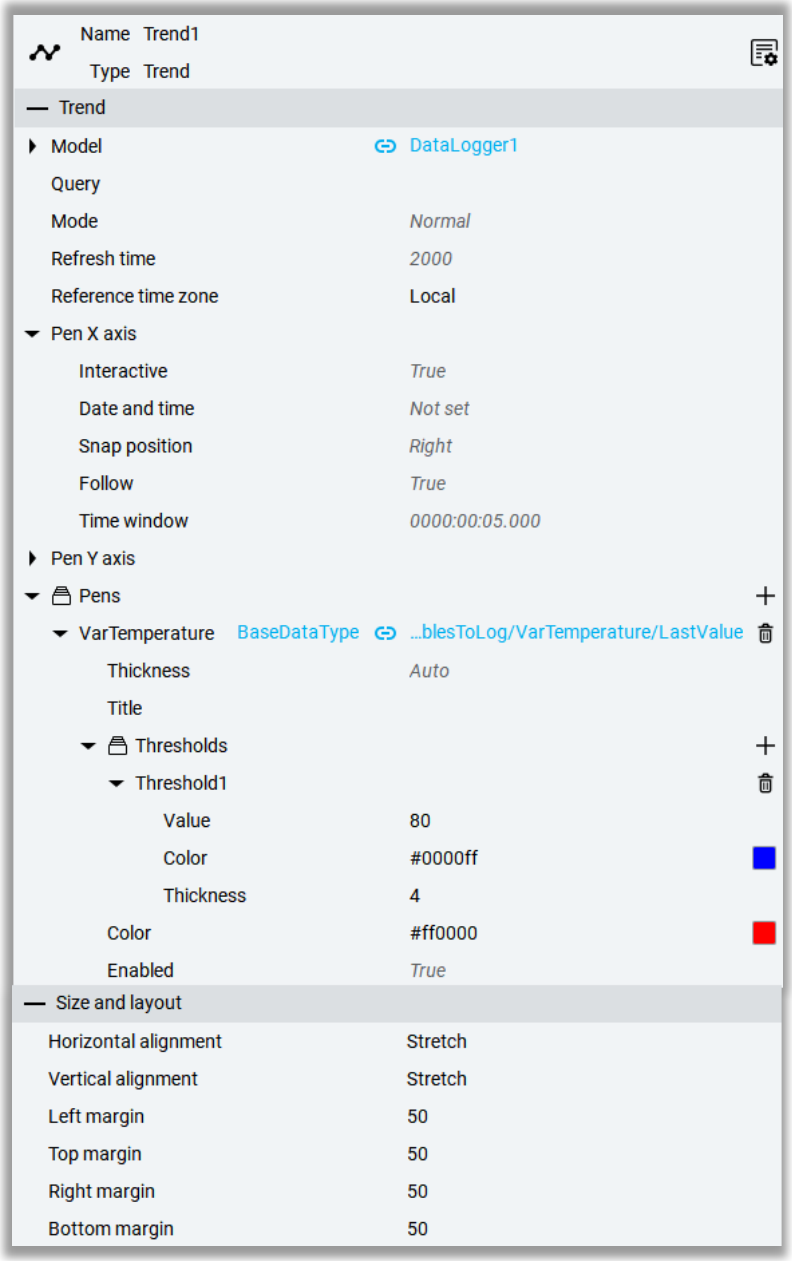




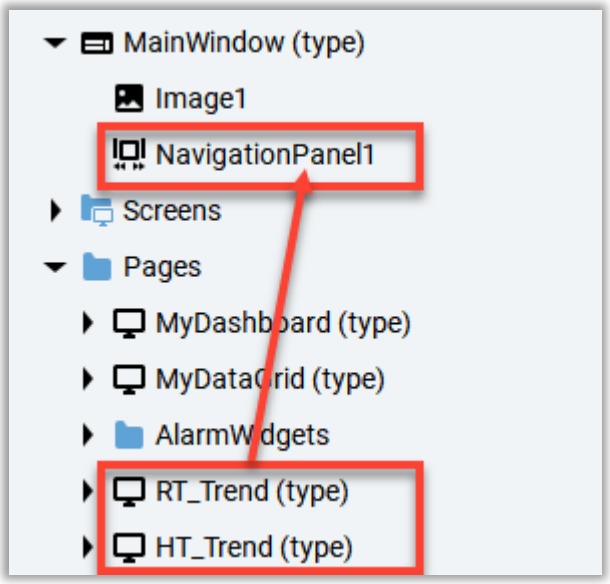
7. The **Color** property is changed to red, and the appropriate hex value of #ff0000 is used.



8. The final configuration of the trend properties will look like the following:



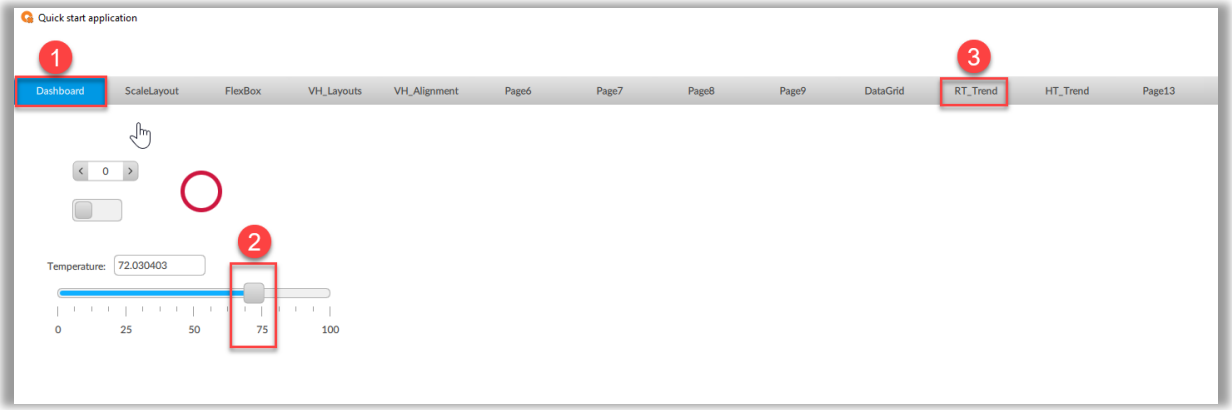
9. Select **RT_Trend** and **HT_Trend** and drag them to **NavigationPanel1**.



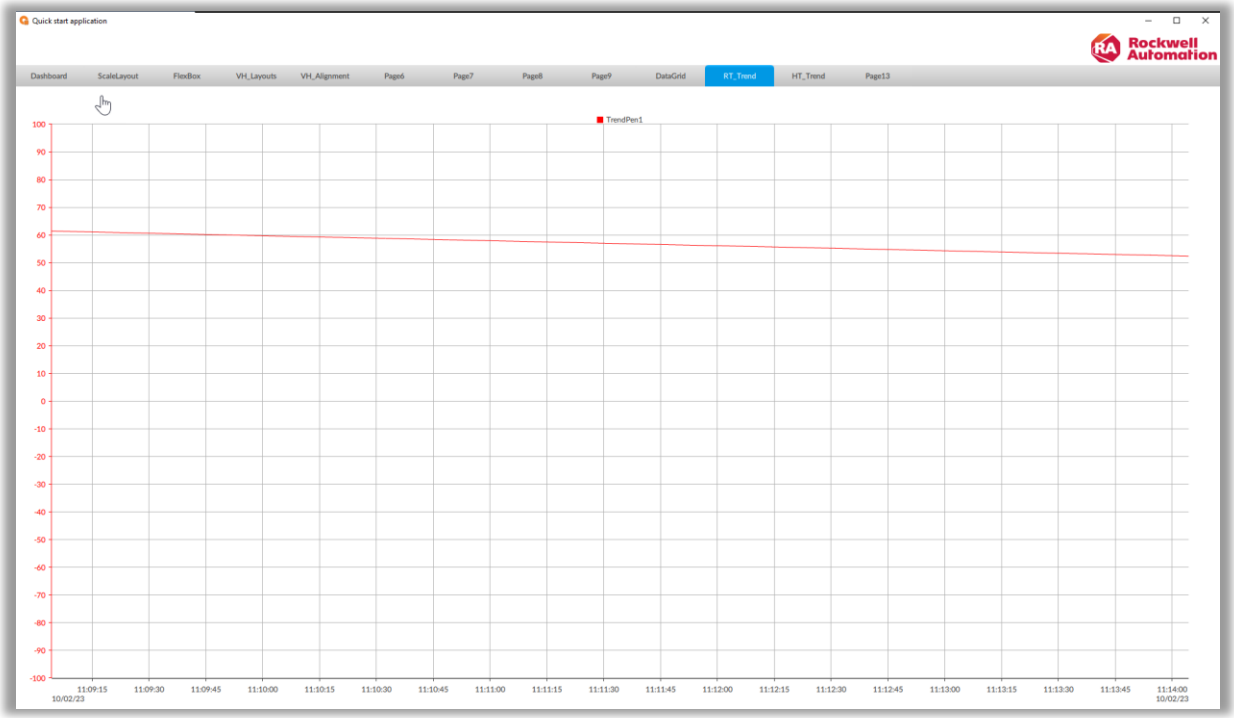
10. **Save** the project.

11. From the toolbar, with Emulator still displayed, click the **Run on Emulator** icon ▶.

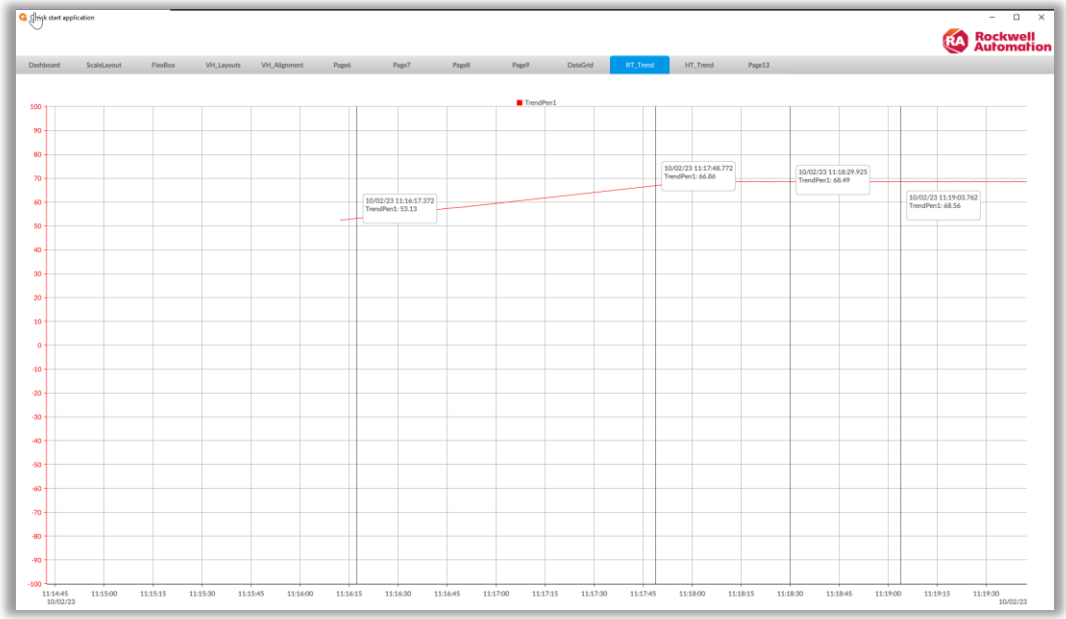
12. From **Dashboard**, change the temperature value using the gauge.



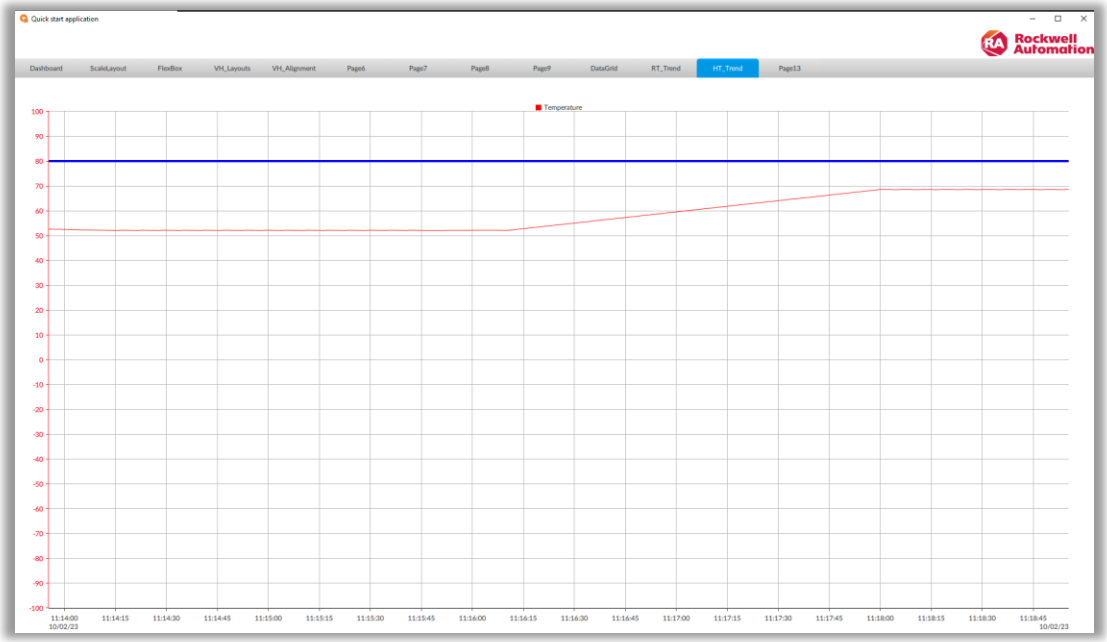
13. Navigate to **RT_Trend** to view the real-time trend. After a few seconds, values will start to appear. Note: We did not change the default refresh time of 2 seconds.



14. You can click on the trend to add traces so you can see the values at specific times.



15. Navigate to **HT_Trend** to view the historical time trend. All historical values in the last few minutes are plotted.



Note: You can zoom in and zoom out of a trend chart in three different ways:

1. Use the pinch-to-zoom gesture.
2. Place the mouse pointer in the area of interest and scroll forward or backward with the mouse wheel.
3. Select the Trend graph and select + or -.

16. **Close** the Emulator.

Security

Objectives

- Examine the Project Settings regarding security
- Create users and groups during design time
- Create screens to be used during runtime to manage users and log into the application
- Add elements to a screen to explore some security and localization settings during runtime

Scenario

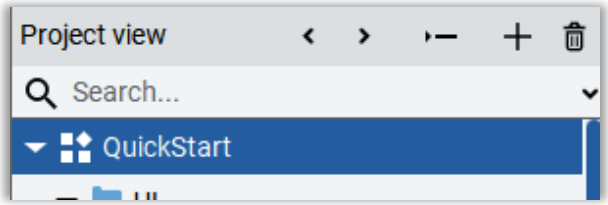
FactoryTalk Optix provides the ability to perform project, local, and domain authentication. Users can be created both during design time and runtime – with runtime creation limited to project-scoped users. FactoryTalk Optix also provides the ability to assign a locale to a user. Therefore, when a specific user logs into the application, the values and associated engineering units will convert to the locale assigned to that user. Additionally, any strings that have been translated will also convert to the user's native language.

Security Project Settings

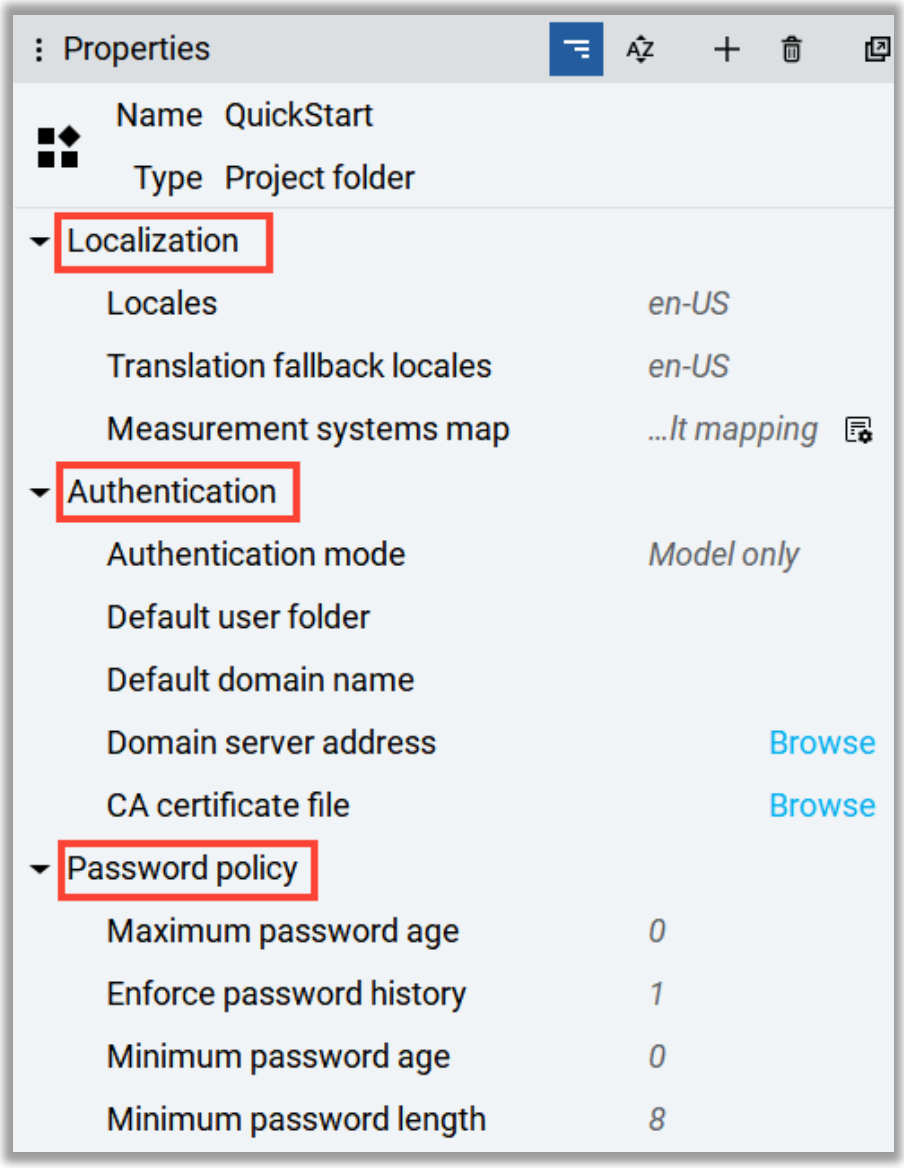
Note: Screenshots may differ depending on what optional sections have been completed previously.

First, you will examine the **Localization**, **Authentication**, and **Password policy** properties that are available for configuration within the project.

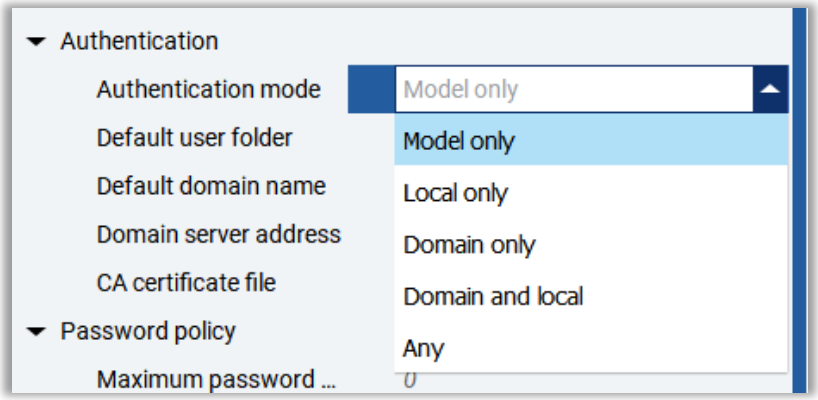
1. In the **Project view**, click **QuickStart**.



2. Notice the **Localization**, **Authentication**, and **Password policy** properties.



3. Click the **Authentication mode** drop-down arrow and observe the available options.



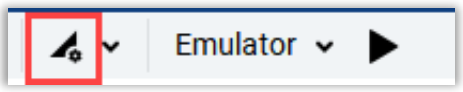
AUTHENTICATION MODES

- **Model only** uses the accounts created within the project.
- **Local only** uses accounts created on the local PC.
- **Domain only** uses accounts created in the specified domain.
- **Domain and local** uses accounts created in both the local PC and the specified domain.
- **Any** uses all of the above.

4. Leave the default option of **Model only**.

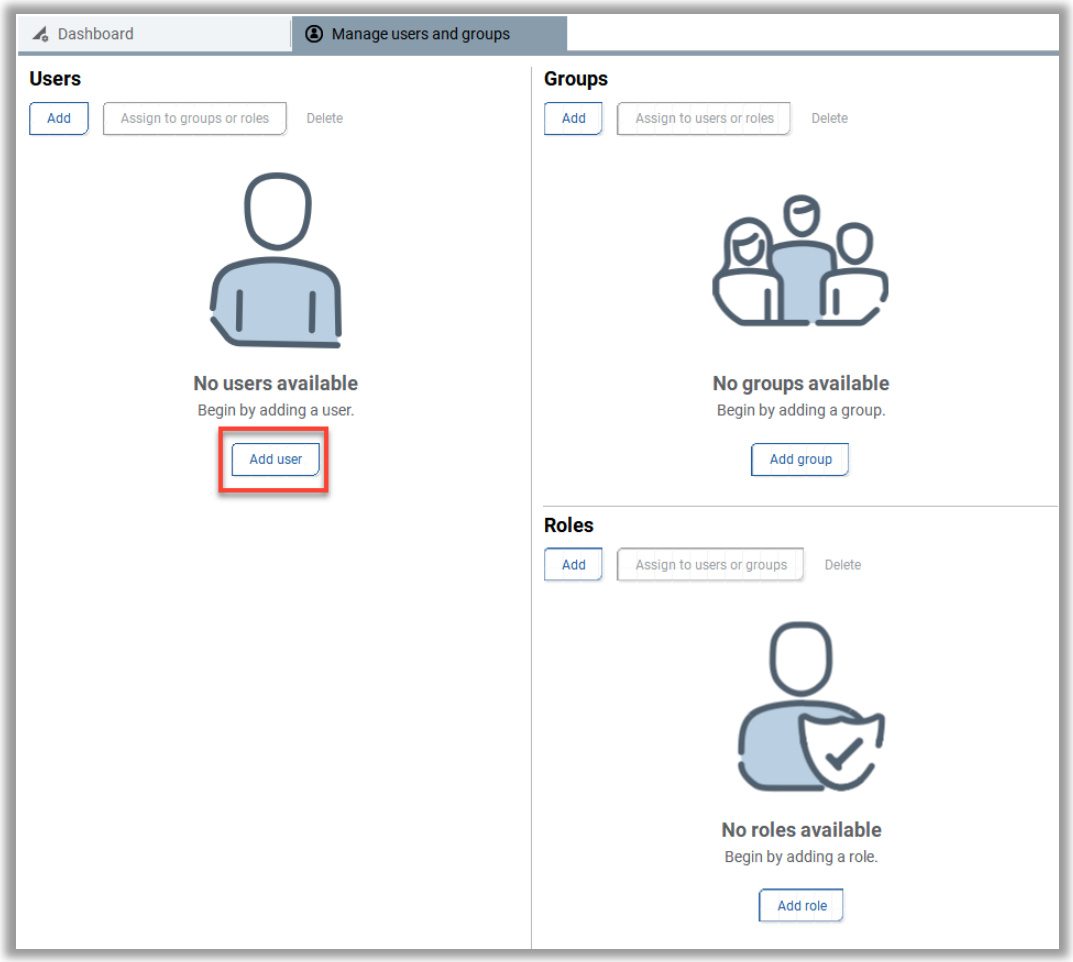
Create Users and Groups

1. Click the **Dashboard** icon if the **Dashboard** is not open already.

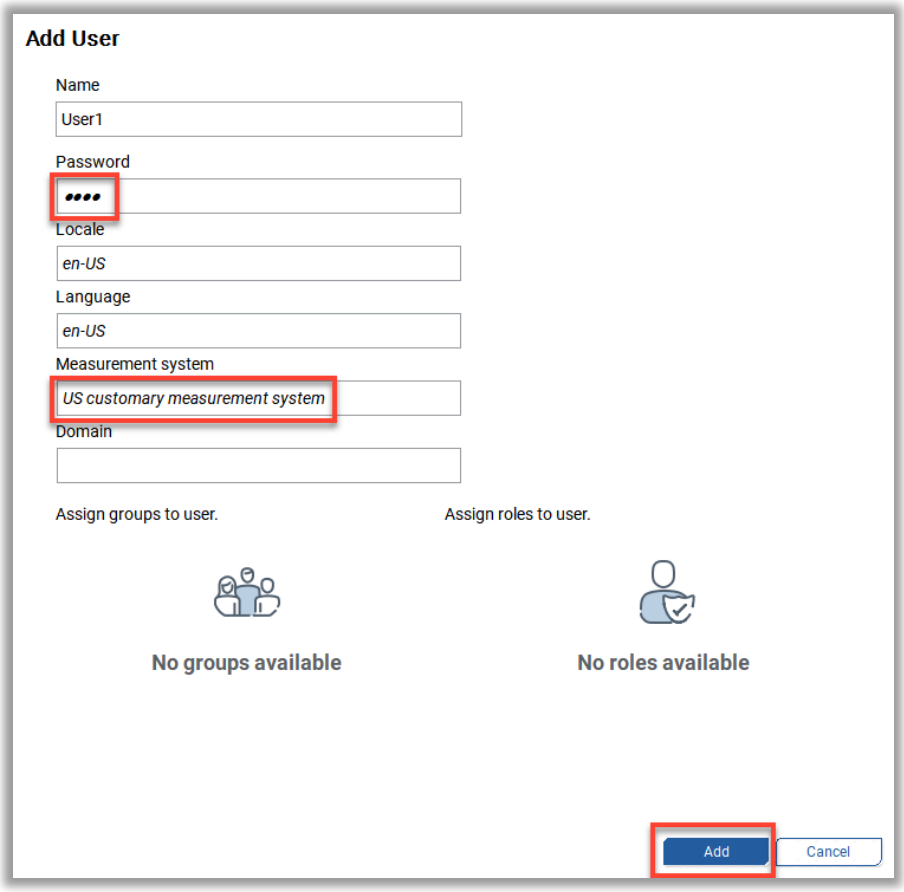


2. Click the **Manage users and groups** tile.

3. Click **Add user** under the **Users** section.



4. In the **Add user** window for **User1**, click in the **Password** field, type “password1”, and press **Enter**. Select **US customary measurement system** from the dropdown and select **Add.**



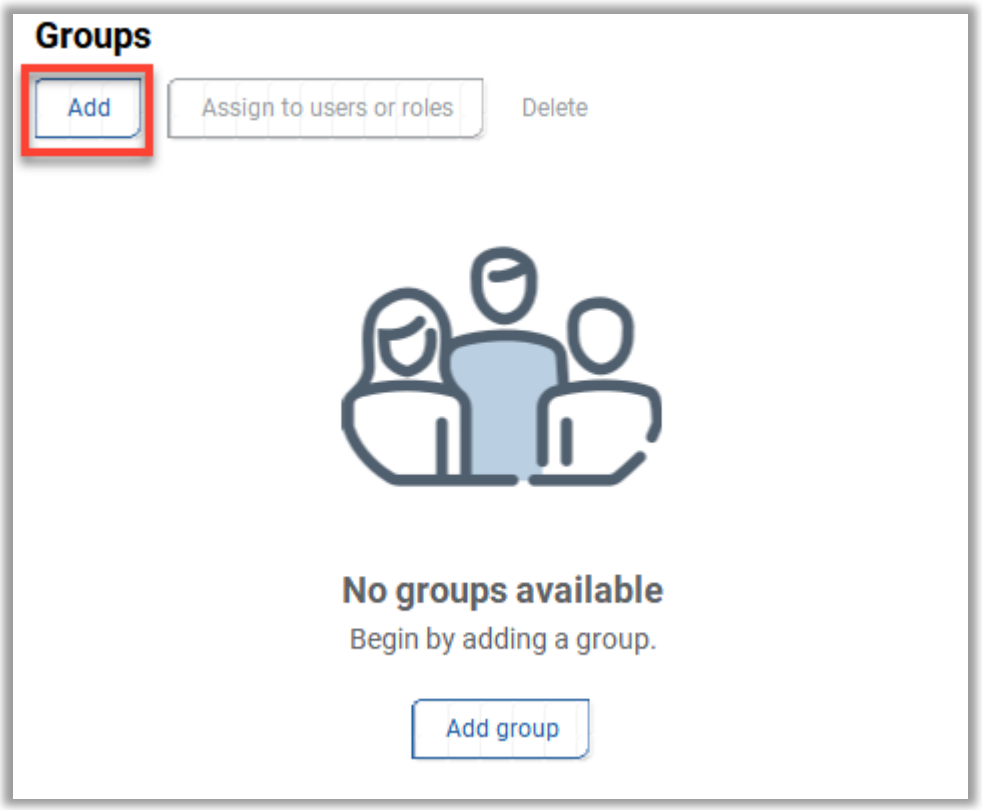
Note: If more than one locale exists in the project, you can select the desired **Locale** and **Language** for the **User**. When the user logs in, any translated text and engineering units will switch to the user’s native language and locale.

5. Leave the **Domain** field blank.

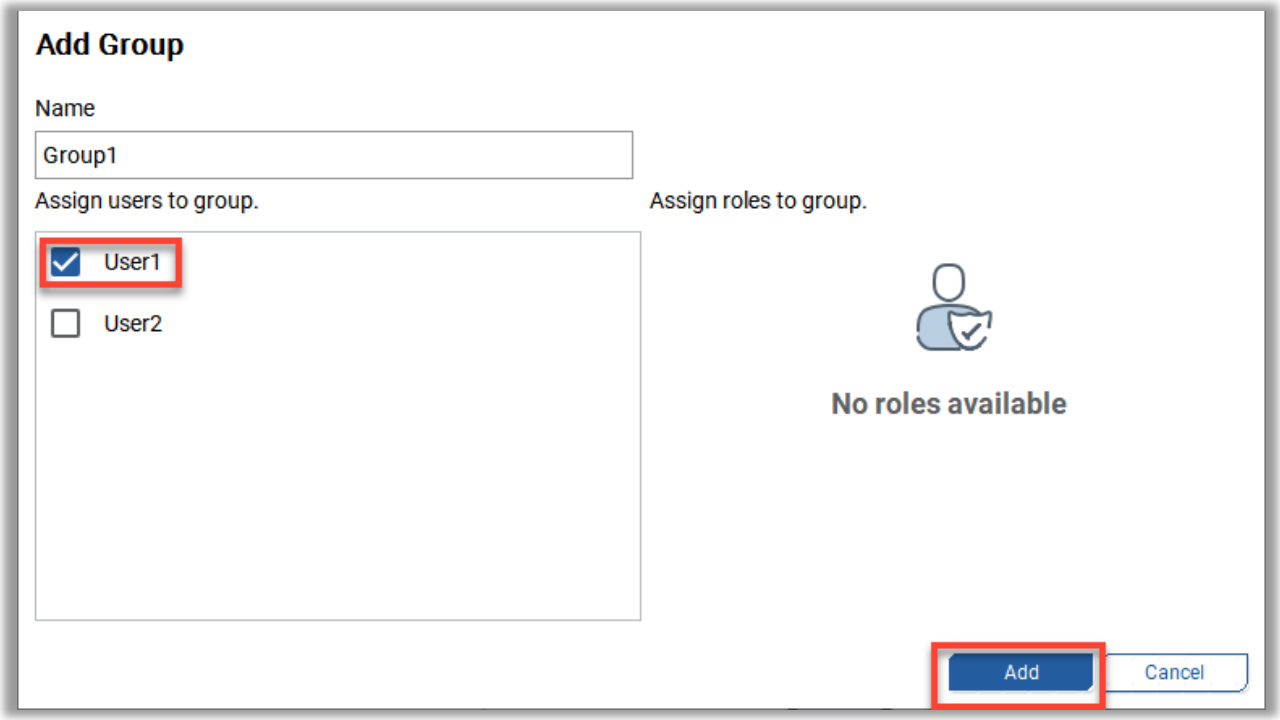
6. Click **Add** in the **Users** section to create **User2** and assign the following properties:
    - **Password** – "password2"
    - **Measurement system** – **US customary measurement system**

Next, you will create two groups.

7. In the **Groups** section, select **Add**

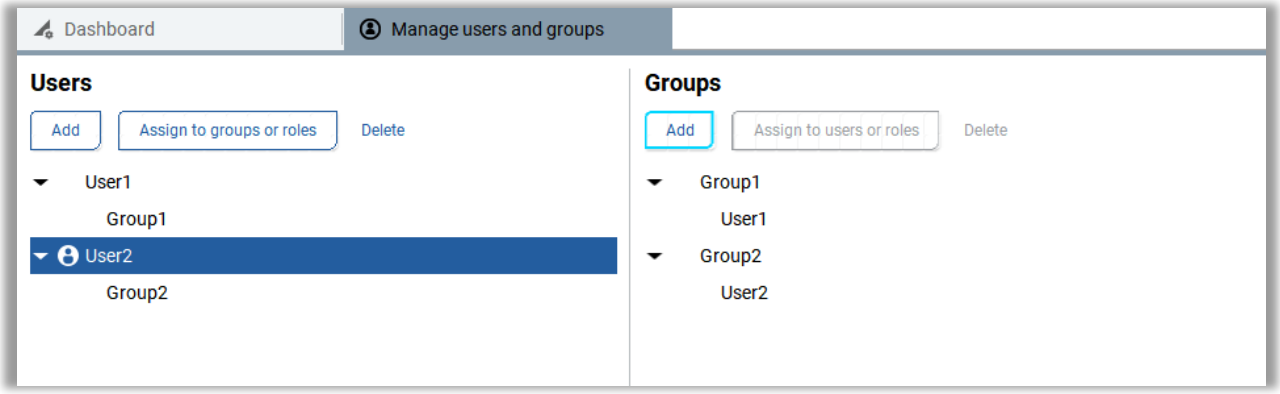


8. Check **User1** in **Assign users to group** then click **Add**.



9. Add another group and select **User2** in **Assign users to group** then click **Add.**

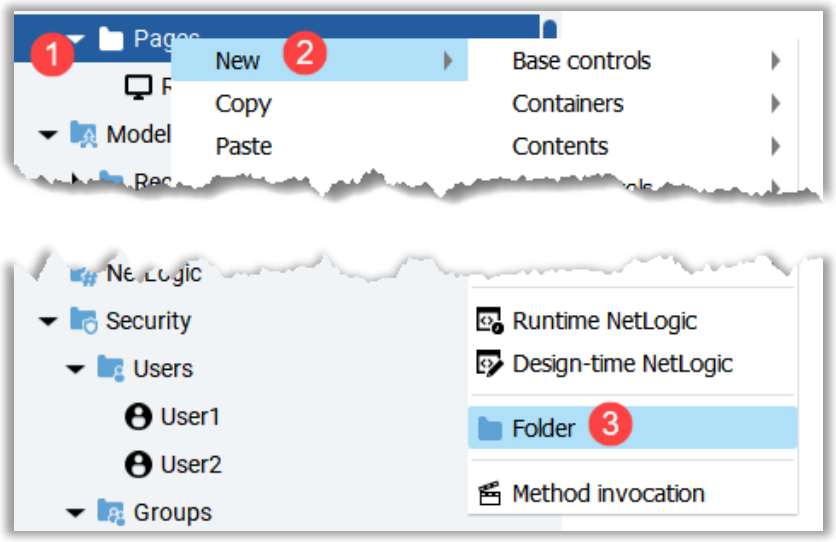
   The **Manage users and groups** tab should look like the following:



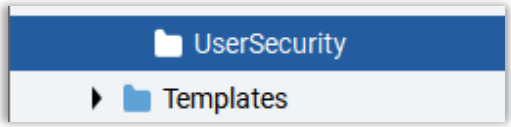
10. Close the **Manage user and groups** and the **Dashboard** screens by clicking the **X** on each tab.

Runtime user management

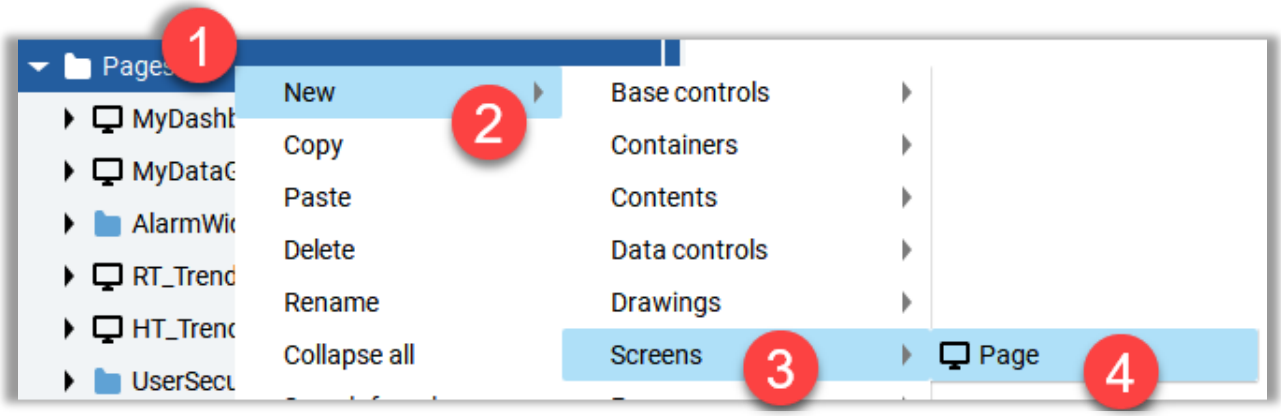
1. Right-click the **Pages** folder, click **New**, and click **Folder**.



2. Rename the folder to "UserSecurity".



3. In the **Project view** pane, right-click **UI > Pages > New > Screens > Page**.



4. Rename **Pagexx (type)** to "UserLogin".

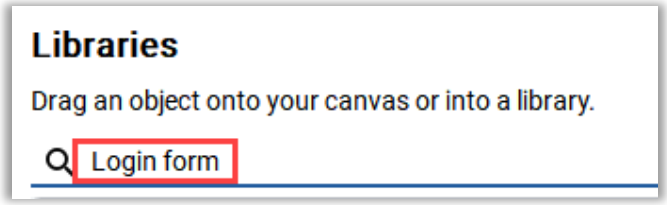
5. Copy **UserLogin (type)** to use for Alarm management in the next section. Right-click and select **Copy**, then right-click **Pages** and select **Paste**.

6. Rename **UserLogin1 (type)** to "UserManagement".

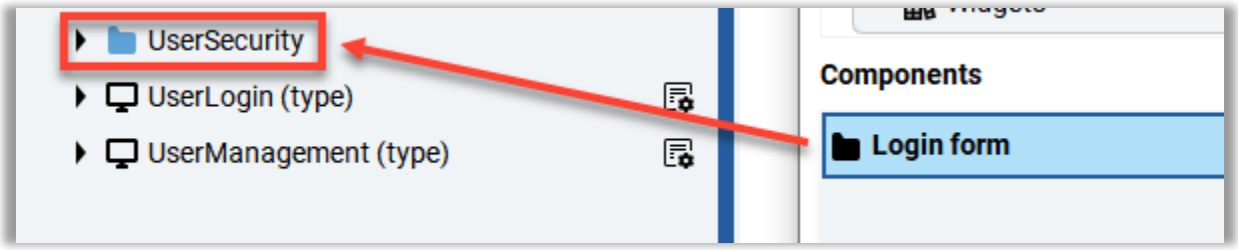
   Next, you will use the Template Libraries to add two pre-built items to your project.

7. Click the **Template Libraries** icon.

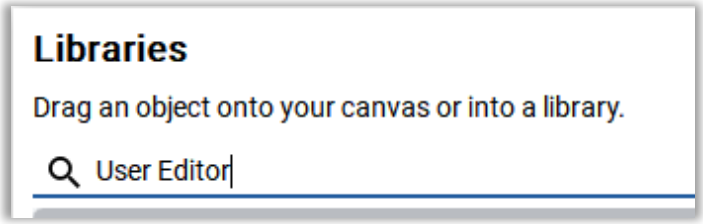
8. Type "Login form" in the **Libraries** search field.



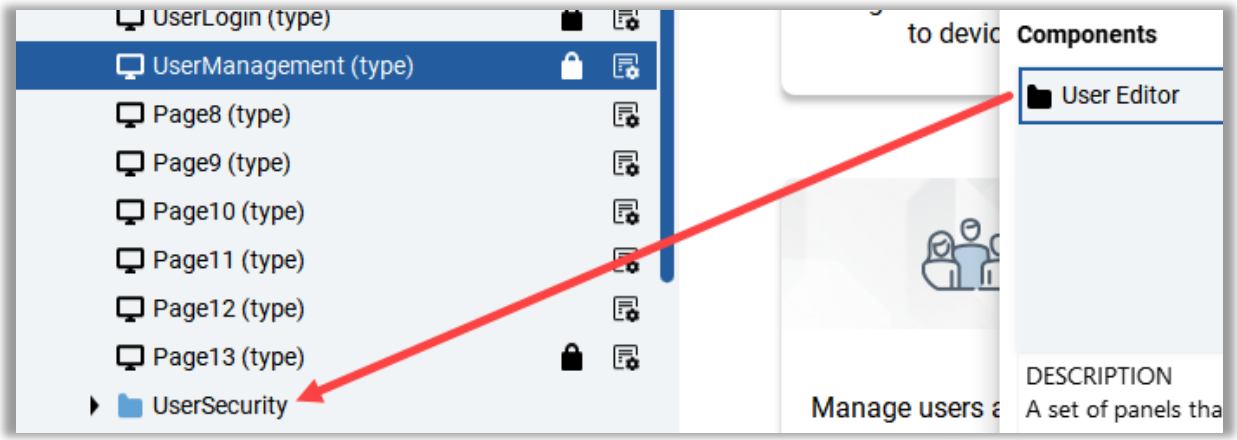
9. Click and drag the **Login form** folder to the **UserSecurity** folder.



10. Type "User Editor" in the **Libraries** search field.

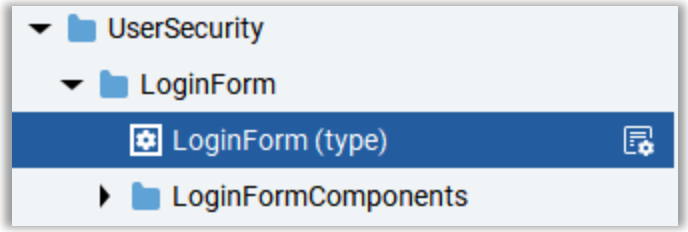


11. Click and drag the **User Editor** folder to the **UserSecurity** folder.

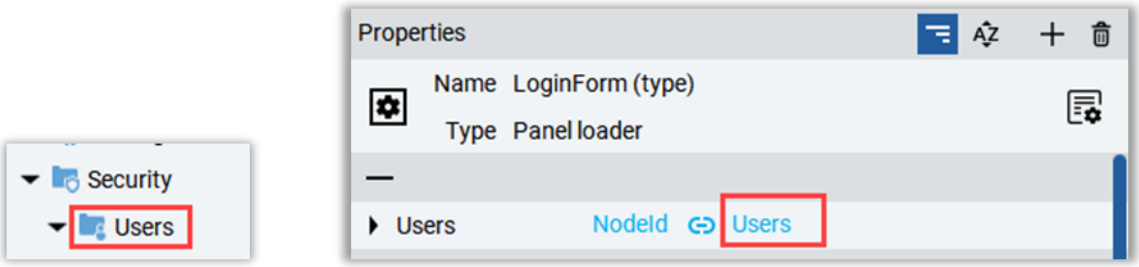


12. Close the **Template Libraries** pop-up by clicking the **Close** button.

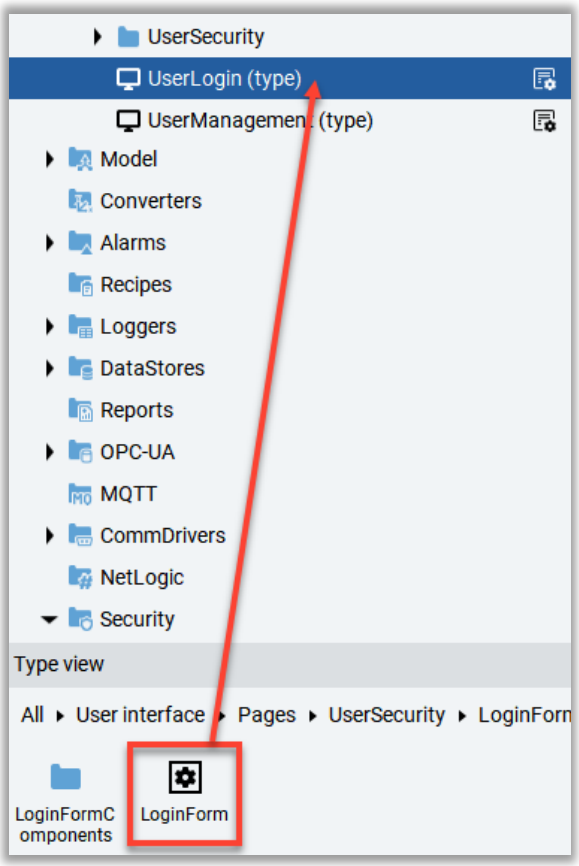
13. Expand **UserSecurity > LoginForm** and click **LoginForm(type)**.



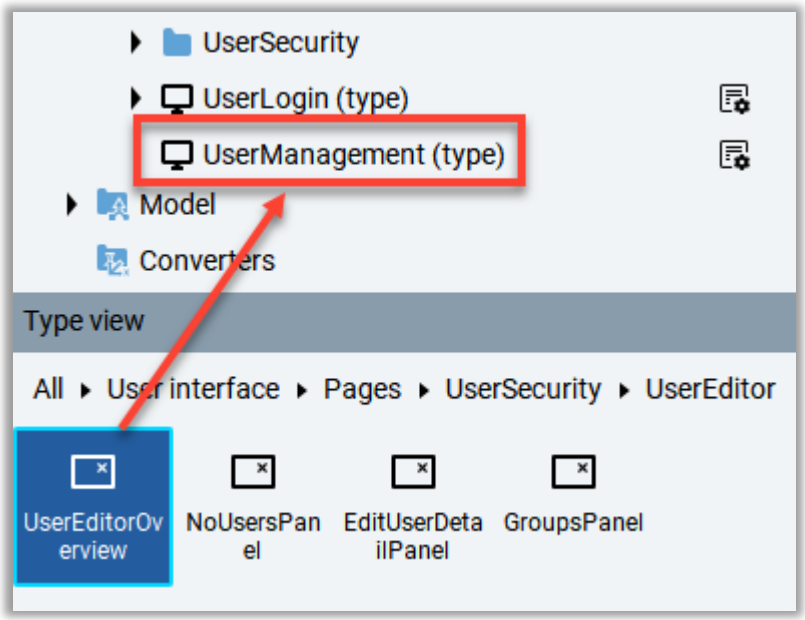
14. Drag and drop the **Users** folder from the **Security** node to the **Users NodeId** property.



15. Use **Type view** to add the **LoginForm** panel to the **UserLogin** screen by dragging it onto the screen from **User Interface > Pages > UserSecurity > LoginForm1**.

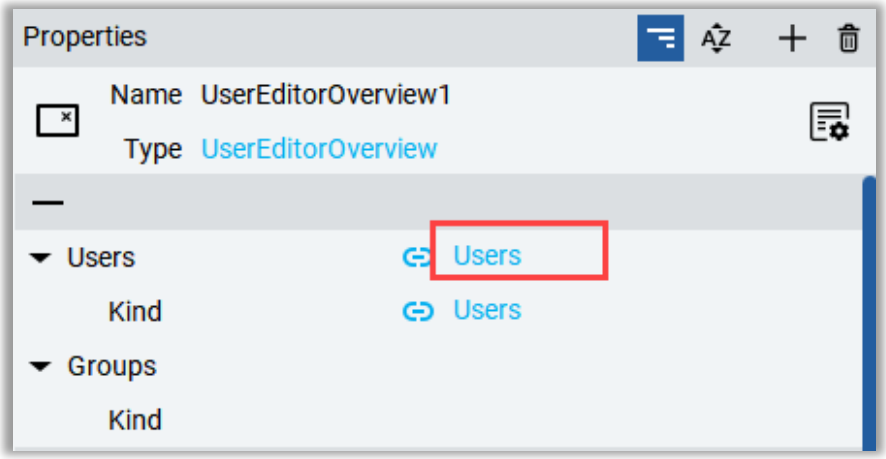


16. Use **Type view** to add the **UserEditorOverview** panel to the **UserManagement** screen by dragging it onto the screen from **User Interface > Pages > UserSecurity > UserEditor1**.

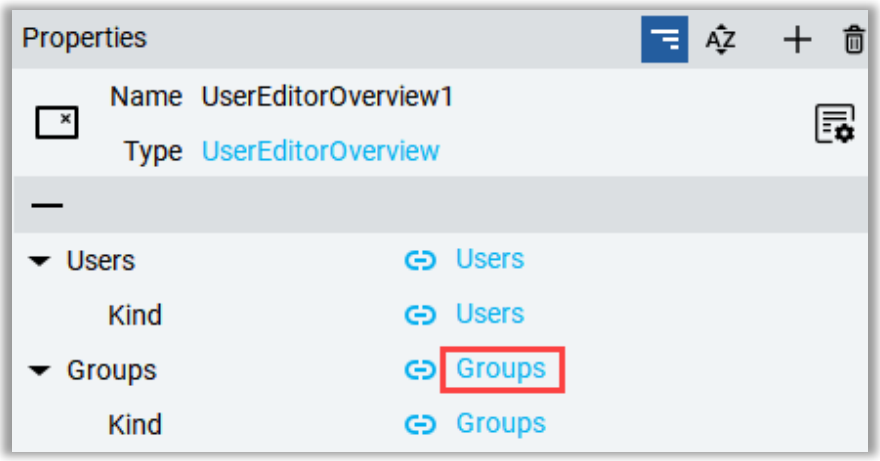


17. Expand **UserManagement** and select **UserEditorOverview1**.

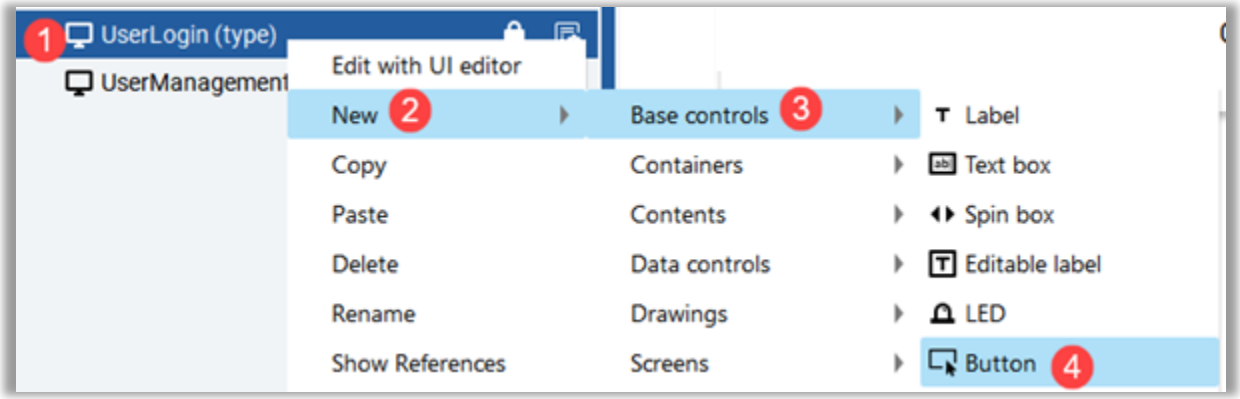
18. Drag and drop the **Users** folder in the **Security** node to the **Users** property of **UserEditorOverview1**. The **Kind** property will be automatically assigned.



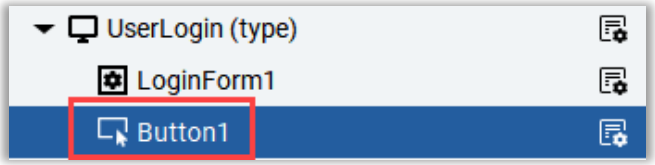
19. Drag and drop the **Groups** folder in the **Security** node to the **Groups** property of **UserEditorOverview1**. The **Kind** property will be automatically assigned.



20. In the **Project view**, right-click **UserLogin (type)**, click **New**, click **Base controls**, and click **Button**.

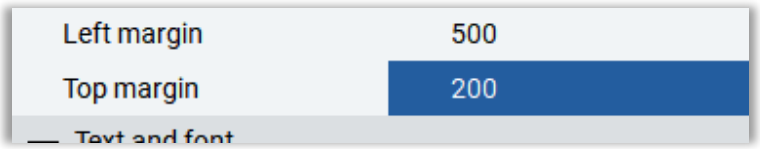


21. Click **Button1**.

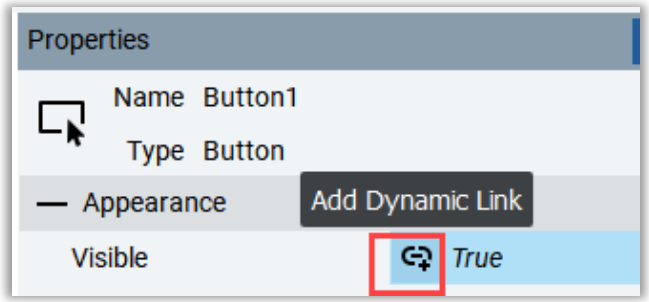


22. Change the **Text** value of the button to "EASY" and press **Enter**.

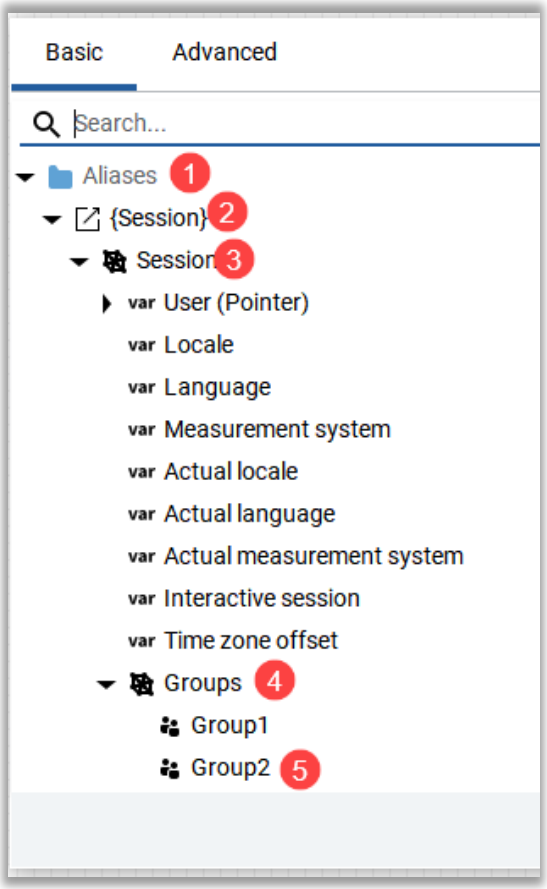
23. Change the **Left margin** to "500" and the **Top margin** to "200".

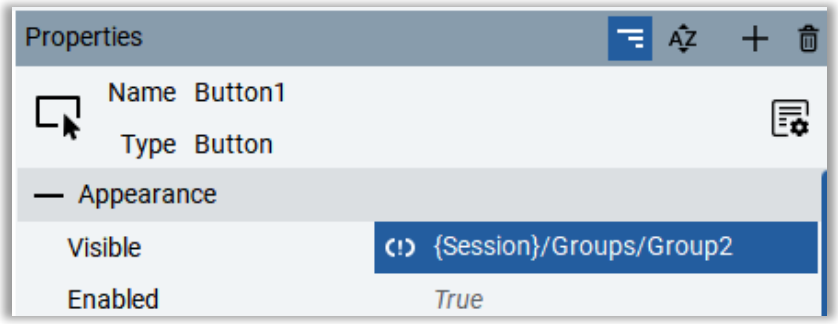


24. Click the **Add Dynamic Link** icon for the **Visible** property.



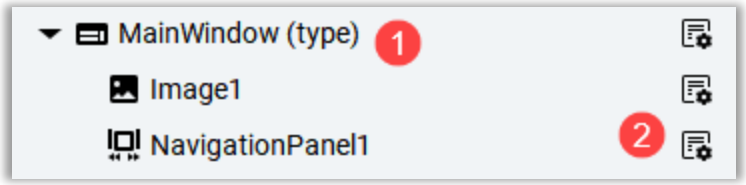
25. Expand the **Aliases** folder, expand **{Session}**, expand the **Session** object, expand the **Groups** object, click **Group2**, and click **Select**.



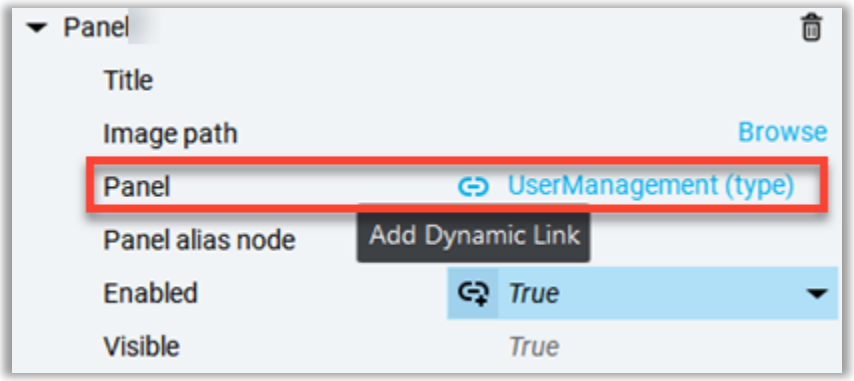


Any user who logs in and is a member of Group2 will have access to the EASY button.

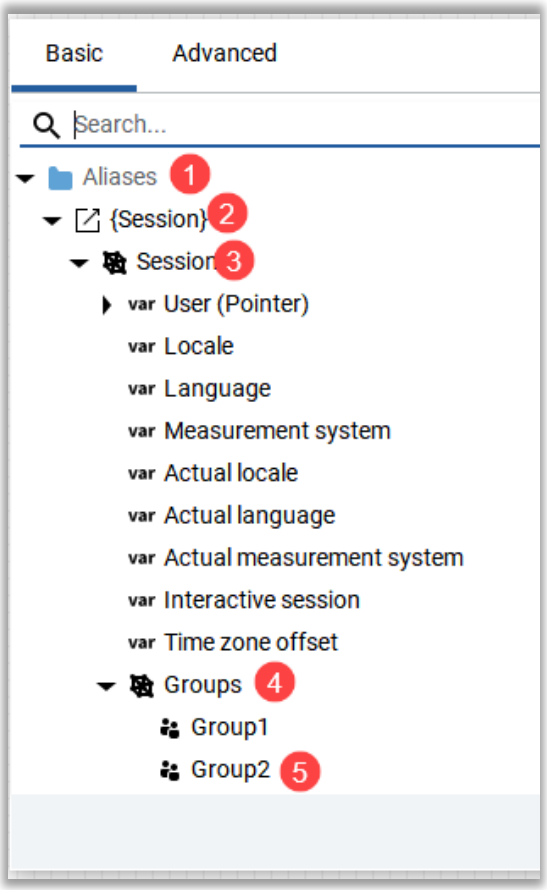
26. Select **UserLogin** and **UserManagement** and drag them to **NaviagationPanel1**.
27. In the **Project view**, expand **MainWindow**, and click the **NavigationPanel1** configuration icon.

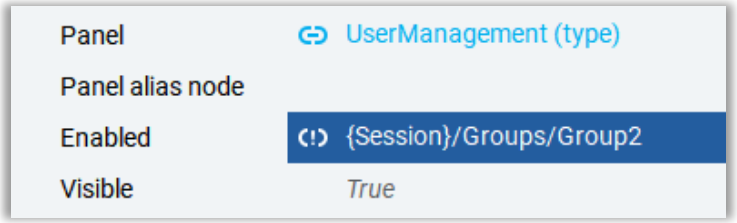


28. Scroll down and find the panel that contains the **UserManagement (type)** panel in the **Panel** Property.
29. Click the **Add Dynamic Link** icon for the **Enabled** property.



30. Expand the **Aliases** folder, expand **{Session}**, expand the **Session** object, expand the **Groups** object, click **Group2**, and click **Select**.





Any user who logs in and is a member of **Group2** will have access to the **User Management** panel.

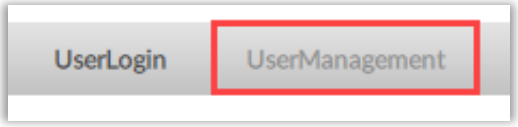
If performing navigation by using a button, this same Dynamic link can be applied to the **Enabled** or **Visible** property of the button.

Emulate and Explore

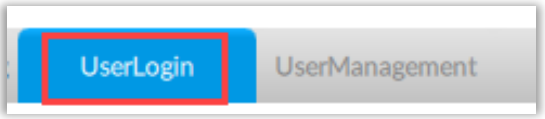
1. From the toolbar, click the **Run on Emulator** icon ▶.

2. Click the **UserManagement** tab on the Navigation panel.
3. Notice that the **UserManagement** panel cannot be navigated to.

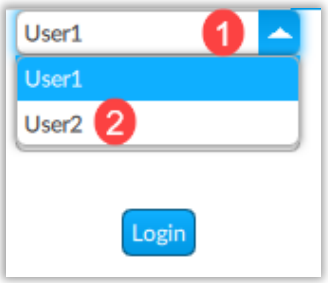
   In addition, the text is grayed out, indicating it is not enabled. This is because no user that is a member of **Group2** is logged in.



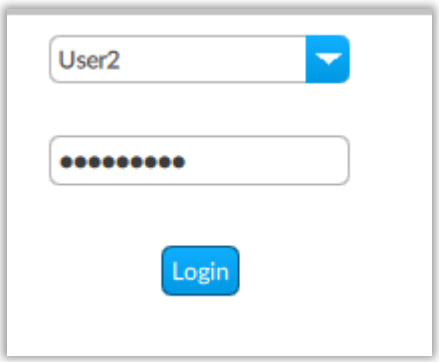
4. Click the **UserLogin** tab.



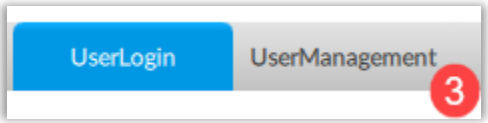
5. Click the **User** drop-down arrow button and click **User2**.



6. Enter “password2” in the **Password** box and press **Enter**.

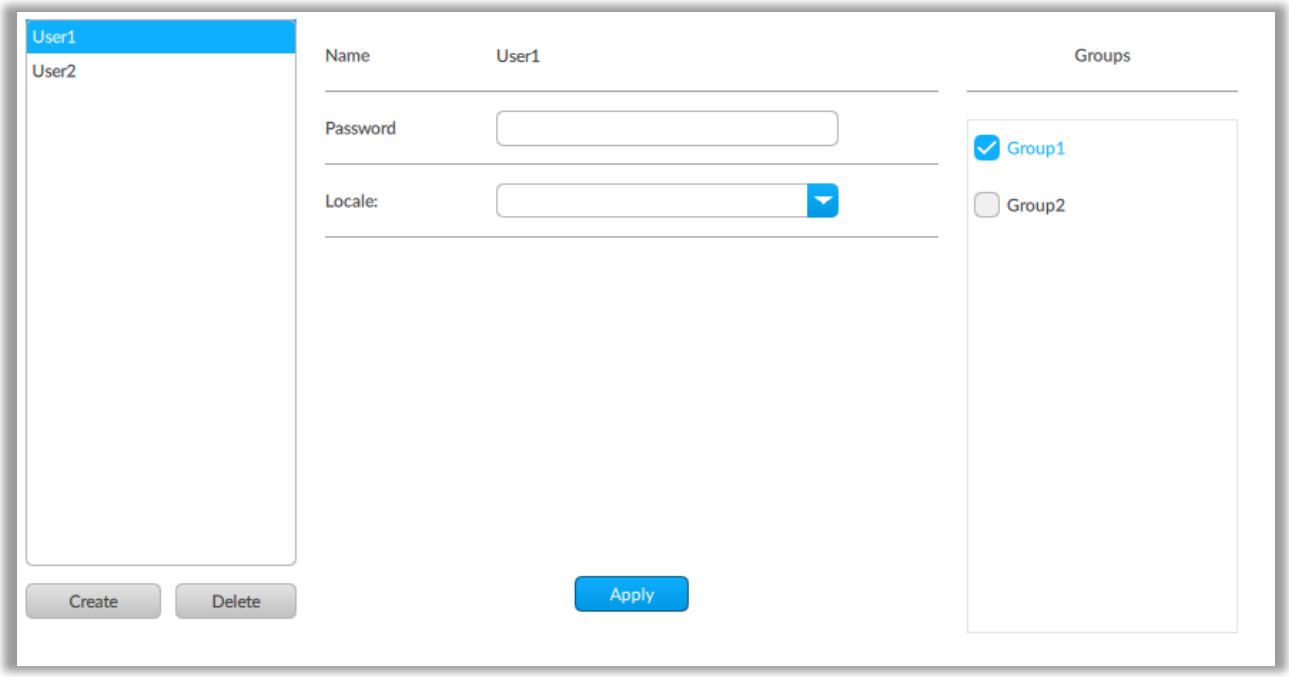


7. Click the **Login** button.

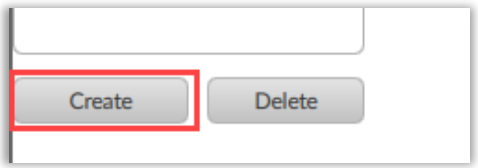


8. Observe the following:
   - **User2** is the **Current User**.
   - The **EASY** button is available for User2 since User2 is a member of Group2.
   - The **UserManagement** tab is available for User2 since User2 is a member of Group2.

9. Click the **UserManagement** tab on the Navigation panel.

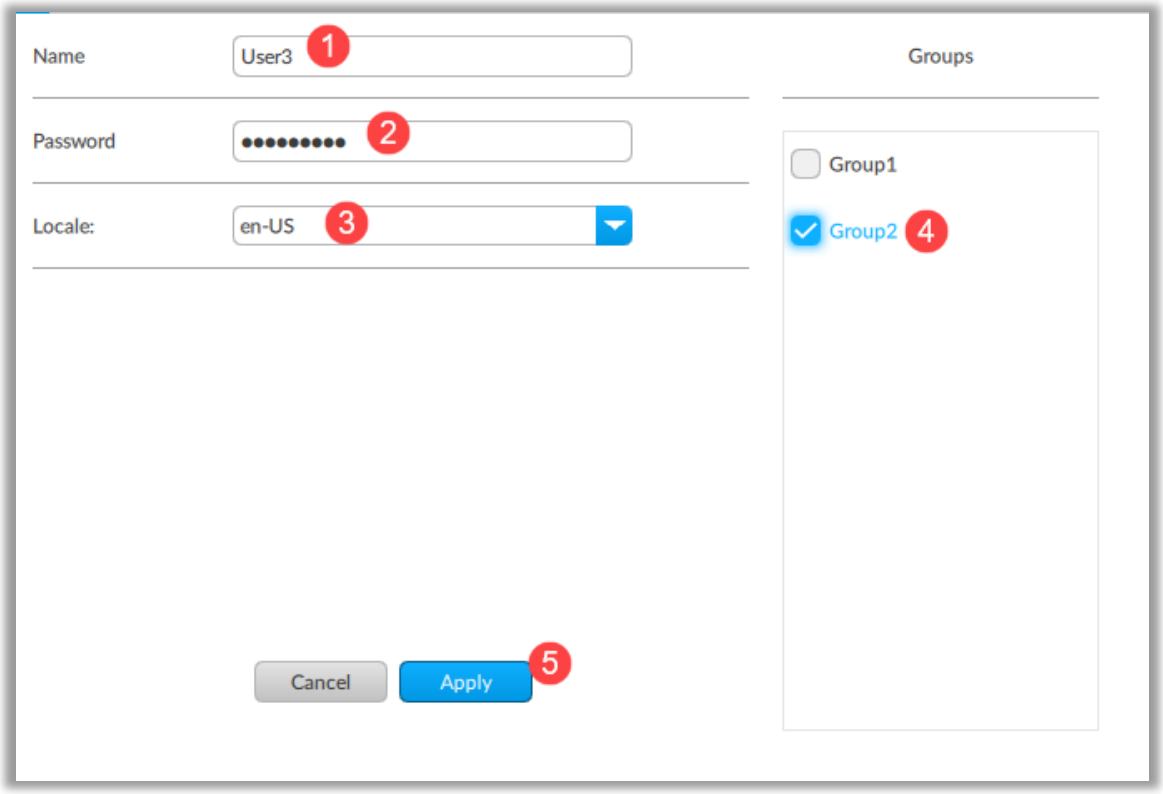


10. Click the **Create** button.



11. Assign the following to create a new user – press **Enter** after any text entry – and click **Apply** when finished.

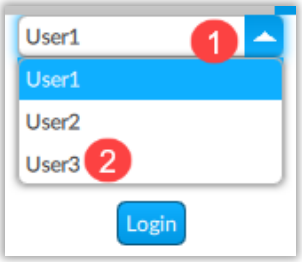
    - **Name** - "User3"
    - **Password** – "password3"
    - **Locale** – **en-US**
    - **Groups** – **Group2**



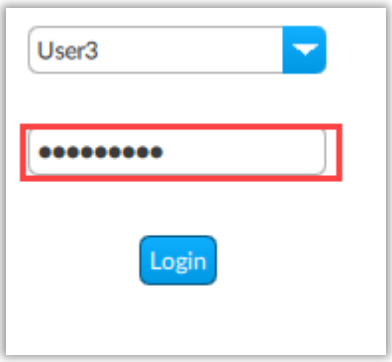
12. **User3** has now been created.

13. Click on the **UserLogin** tab in the Navigation panel.

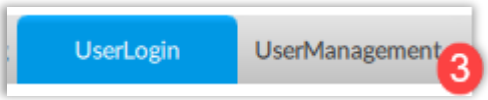
14. Click the **Logout** button.
15. Click the **User** drop-down arrow button and click **User3**.



16. Enter "password3" in the **Password** box and press **Enter**.



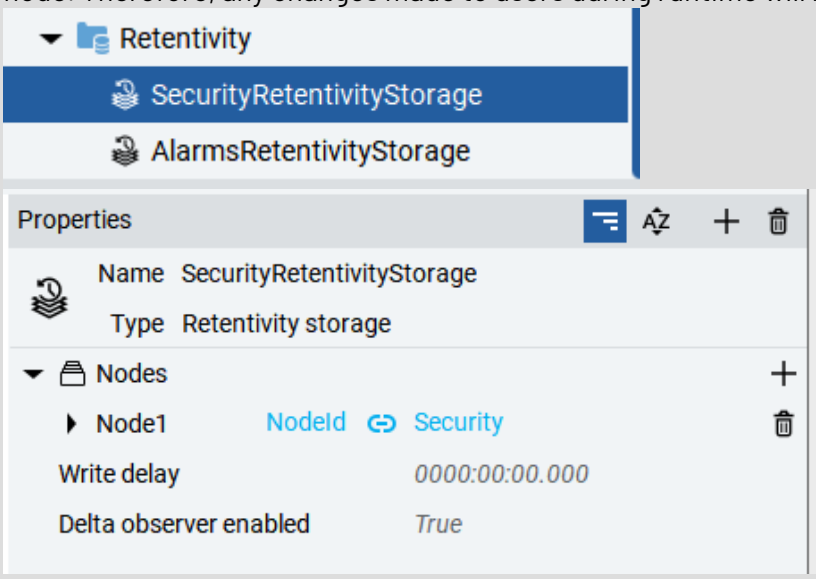
17. Click the **Login** button.



18. Observe the following:
    - **User3** is the **Current User**.
    - The **EASY** button is available for User3 since User3 is a member of Group2.
    - The **UserManagement** tab is available for User3 since User3 is a member of Group2.
19. Click the **Logout** button.

RETENTIVITY

There is a Retentivity folder that is part of any project by default. Also, by default, this folder contains a database called SecurityRetentivityStorage, which is assigned to the Security node. Therefore, any changes made to users during runtime will be retained.



20. Close the **Emulator**.
21. Close any open tabs in the Display canvas.

SUMMARY

FactoryTalk Optix provides the capability of creating local users and groups and/or using Windows domain accounts. When developing a project, designers can use the **Template Libraries** to quickly and easily add pre-built login and user management functionality. Finally, FactoryTalk Optix provides the ability to add locale information to individual users. Therefore, when a user logs into the application, the text, values, and engineering units will switch to the user's native language and measurement system.

If you would like to learn more about using Locales with Users, proceed to the next section, **Language Switching**.

Language Switching

Objectives

- Understand languages in FactoryTalk Optix.
- Configure localization dictionary and use translations in the project.
- Create buttons to switch languages during runtime.
- Learn how to create new dictionary entries from existing objects.
- Understand locales and learn how to change them during runtime.

Scenario

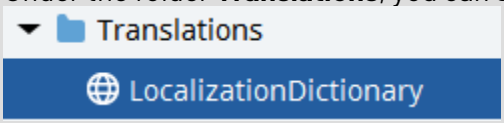
In this section of the lab, you will use the Localization Dictionary to add new translations and use them around the project. You will also be able to see how by using different locales the information on the screen will be displayed differently depending on the locale being used: language used, measuring system.

Add translations to the Localization Dictionary

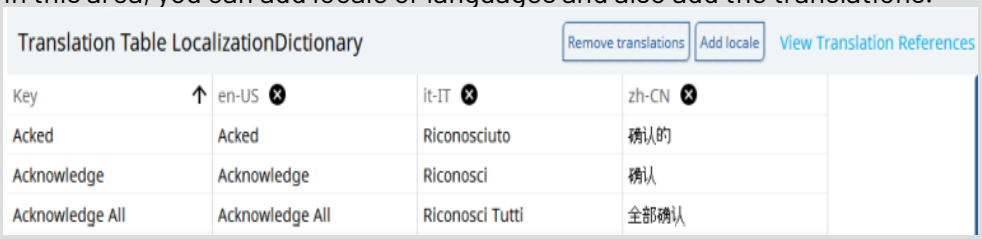
Note: Screenshots may differ depending on what optional sections have been completed previously.

LANGUAGES

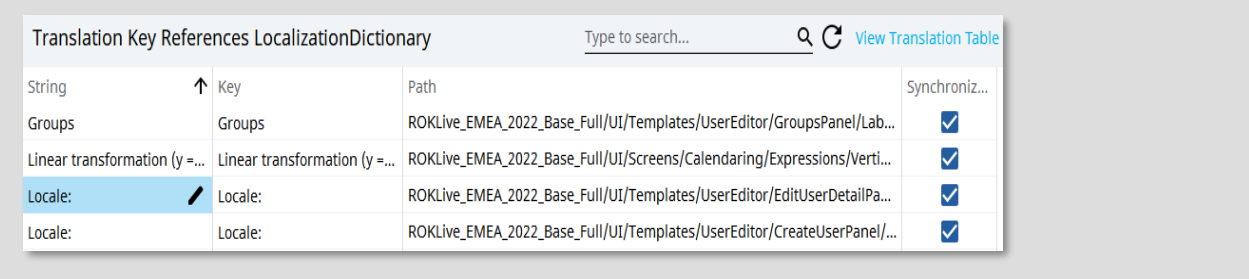
Read about Languages
Under the folder **Translations**, you can see the Localization Dictionary.



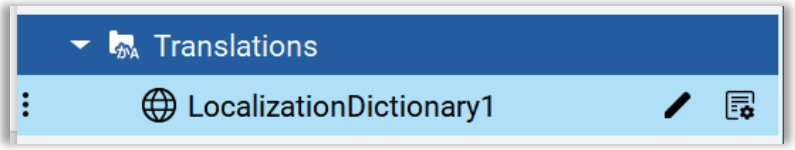
In this area, you can add locale or languages and also add the translations.



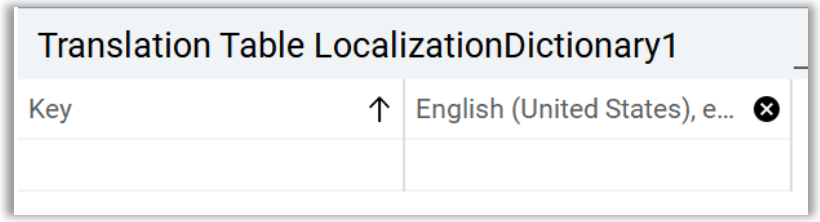
Every text field in the application gets a unique name called a Key which is the first column you see in the table.
When you click on **View Translation References**, you will see all the strings used in the application and that each string has a unique Key. So, for example, even if "Start" is used multiple times in your application, it will only be shown as one key, so you only need to translate it once.



1. In the **Project view**, expand **Translations** and double-click on **LocalizationDictionary1**.

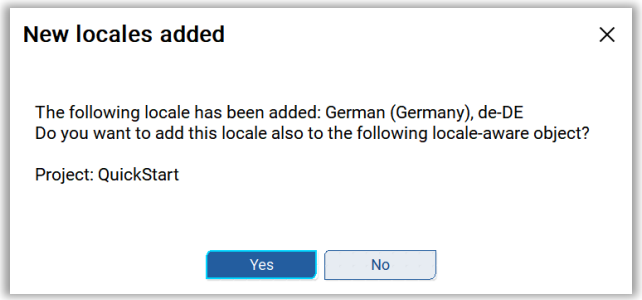


2. Notice that **English (United States)** locale is already available by default.



3. Click on **Add Locale**, choose **German (Germany) de-DE** from the drop-down list and add the selected locale:

4. You will get a confirmation message:



5. Click **Yes**.

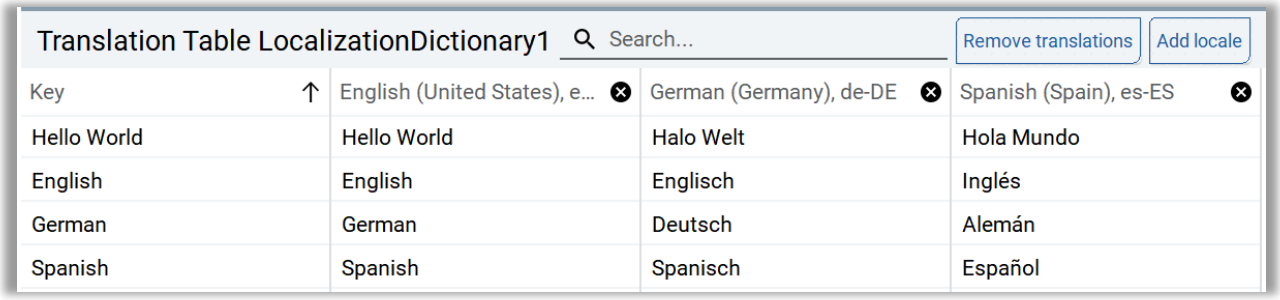
6. Now repeat steps 3, 4, and 5 to add **Spanish (Spain) es-ES**
7. On the first row, under **Key**, type "Hello World"
8. Fill in the columns of the first row (double-click on the cell or click the pencil icon to edit).

9. Under the **English (United States)** en-US, type "Hello World"
10. Under **German (Germany) de-DE**, type "Halo Welt"
11. Under **Spanish (Spain) es-ES**, type "Hola Mundo"

Key	English (United States), e...	German (Germany), de-DE	Spanish (Spain), es-ES
Hello World	Hello World	Halo Welt	Hola Mundo

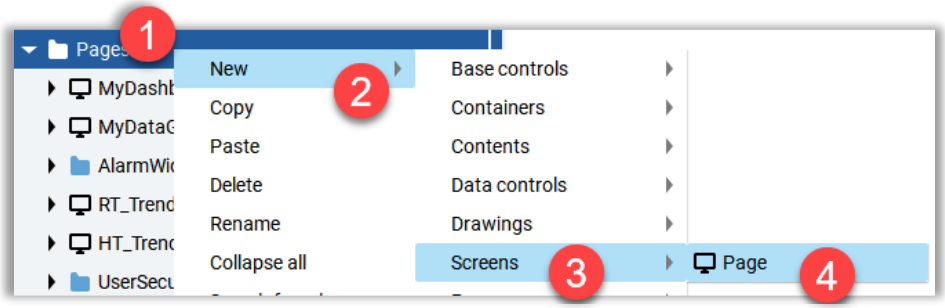
12. Do the same on the second row for Keys English, German and Spanish:
    Key: English
    English: English
    German: Englisch
    Spanish: Inglés **Note*: Press **Alt** + **130** to enter **é**
13. On the third row:
    Key: German
    English: German
    German: Deutsch
    Spanish: Alemán **Note*: Press **Alt** + **160** to enter **á**
14. On the fourth row:
    Key: Spanish
    English: Spanish
    German: Spanisch
    Spanish: Español **Note*: Press **Alt** + **164** to enter **ñ**

15. Your Localization Dictionary should look like the following:



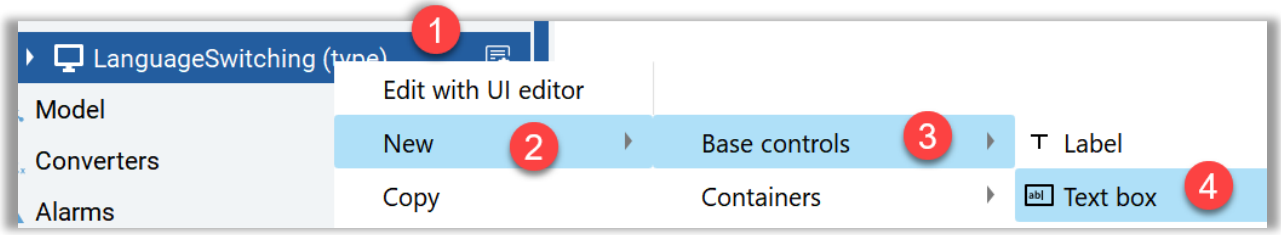
Create objects using translations from the Localization Dictionary

1. In the **Project view** pane, right-click **UI > Pages > New > Screens > Page**.

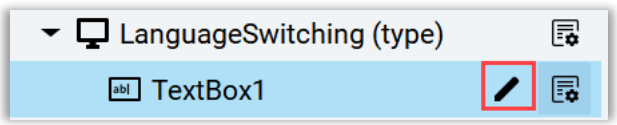


2. Rename **Pagexx (type)** to "LanguageSwitching".

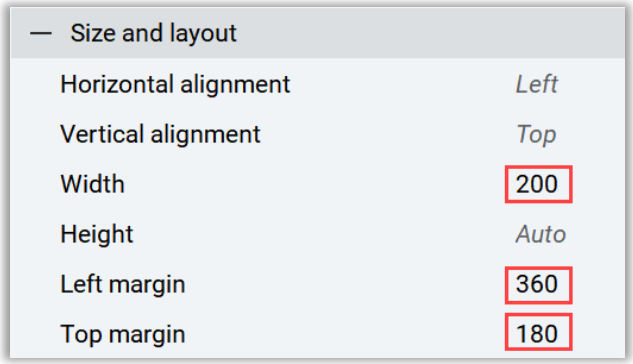
3. Right-click on **LanguageSwitching**, click **New > Base controls**, and click **Text box**.



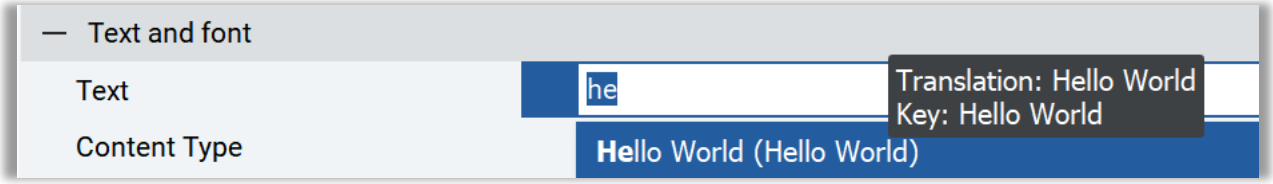
4. Rename it on the project view to "Message"



5. Set the following properties for this text box:
Width: 200
Height: Auto
Left Margin: 360
Top Margin: 180



6. On **Text** property: type "he" and a suggestion to use the key "Hello World" from the dictionary will come up as shown below. Press **Enter** (or double-click on it) to accept the suggestion.



7. Right-click on **LanguageSwitching**, and select **New> Base control > Button**.

8. Rename the button to "English".

9. Set the following properties for the **English button**:
Width: 50
Height: Auto
Left Margin: 140
Top Margin: 120

10. On property **Text**: type "en" and a suggestion to use the key "English" from the dictionary will come up as shown below. Press **Enter** (or double-click on it) to accept the suggestion.

11. Right-click **on** LanguageSwitching and select **New > Base control > Button.**

12. Rename the button to "German".

13. Set the following properties for the **German button**:
    Width: 50
    Height: Auto
    Left Margin: 210
    Top Margin: 120

14. On property **Text**: type "ge" and a suggestion to use the key "German" from the dictionary will come up as shown below. Press **Enter** to accept the suggestion.

15. Right-click on **LanguageSwitching**, and select **New > Base control > Button**.

16. Rename the button to "Spanish"

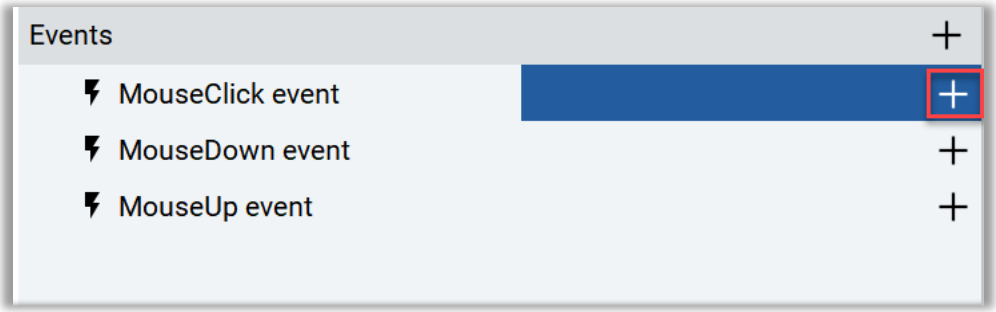
17. Set the following properties for the **Spanish button**:
    Width: 50
    Height: Auto
    Left Margin: 280
    Top Margin: 120

18. On property **Text**: type "sp" and a suggestion to use the key "Spanish" from the dictionary will come up as shown below. Press **Enter** to accept the suggestion.

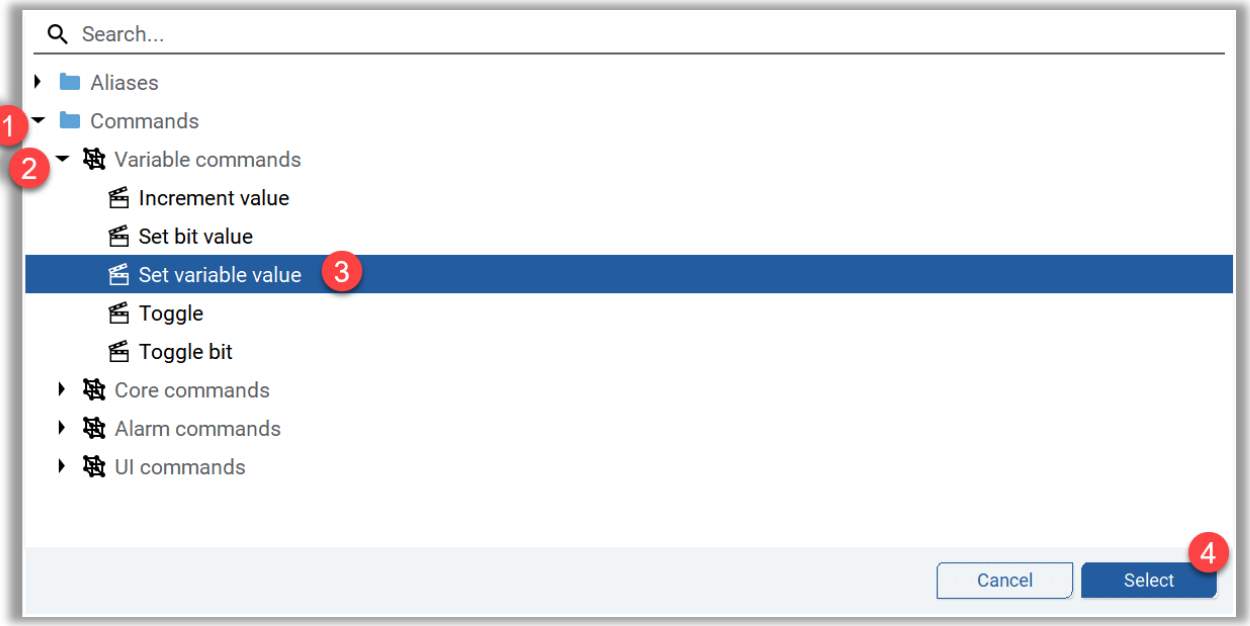
    Your screen should look like this:

Configure buttons to change the Application's language during runtime

1. Click the **English button**. On **Properties**, click the **+** symbol next to **MouseClick event**:

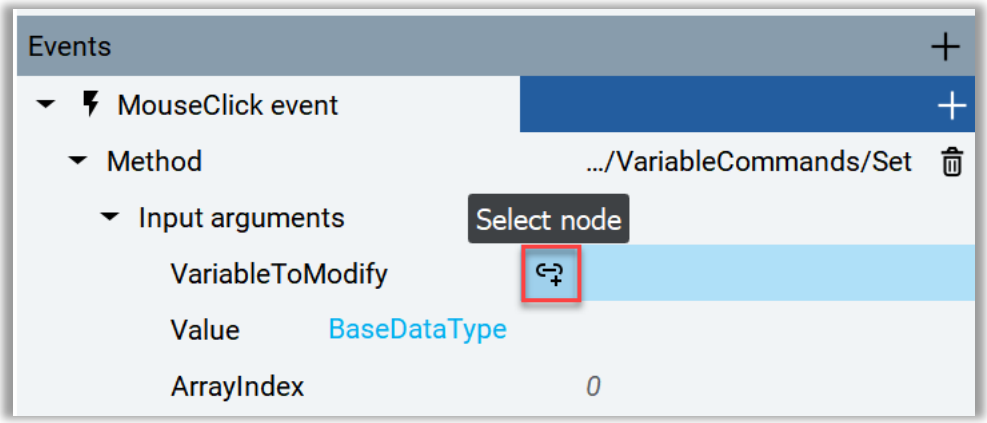


2. Expand **Commands**, expand **Variable commands**, select **Set variable value**, and then click **Select**.

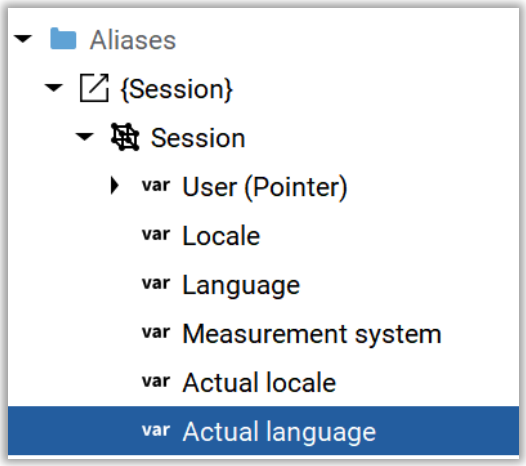


3. A method will appear under the **Mouse click event** field.

4. Click on the **dynamic link** icon ( ) next to **VariableToModify**

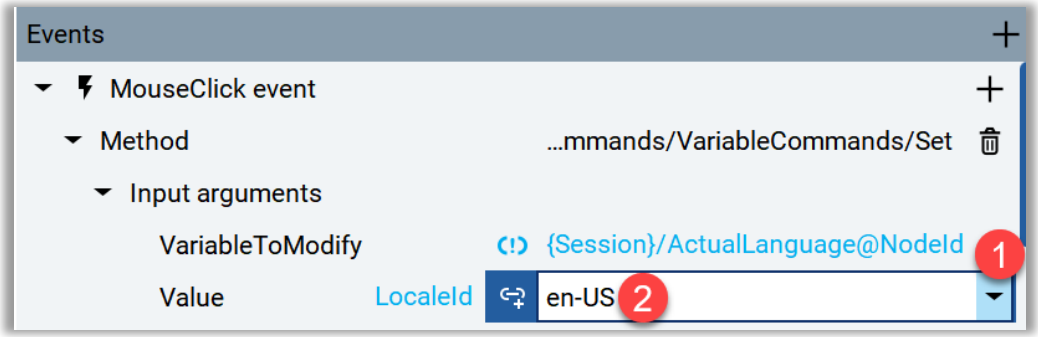


5. Navigate to language: **Aliases > {Session} > Session** and select **Actual Language**.



6. Click **Select**.

7. On the property **Value**, select the locale that we want to change the application to when pressing the button. In this case **en-US**:



8. Click on the **German button**. On properties, click the **+** symbol next to **MouseClick event**

9. Expand **Commands**, expand **Variable commands**, select **Set variable value**, and then click **Select**

10. A method will appear under **MouseClick event**.

11. Click on the **dynamic link** icon next to **VariableToModify.**

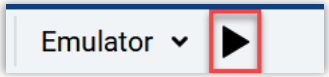
12. Navigate to language: **Aliases > {Session} > Session** and select **Actual Language** and click **Select**.

13. On the property **Value**: Select the locale we will want to change the application to when pressing the button. In this case **de-DE**:

14. Click on **Spanish button**. On properties, click the **+** symbol next to **MouseClick event.**
15. Expand **Commands**, expand **Variable commands**, select **Set variable value**, and click **Select**.
16. A method will appear under **MouseClick event**.
17. Click on the dynamic link icon ( ) next to **VariableToModify**.
18. Navigate to language: **Aliases**->**{Session}**->**Session** and select **Actual Language** and click **Select.**
19. On the property **Value**: Select the locale we will want to change the application to when pressing the button. In this case **es-ES**.

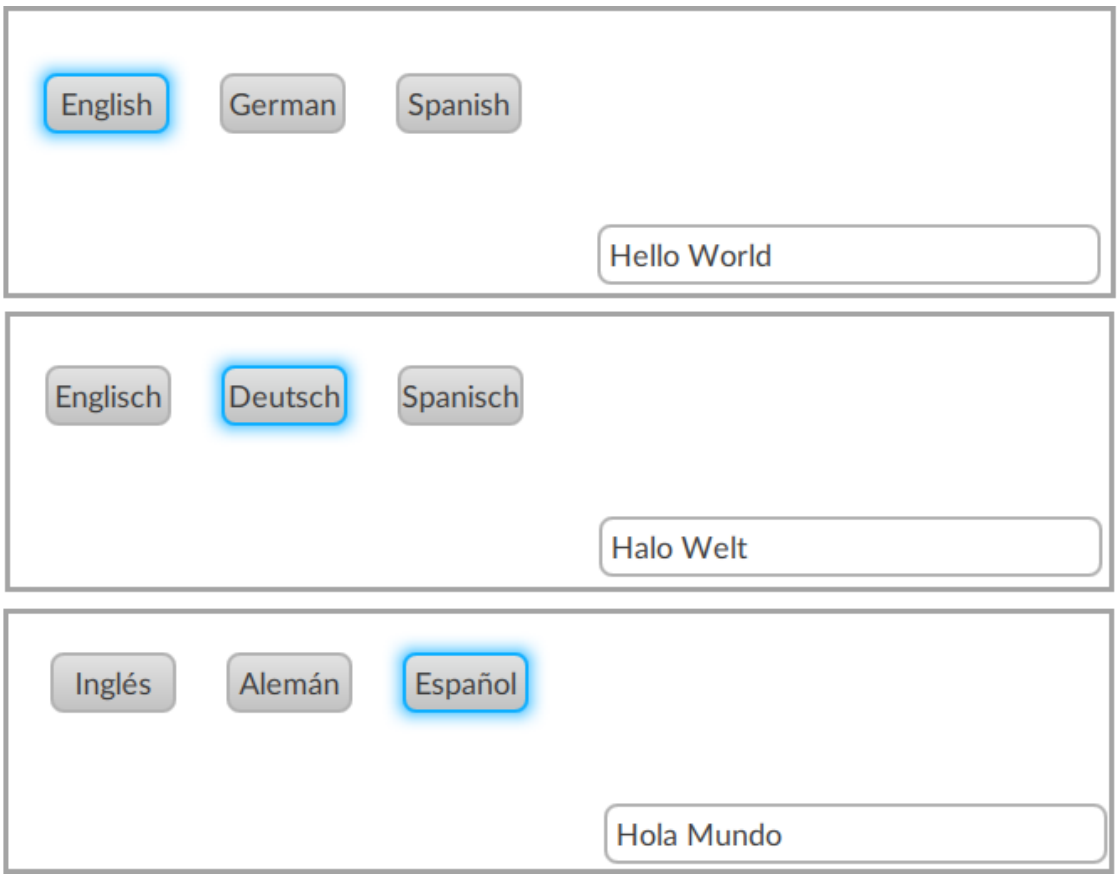
Let's test this screen:

20. Click the play button next to the emulator.



21. Navigate to **LanguageSwitching**.

22. Test that the buttons you just created change the language of the text box and the text of the buttons:



Create translations using text from existing objects

In this section, we are going to pretend that we already had a project containing objects with some text. We want to include these texts in the localization dictionary. For this, we are going to create two buttons: one to open and one to close a valve.
To keep this section short, as all we want to see is that the text is translated, we won't add actions to these buttons.

1. Right-click the **LanguageSwitching** screen, click **New**, **Base controls,** and then click on **Button**.

2. Rename it to "OpenValve".

3. Change the following properties of OpenValve:
   Width: 100
   Height: Auto
   Left Margin: 360
   Top Margin: 210
   Text: Open Valve

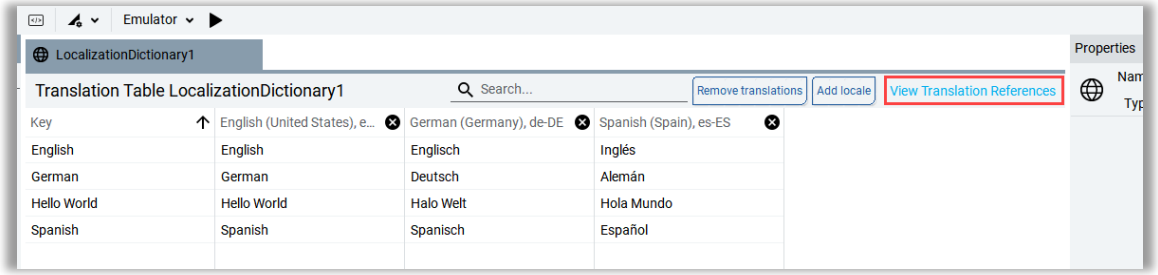
4. Right-click the **LanguageSwitching** screen, click **New**, **Base controls** and then click on **Button**.

5. Rename it to "CloseValve".

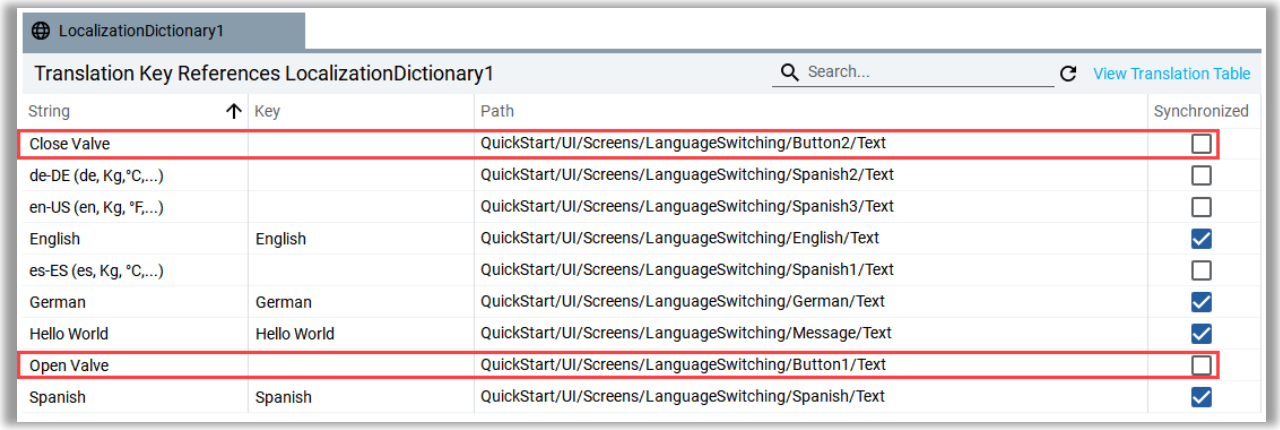
6. Change the following properties of CloseValve:
   Width: 100
   Height: Auto
   Left Margin: 360
   Top Margin: 240
   Text: Close Valve

7. In the **Project view**, expand Translations and double-click on **LocalizationDictionary1**.

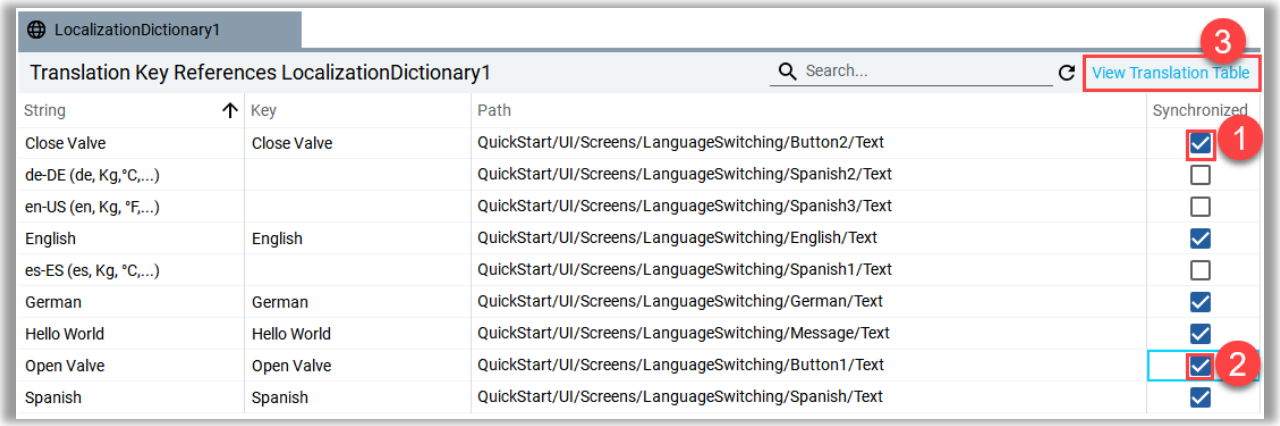
8. Click on **View Translation References**.



9. Find "Open Valve" and "Close Valve" on the list of references:



10. Select both keys on the Synchronized column and click on **View Translation Table**.



11. Notice that these two keys have been added to the translation table.

12. Add translations for these new entries in the Translation Table:

    For key "Open Valve":

    de-DE: Ventil öffnen

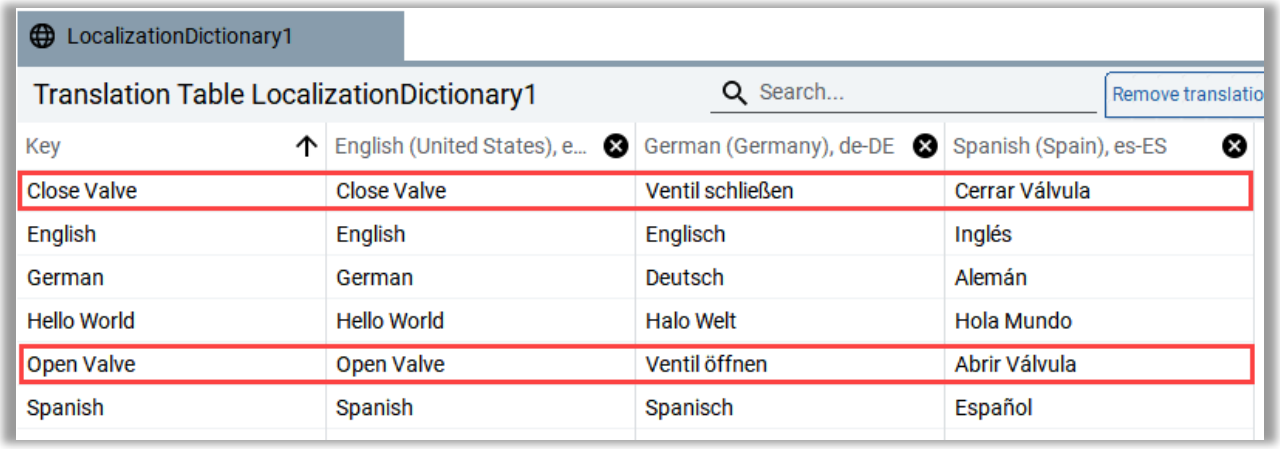
    es-ES: Abrir Válvula

    For key "Close Valve":

    de-DE: Ventil schließen

    es-ES: Cerrar Válvula

13. The translation table should look like the following:



LOCALES

Read about Locale

Introduction

The term locale means the set of display settings of a user interface based on language and country. It is represented by a label called locale ID, made up of language and country (e.g., en-US, en-UK, it-IT etc.).

A project can support multiple locales. The locales supported by the project are set in the Locales property of the project node.

In multilingual projects, the Presentation engine displays the interface based on the session locale, if configured, and the texts displayed based on the translations available in the LocalizationDictionary object.

Locale IDs

The Locale ID specifies a language and a country, e.g. en-US, en-UK, it-IT. In particular, the second segment determines the date and time format, the date separator and the measurement system (International Measurement System, United States Customary System or British Imperial System).

Session locale

The session locale determines the locale of the user interface (i.e. the translation of the texts according to the project settings), the data display format, and the conversion of all values according to the required measurement system. It is set at runtime based on the user or object locale UI Session Session UI.

Note

If there are no locale settings at the user or Session UI object level, the session locale is set based on the locales supported by the project: in particular, the first locale in order of writing in the Locales property has priority (see Fallback locale).

Fallback locale

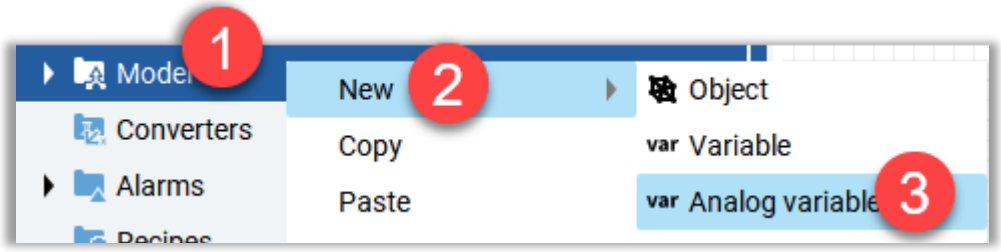
If the user-determined session locale is not configured in a project, or if translations of some interface texts are not available for the session locale, the texts are displayed in the project fallback locales, configured in the Fallback locales for translations property of the project node.

Note

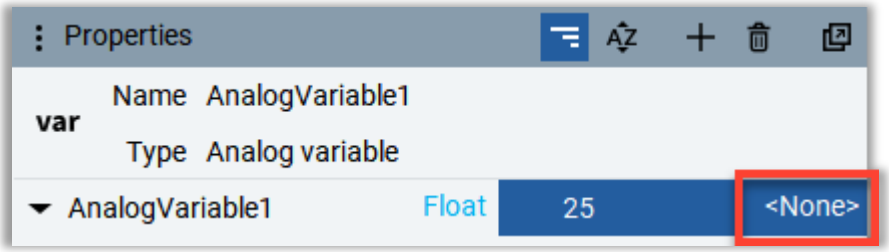
When there are multiple fallback locales, at runtime the system uses the fallback locale based on the order of insertion of the different locales in the Fallback locales for translations property. If the list of fallback locales contains the locale not supported by the project, this fallback locale takes priority over the others.

Create Analog items and display values according to measurement system

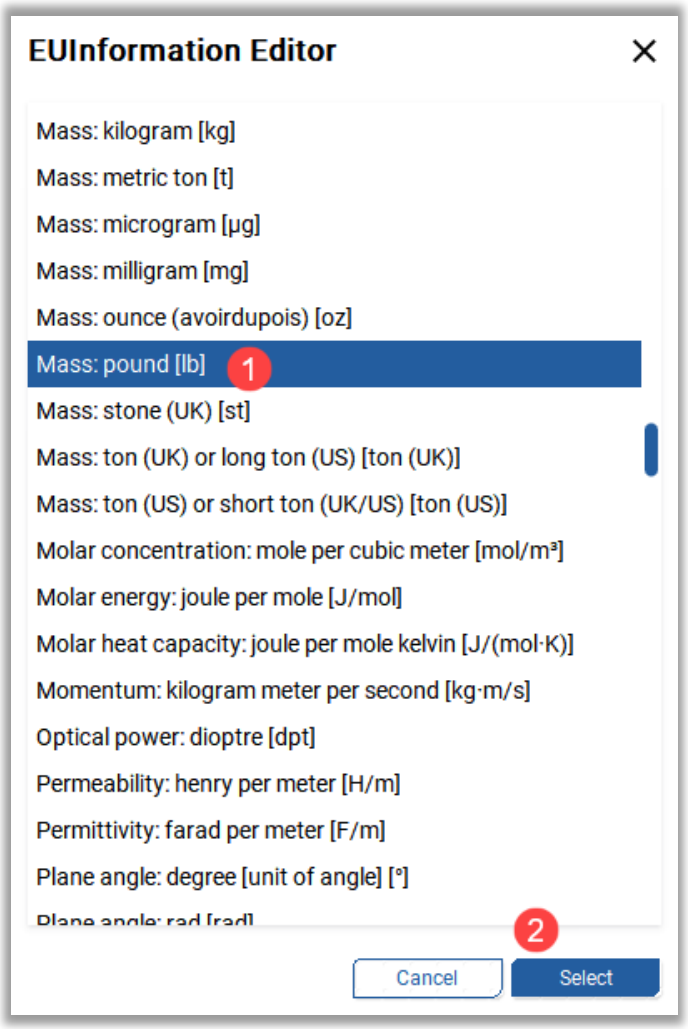
1. Right-click the **Model** folder, click **New**, and click **Analog variable**.



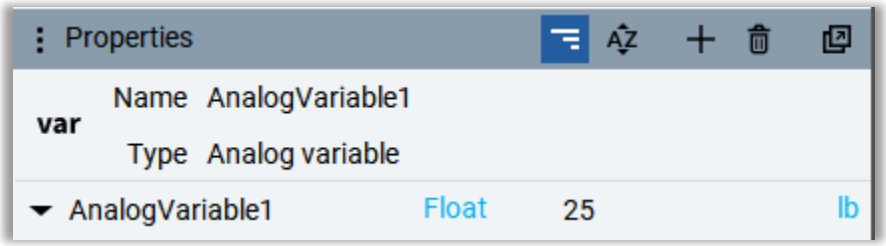
2. In the **Properties** window, next to **AnalogVariable1 > Float**, change the value from "0" to "25".
3. Click on **<None>** to bring up the **EUInformation Editor**.



4. Scroll down, click **Mass: pound[lb]**, and click **Select**.



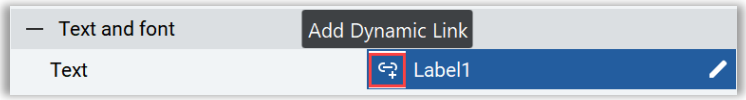
5. The properties of **AnalogVariable1** should look like this:



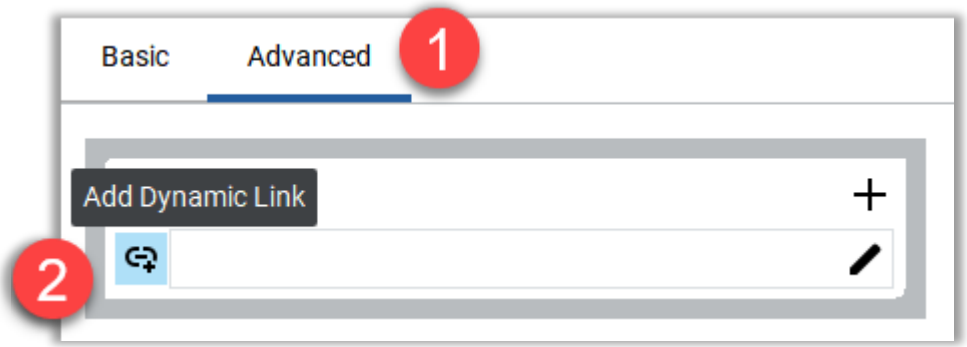
6. From the **Project view**, right-click **LanguageSwitching**, click **New**, click **Base controls**, and then click **Label**.

7. Change the **Left margin** to “360” and the **Top margin** to “275”.

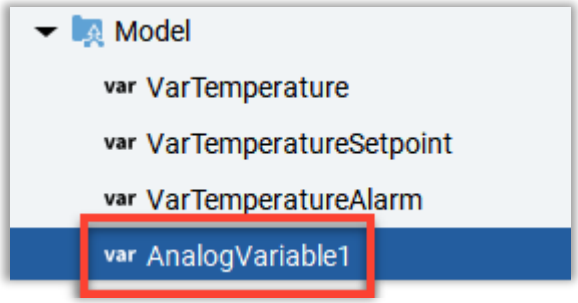
8. Click the **Add Dynamic Link** icon for the **Text** property.



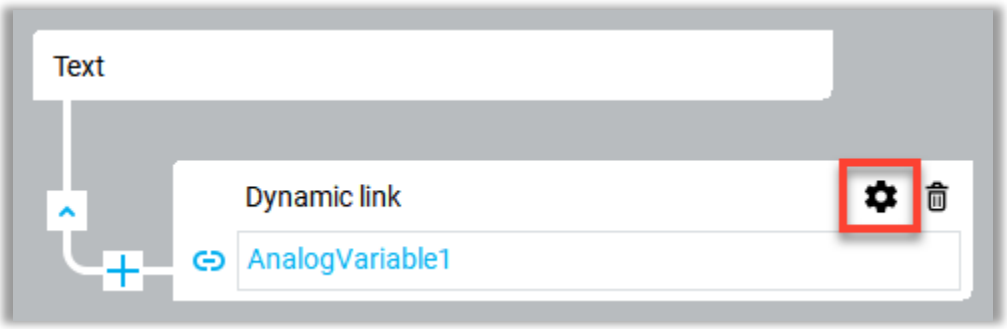
9. Click the **Advanced** tab and click on the **Dynamic Link**



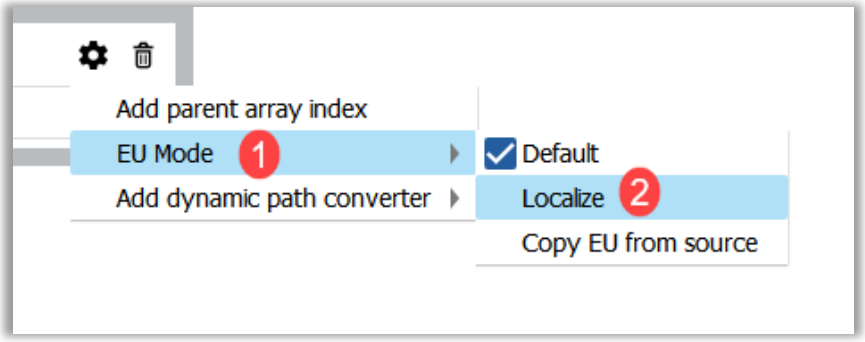
10. Scroll down and expand **Model**, click on **AnalogVariable1** and click **Select.**



11. Click the **Gear** icon.



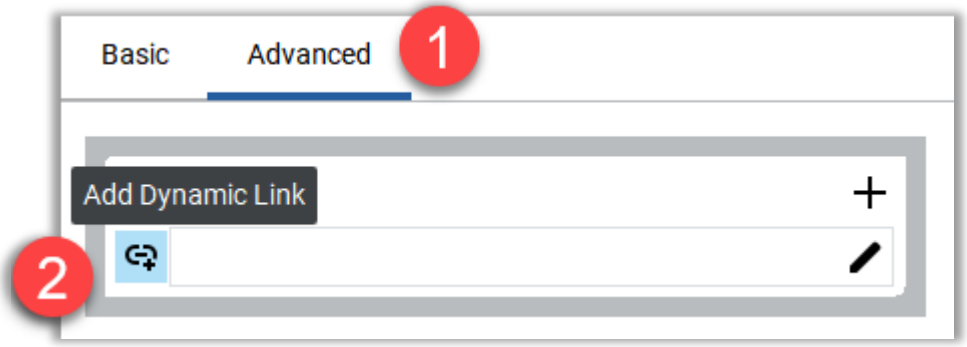
12. Click **EU Mode** and then select **Localize**.



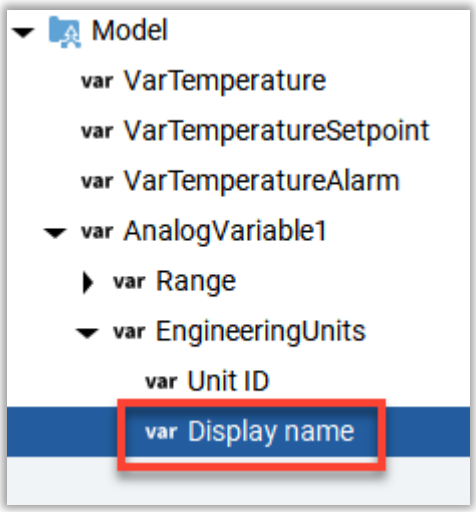
This option allows the value to be converted to different engineering units.

13. **Close** the Add Dynamic Link pop-up.
14. From the **Project view**, right-click **LanguageSwitching**, click **New**, click **Base controls**, and click **Label**.
15. Change the **Left margin** to "420" and the **Top margin** to "275".
16. Click the **Add Dynamic Link** icon for the **Text** property.

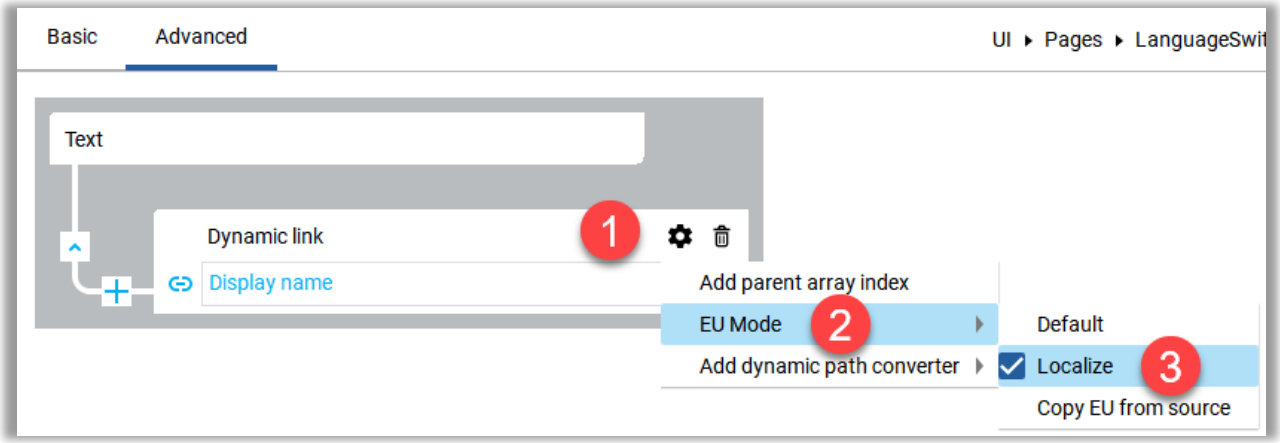
17. Click the **Advanced** tab and click on the **Dynamic Link**.



18. Expand the **Model** folder, expand **Analogitem1**, expand **EngineeringUnits**, select **DisplayName**, and click **Select**.



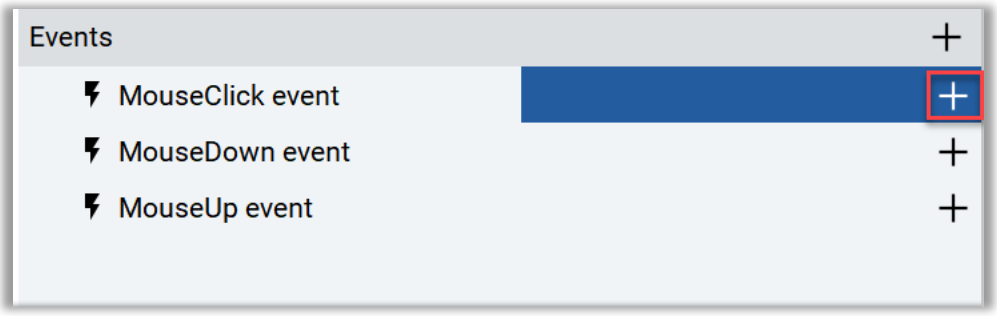
19. Click the **Gear** icon, expand **EU Mode** and then click **Localize.**



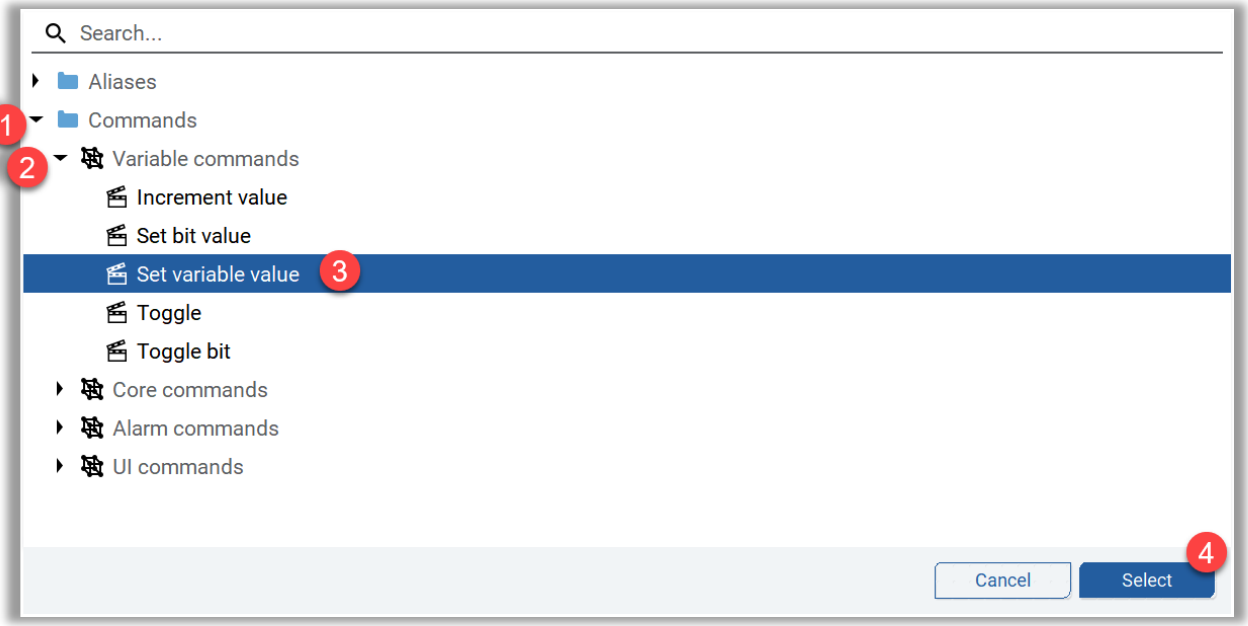
20. Close the **Add Dynamic Link** pop-up.

Create buttons to change locale during runtime

1. Right-click the **LanguageSwitching** screen, click **New > Base controls** and then click on **Button**.
2. Rename this button "en-US".
3. Set the following properties for this button:
   Width:120
   Height: Auto
   Left Margin: 600
   Top Margin: 120
   Text: en-US (en, lb)
4. Click on the **en-US** button. On properties, click the **+** symbol next to **MouseClick Event**:

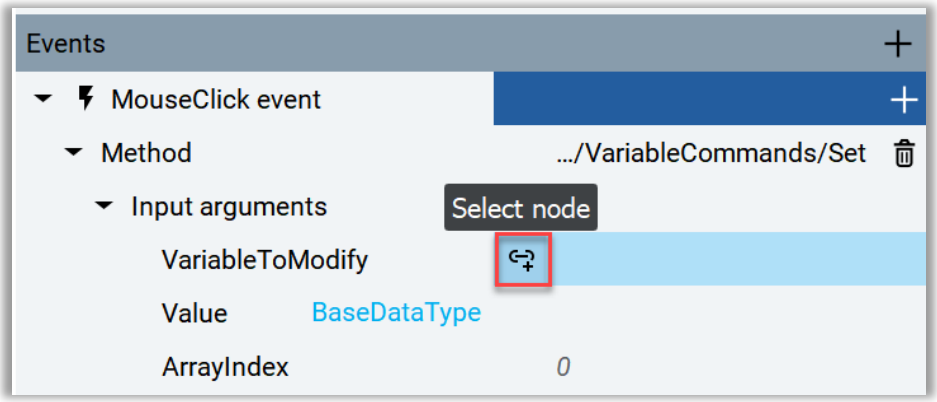


5. Expand **Commands**, expand **Variable commands**, select **Set variable value**, and click **Select**.

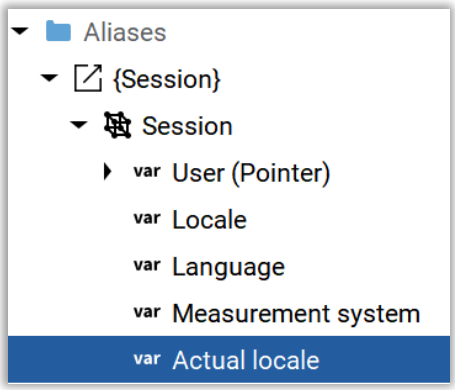


6. A method will appear under **MouseClick event**.

7. Click on the **dynamic link** icon ( ) next to **VariableToModify**.



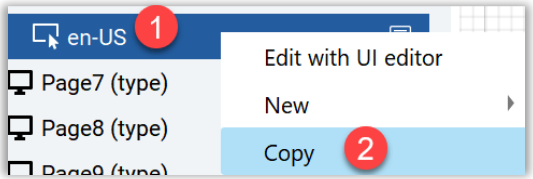
8. Navigate to: **Aliases > {Session} > Session** and select **Actual Locale**.



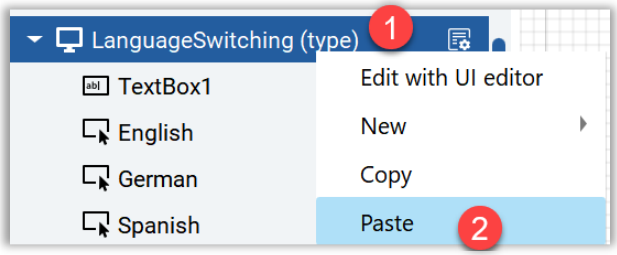
9. Click **Select**.

10. For the property Value: select the locale we will want to change the application to when pressing the button. In this case "en-US":

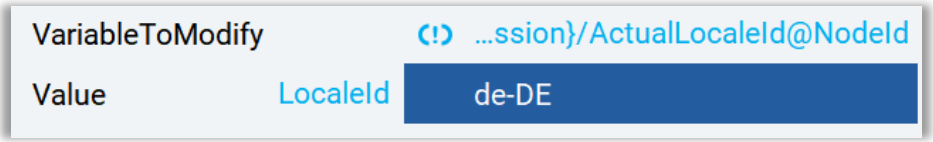
11. Under **LanguageSwitching**, right-click on the **en-US** button, and click **Copy**.



12. Right-click on **LanguageSwitching** and click **Paste**.

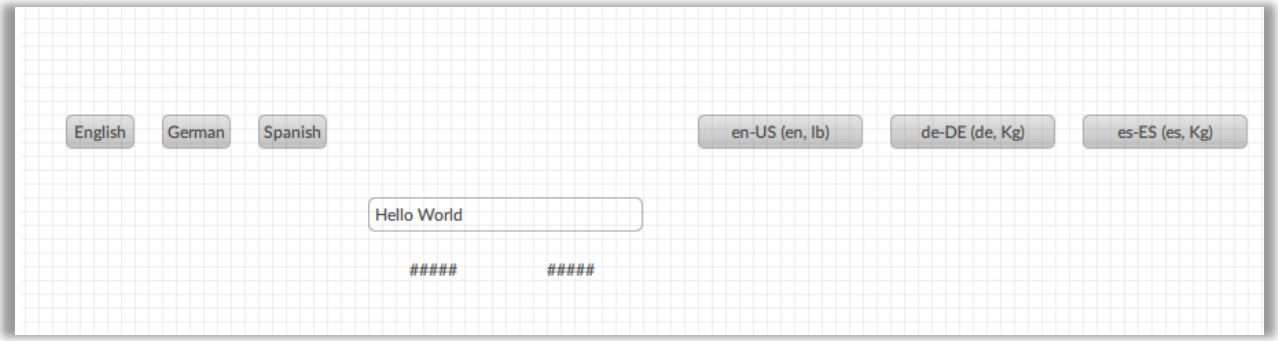


3. Rename this button "de-DE".
4. Change the **Left Margin** to "740" and the **Top Margin** to "120".
5. Change **Text** to: "de-DE" (de, Kg).
6. On the Events section, under **VariableToModify**, change the property **Value**: **de-DE**.



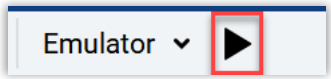
7. Under **LanguageSwitching**, right-click on the **en-US** button, and click **Copy.**
8. Rename this button "es-ES".
9. Change the **Left Margin** to "880" and **Top Margin** to "120".
10. Change **Text** to: es-ES (es, Kg).
11. On the Events section, under **VariableToModify**, change property **Value**: **es-ES**.

    The screen should look similar to the following:



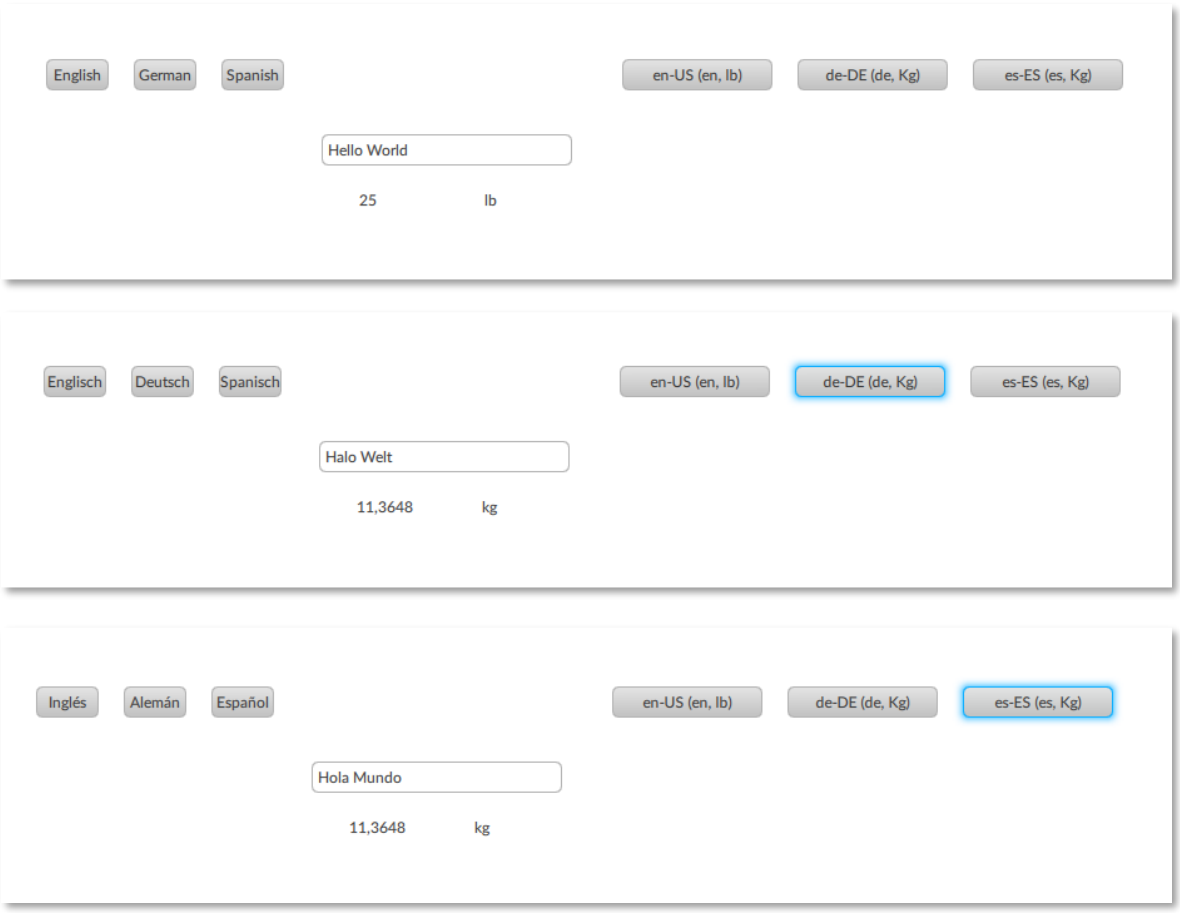
12. Add the **LanguageSwitching** screen to the **NavigationPanel1**.

13. Click the **Play** button next to the emulator.



14. Navigate to **LanguageSwitching**.

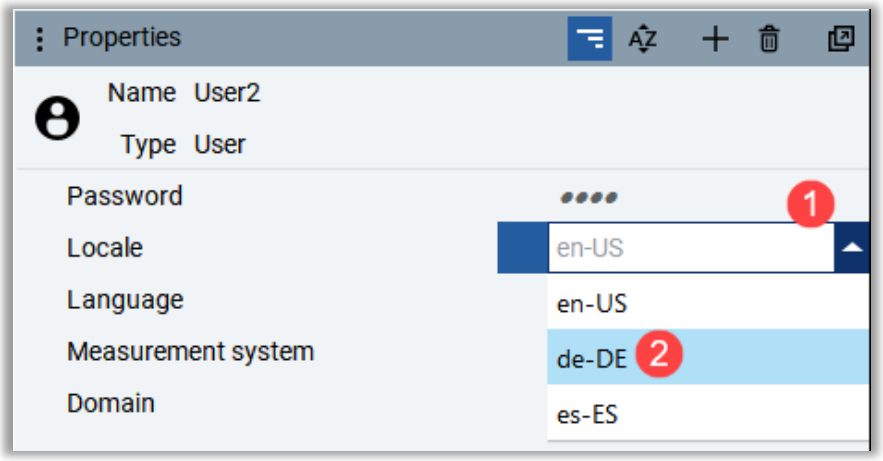
    Notice how the language of the text box, value of the analog variables and units change depending on the selected locale.



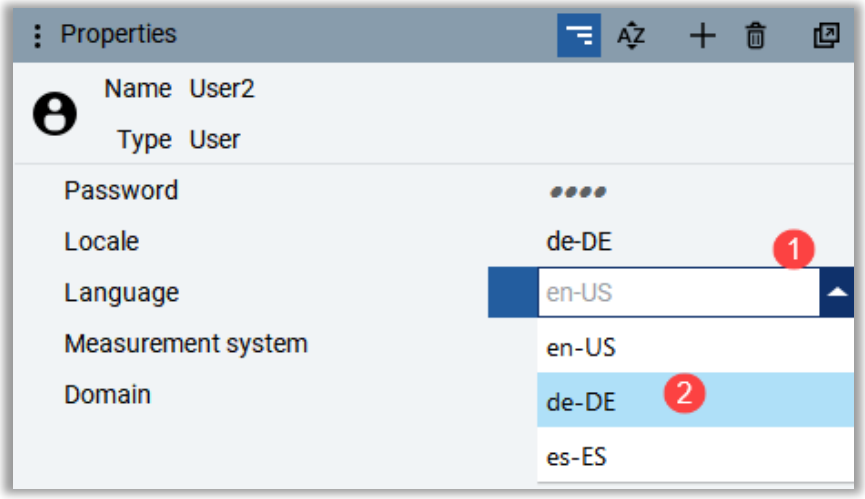
User Login with Locales (Optional)

Note: You must complete the Language Switching lab section before performing the following steps.

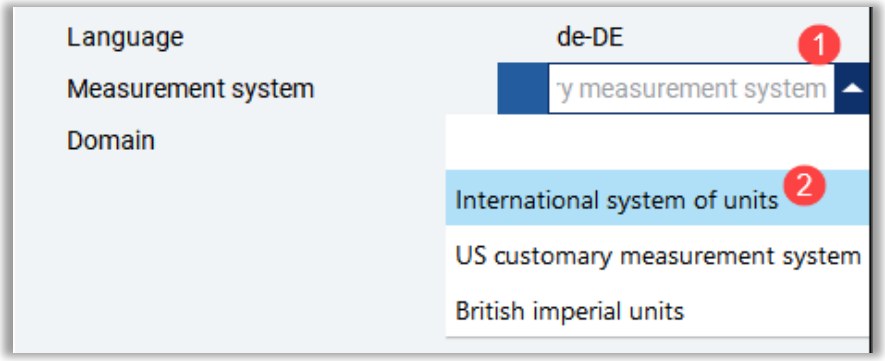
1. In the **Project view**, expand the **Security** node, expand the **Users** folder, and click **User2**.
2. Click the **Locale** drop-down arrow and click **de-DE**.



3. Click the **Language** drop-down arrow and click **de-DE**.



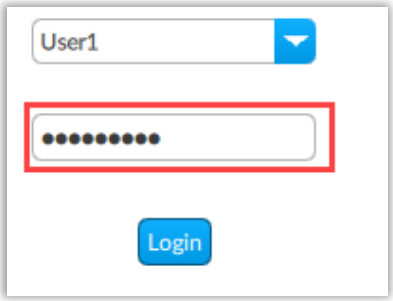
4. Click the **Measurement system** drop-down arrow and click the **International system of units**.



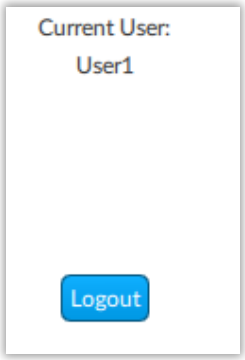
Emulate and Explore

1. From the toolbar, click the **Run on Emulator** icon ▶.

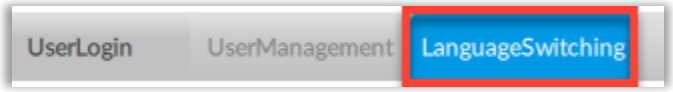
2. Click on the **UserLogin** tab in the Navigation panel.
3. Enter "password1" in the **Password** box and press **Enter**.



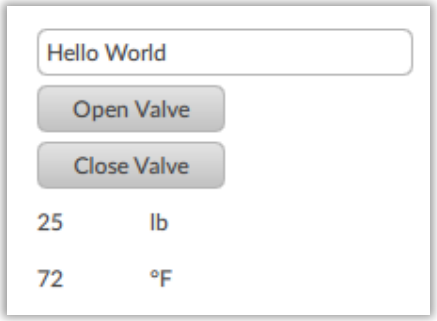
4. Click the **Login** button.



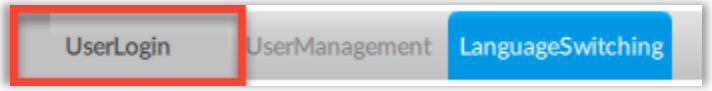
5. Click the **LanguageSwitching** tab in the Navigation Panel.



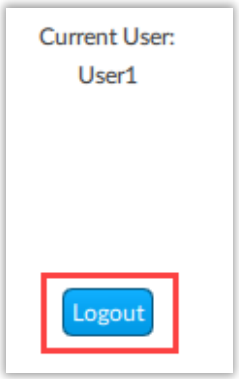
Observe that the text being displayed is in **English** as well as the US measurement system.



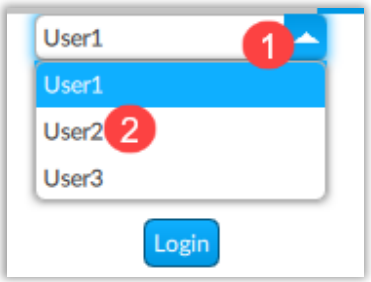
6. Click on the **UserLogin** tab in the Navigation panel.



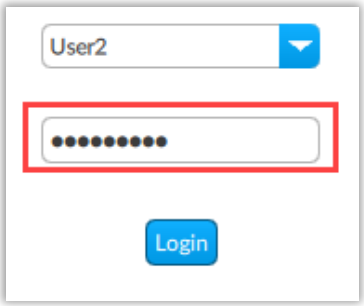
7. Click the **Logout** button.



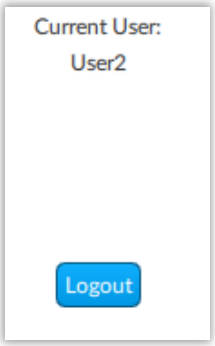
8. Click the **User** drop-down arrow button and click **User2**.



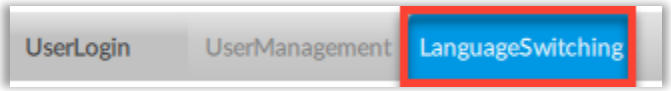
9. Enter “password2” in the **Password** box and press **Enter**.



11. Click the **Login** button.



12. Click the **LanguageSwitching** tab in the Navigation Panel.



13. Observe the following:
    - The text has changed to the German equivalent.
    - The value and engineering units have changed to the equivalent in the International system.

14. Close the **Emulator**.

Appendix A - Add Pre-configured Device Library Objects

Objectives

By the end of this exercise, you will be able to:

- Open a FactoryTalk Optix project
- Add pre-configured device library objects from a registered library template
- Add a graphic symbol to the main window
- Configure graphic symbol parameters
- Simulate and test the project
- Review the faceplate functionality

Scenario

You are adding a PowerFlex 755 variable speed drive to your system and would like the ability to easily commission the drive as well as diagnose and troubleshoot issues. To do so, you will familiarize yourself with the **FactoryTalk Optix Visualization Platform** by exploring and interacting with pre-configured PowerFlex 755 "**PF755" library objects** from the **Power Device Library**.

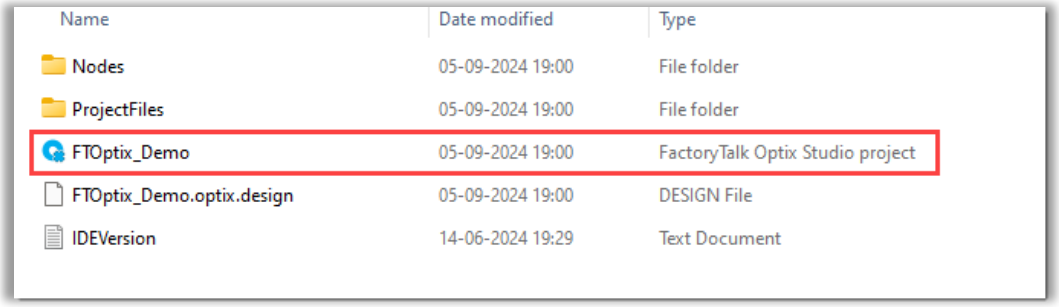
Libraries within FactoryTalk Optix serve as a collection of object definitions which may include **graphic elements (HMI faceplate displays, graphic symbols, etc.),** scripts, stylesheets, widgets, and other supporting objects. These library objects can be added to your projects to simplify development and ensure consistency. Users can also create customized libraries to suit their unique automation scenarios. A user-created library may contain a mix of objects curated from template libraries or customized elements, ensuring that automation solutions meet specific operational needs.

In this lab, you will be focusing on the **PF755 library objects**, which are designed for PowerFlex 755 drives. These objects are versatile and can be used to control and monitor various industrial equipment such as pumps, motors, conveyors, and agitators. The included faceplates and graphic symbols ease the ability to commission and troubleshoot the drive. By familiarizing yourself with these objects, you'll gain valuable insights into how to effectively use objects from **library template** within the FactoryTalk Optix environment.

Open FactoryTalk Optix project

1. Select **Lab Files** from the Windows Start Menu.

2. Locate C:\Lab Files\Introduction Lab\FTOptix_Demo.
3. Double-click **FTOptix_Demo**.



In the opened FactoryTalk Optix project, we have already pre-configured some components to simplify some steps. Since the following tasks have already been completed, these details are provided just for your information:

4. The **Power Device Library** is pre-registered.
5. The default stylesheet is set to **ISA Stylesheet**.
6. A ControlLogix **EtherNet/IP station** is already configured and the required tags have been imported into the project.

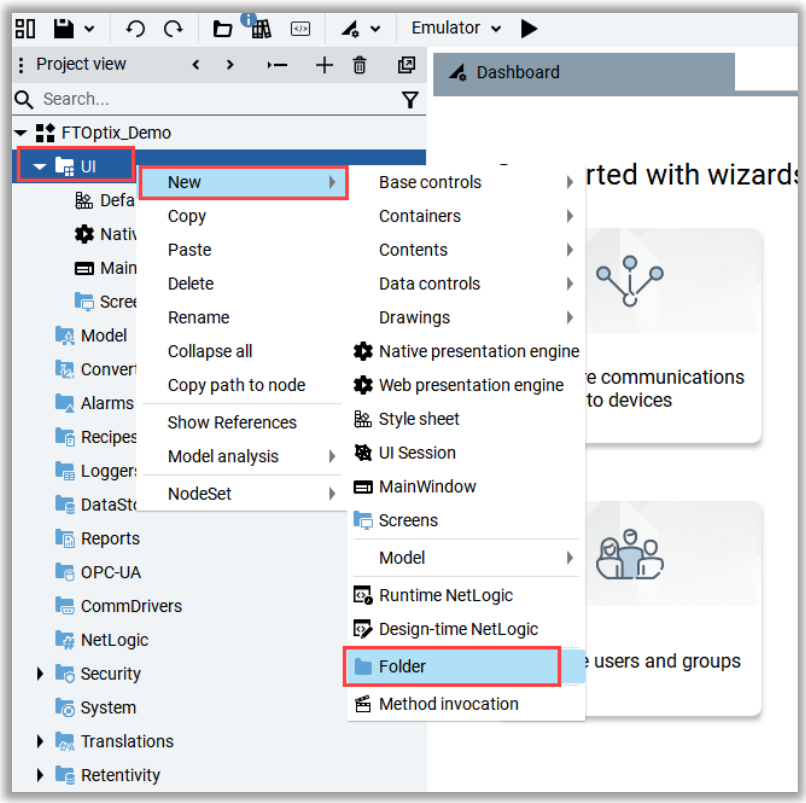
Import library objects to the project

In this section, you will add objects from the **Power Device Library** to your FactoryTalk Optix project.

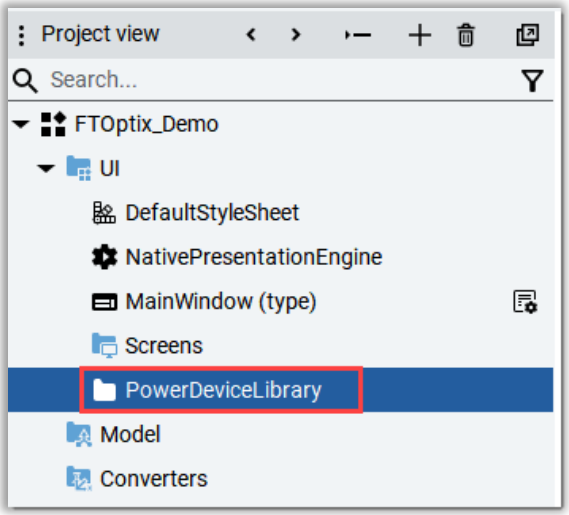
Note: These objects are already registered in the library template and can be used as needed. These objects were previously registered using the **Setup.cmd** script: Run the provided **Setup.cmd** script or extract the downloaded zip file to the Windows user's FactoryTalk Optix library directory. By default, this directory is located at: **C:\Users\<UserName>\Documents\Rockwell Automation\FactoryTalk Optix\Libraries**

1. It is a good idea to organize libraries and folders. Once the project is opened, create a new sub-folder under UI and rename it to "**PowerDeviceLibrary**".

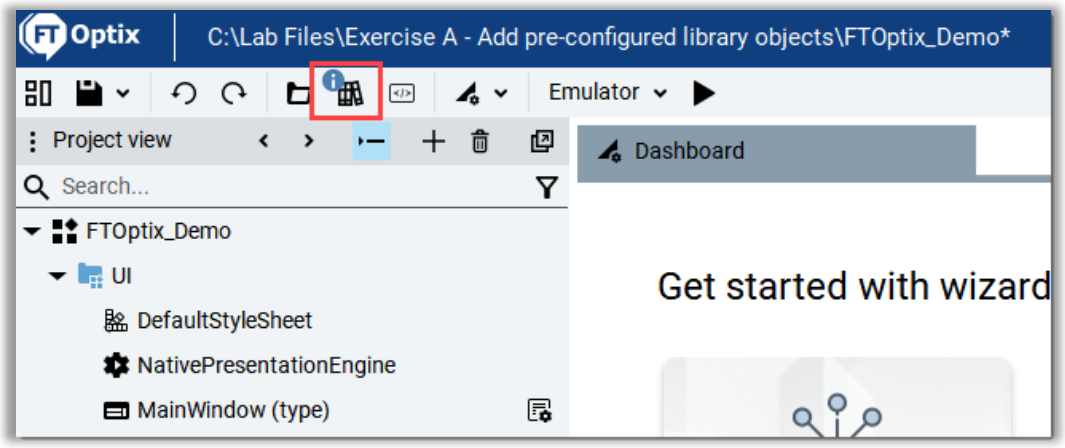
2. Under the **Project view**, right-click on **UI > New > Folder**.



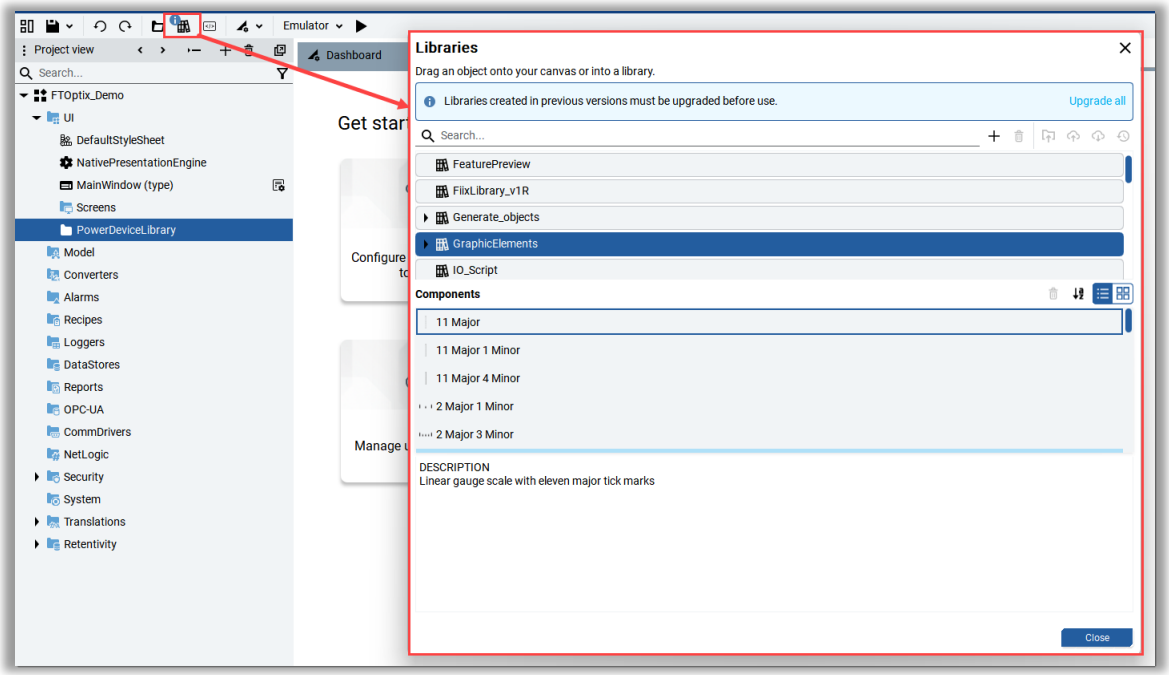
3. Hover over **Folder1**, select the edit icon , and enter "PowerDeviceLibrary" as the new name.



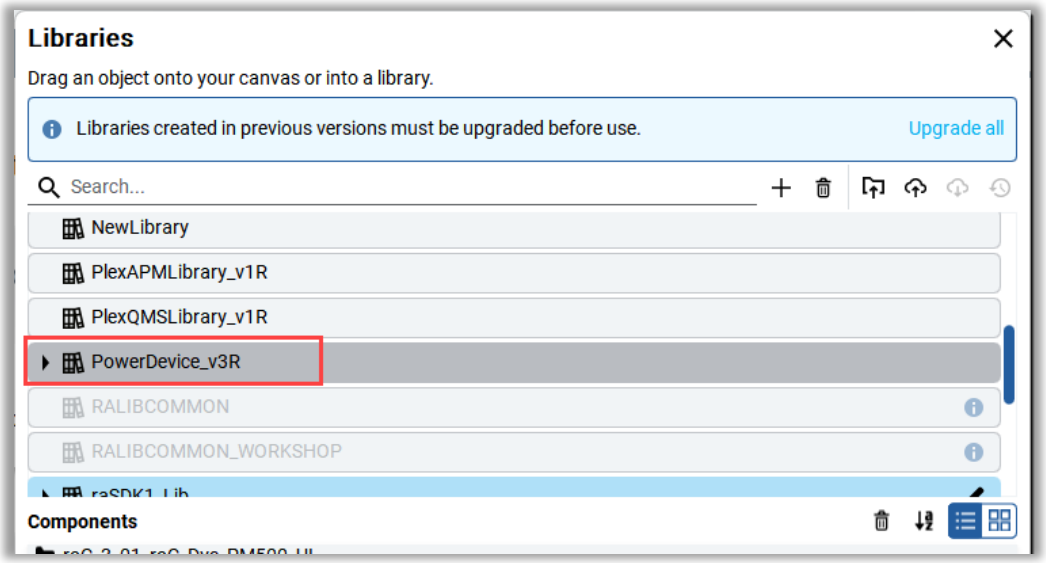
4. Now, click on the **Libraries** icon  in the menu bar to open the Libraries window.



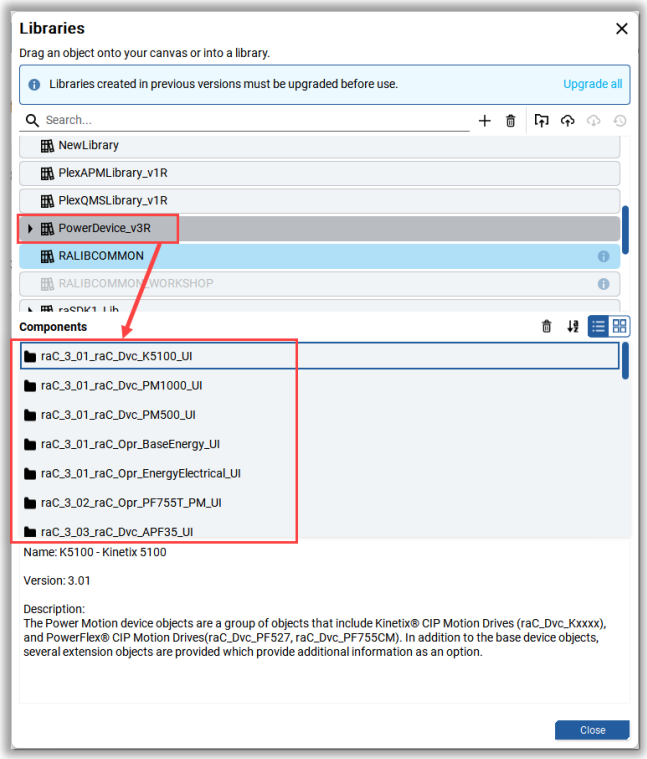
The libraries window will display all the libraries available from the default libraries location which have already been registered.

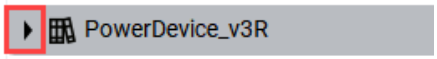


5. In the library list, scroll and look for library **"PowerDevice_v3R".** This folder contains all of the library objects for the Power Device Library v3.



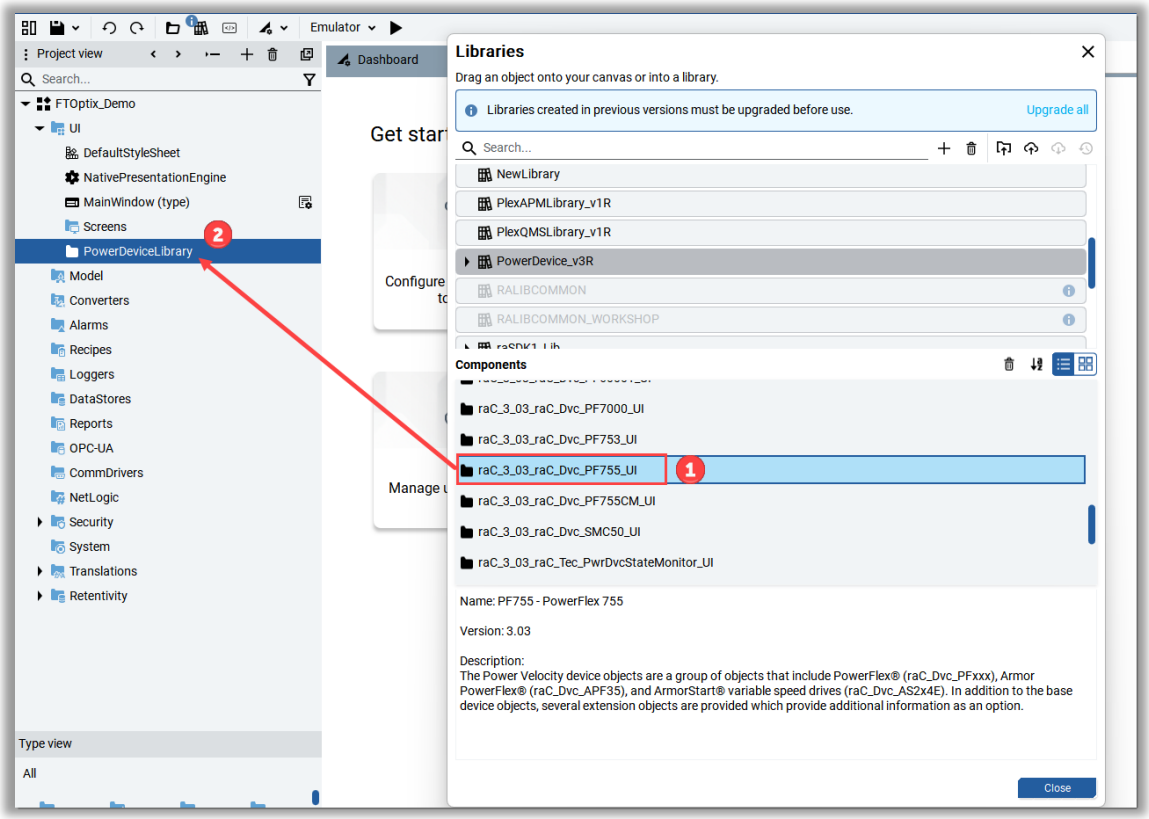
6. Click on the "**PowerDevice_v3R"** library to see its components. When you click it, the available objects will be displayed in the "**Components**" pane.



Note: When you click the "**Expand**" button, 

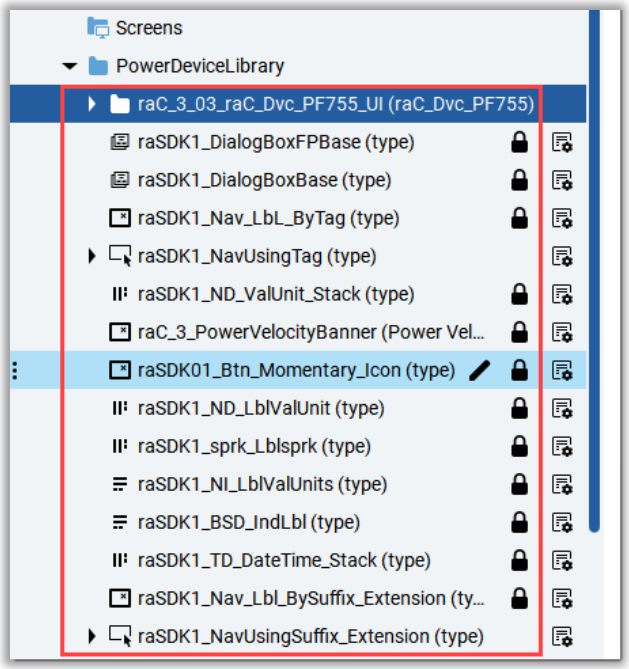
it will display the library's subcategory folders, making it easier to filter objects.

7. Select the "**raC_3_03_raC_Dvc_PF755_UI**" object and then drag and drop it into the new library folder in the project.



Note: If it appears the object is not dropping into the folder, try moving the cursor back and forth over the target folder while keeping the mouse clicked down.

8. This folder now contains types for the device faceplate, graphic symbols, private elements like panels and widgets, and the help panel of the object. It has everything you need to use this library object in your project!



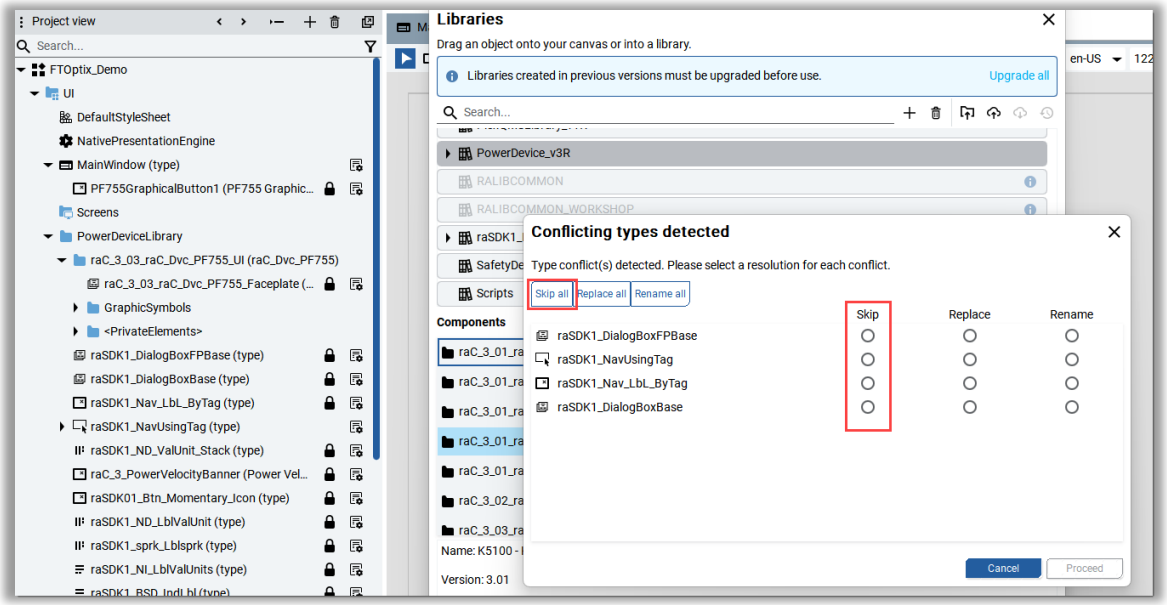
Note: Some of the common parent components like SDK components (which are commonly used in the Device Object libraries) are placed outside the object folder.

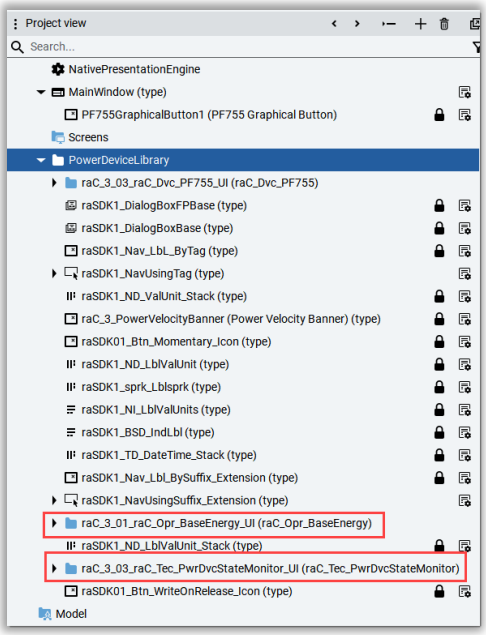
9. Similarly, you can add the below objects to **PowerDeviceLibrary** folder by dragging and dropping them.

   - raC_3_01_raC_Opr_BaseEnergy_UI
   - raC_3_03_raC_Tec_PwrDvcStateMonitor_UI

Note: At this time, you may see a "**Conflicting types detected**" window pop-up with three options to address the conflicts: "Skip all", "Replace all" or "Rename all". You can select "**Skip all**" to avoid overwriting existing types.

10. Once the objects are added to the project, you can **close** the libraries dialog.





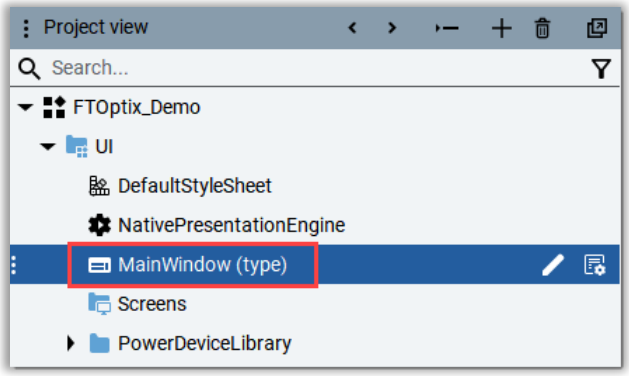
Add and configure graphic symbol parameters

It's your turn to try it:

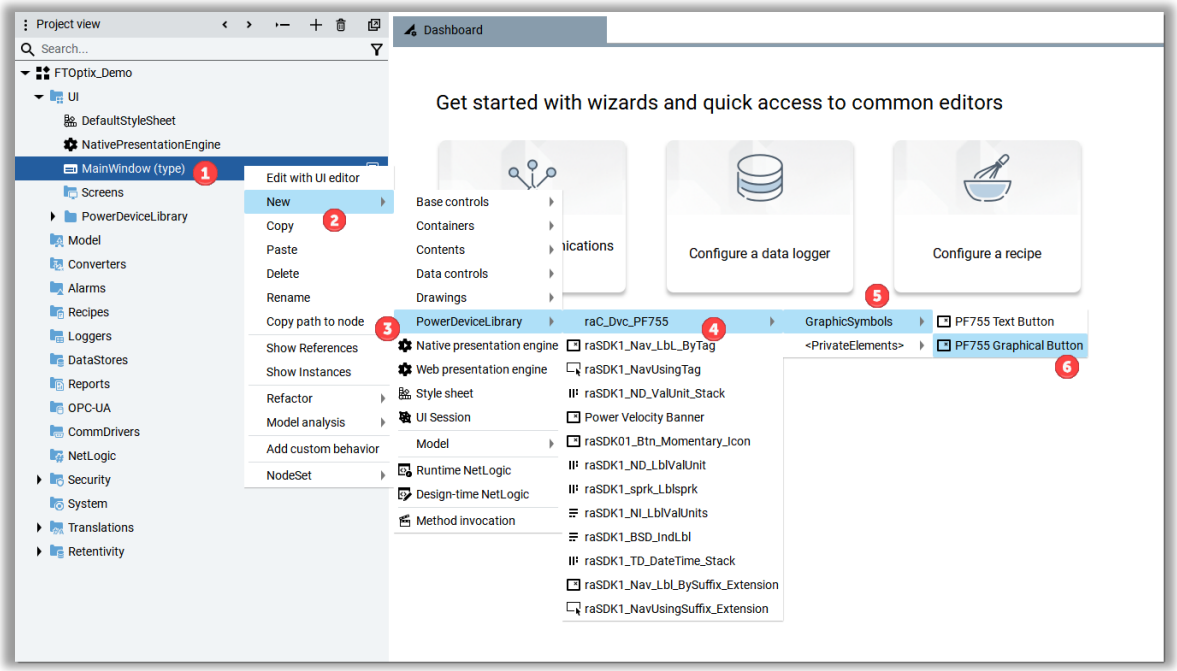
In this section, you will add pre-configured Graphic symbols to your FactoryTalk Optix project main window. Graphic symbols are graphical representations of the devices, which include buttons to open the related faceplates and include some basic diagnostics.

1. Once the Objects have been imported into the FactoryTalk® Optix Studio project, you can begin using them in your application.

2. To add a new Launch Button to a Main window, right-click on **MainWindow (type)** under **Project view > UI**.

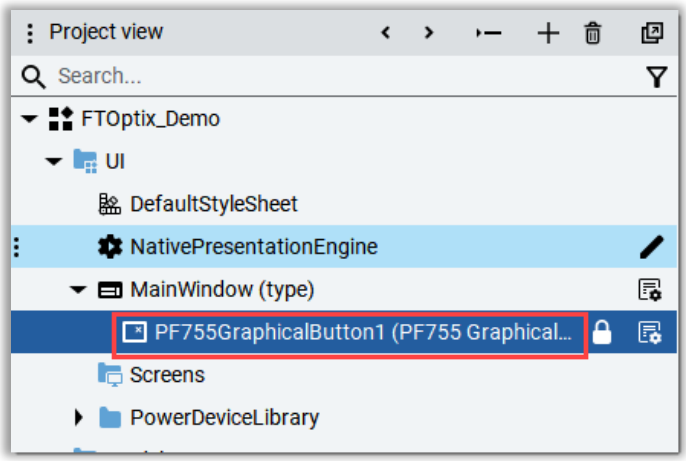


3. Navigate to **MainWindow (type) > New > PowerDeviceLibrary > raC_Dvc_PF755 > GraphicSymbols >** and select **PF755 Graphical Button**.



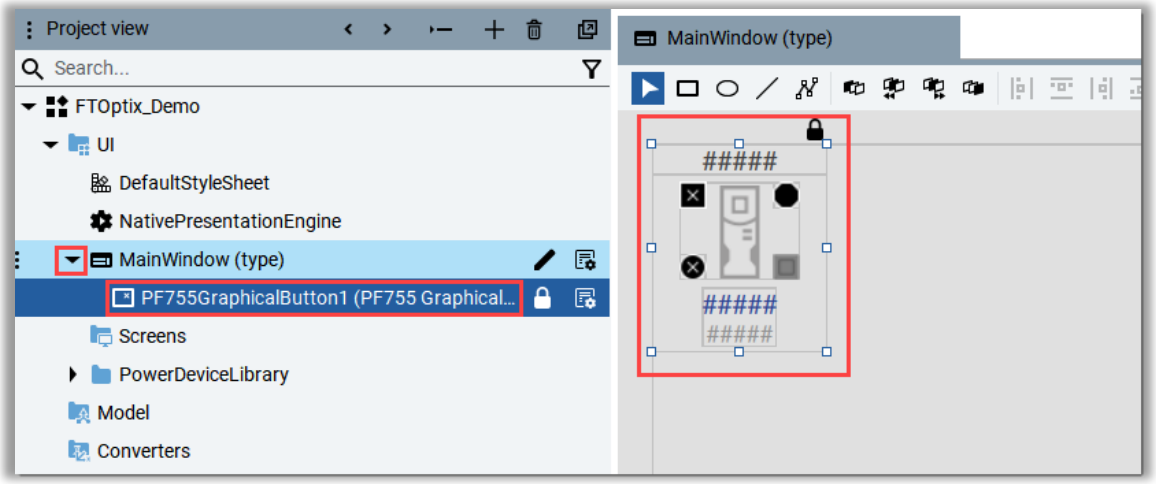
Note: You can see there are multiple types of **Graphic Symbols** available. Depending on your needs, you can choose between a simple text button and graphical buttons.

4. This will add an instance of the **PF755 Graphical Button** named "PF755GraphicalButton1" to the main window.

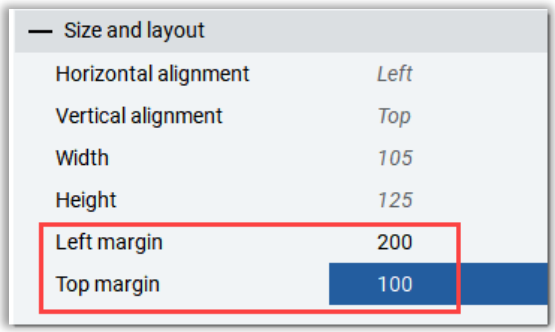


5. Double-click on **MainWindow (type)** to open the screen in UI Editor.

6. Expand **MainWindow (type)** and select **PF755GraphicalButton1**.

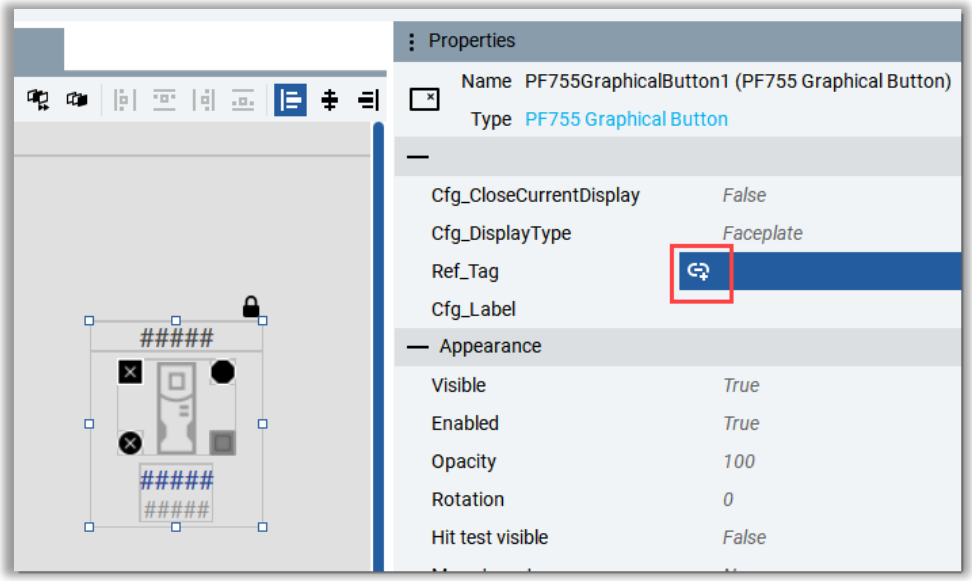


7. In the **Properties panel > Size and layout >** set the **Left margin** to "200" and the **Top margin** to "100".

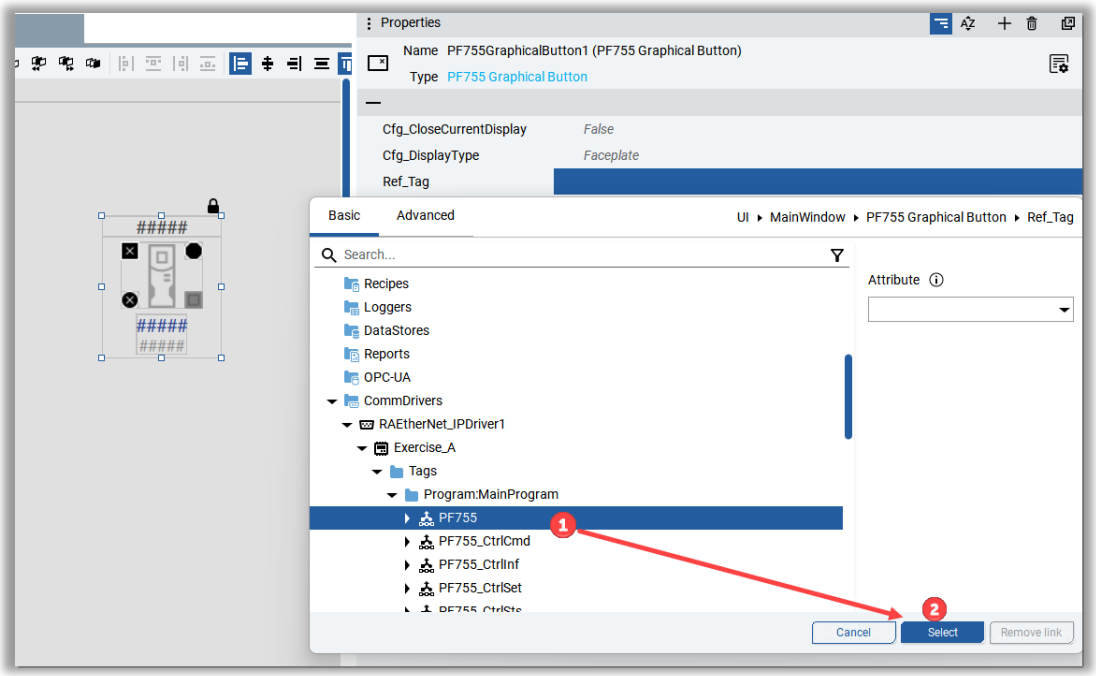


Now we can link the instance of the graphic symbol button to the instance of the device object Add-On Instruction (AOI) in the Logix controller.

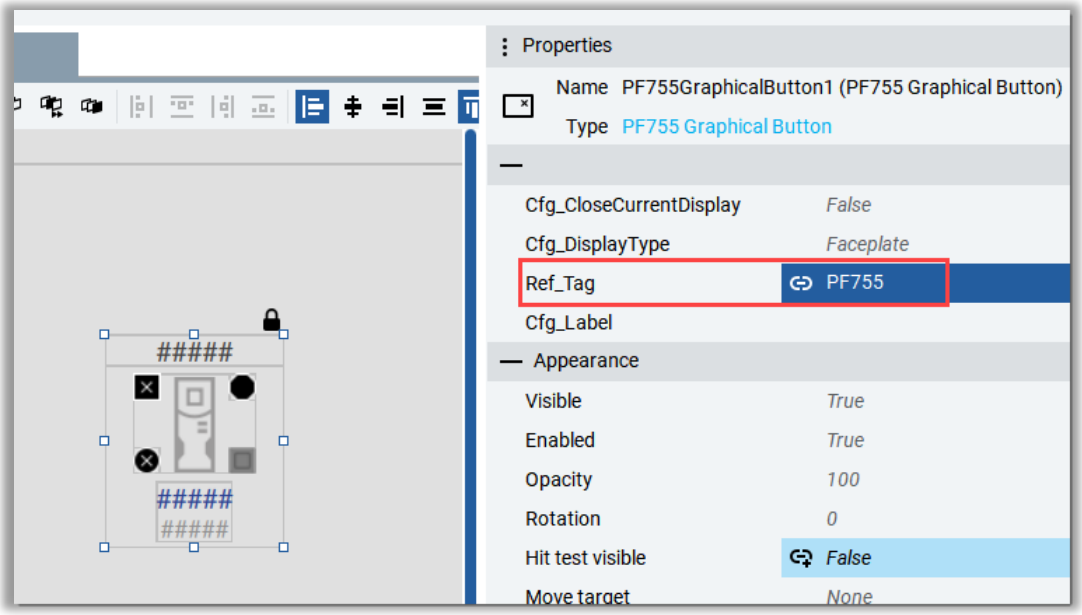
8. Select the **PF755GraphicalButton1 > Properties** panel. Next to the **Ref_Tag,** click the **browse** link button to search for the Add-On Instruction tag in the ControlLogix EtherNet/IP station.



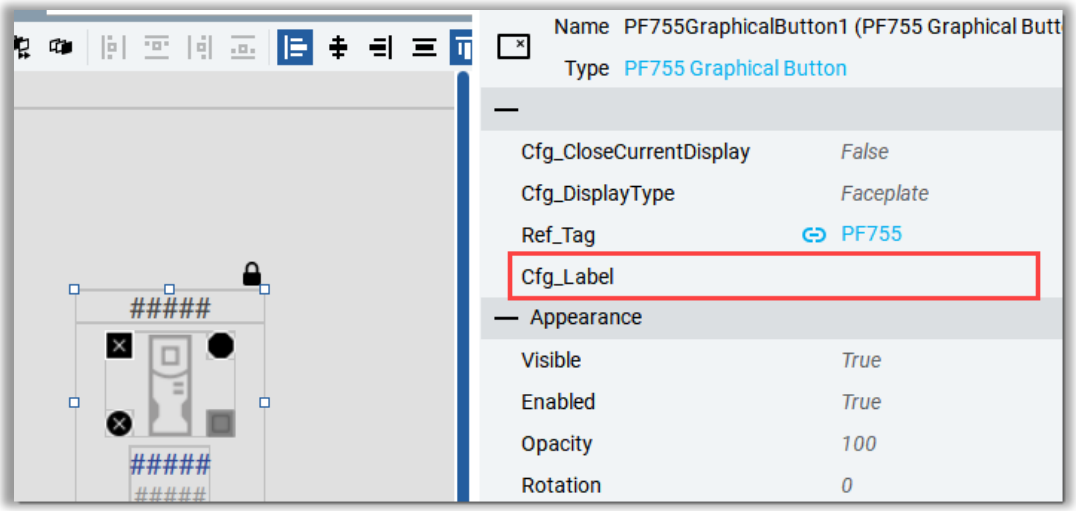
9. This will open node browser window. Using the search bar, search for "**PF755**" or browse for **FTOptix_Demo > CommDrivers > RAEtherNet_IPDriver1 > Exercise_A > Tags > Program:MainProgram**, click "**PF755**", and click **Select.**



10. This will assign **Ref_Tag** to the **PF755** Add-On Instruction instance tag.



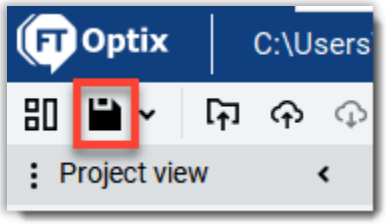
11. For the **Cfg_Label** Configuration Label parameter, you can leave it blank to automatically display the tag's description metadata stored in the ControlLogix program. Alternatively, you can type in a custom string to be displayed on the graphic symbol.



12. All other parameters can remain at their default values.

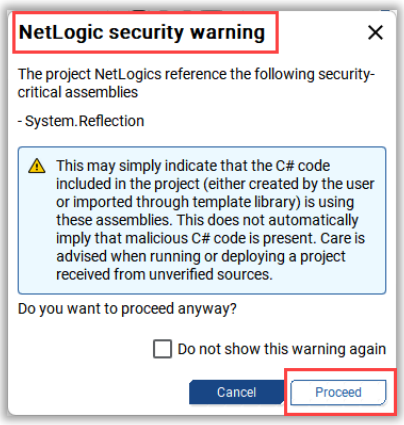
Simulate and test the project

1. Save the project by using the **Save** icon.



2. Click the **Play** button beside **Emulator** to deploy the project.

3. If you receive **NetLogic security warning**, click **Proceed** to deploy the project.



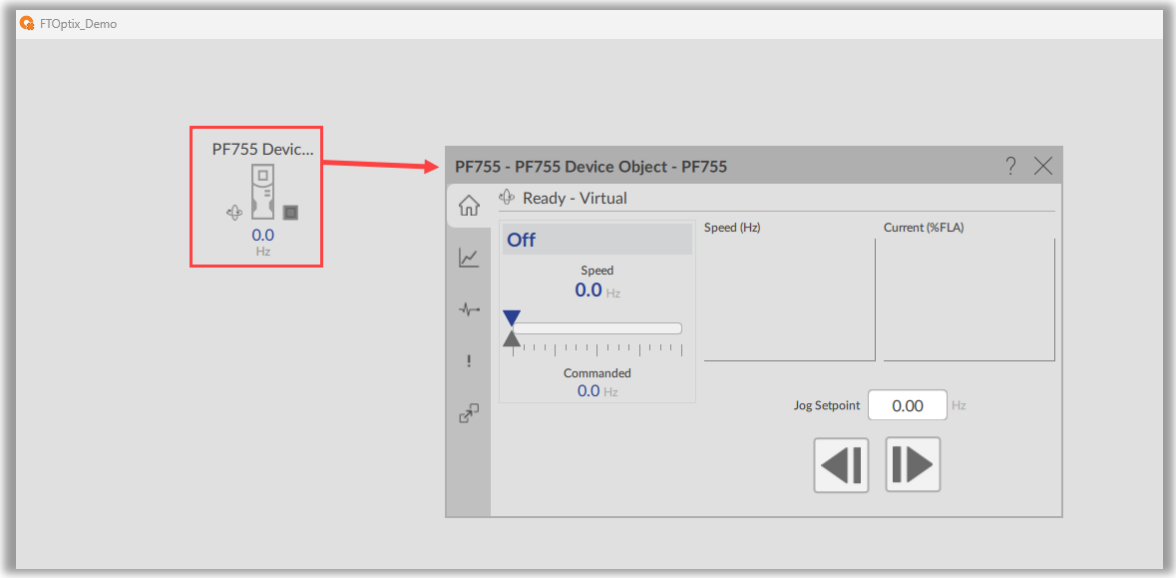
4. This will launch Emulator with the MainWindow with the caption “FTOptix_Demo”.

5. The main window now displays the **PF755 graphical symbol button**.

Explore the faceplate functionality

It's your turn to try it:

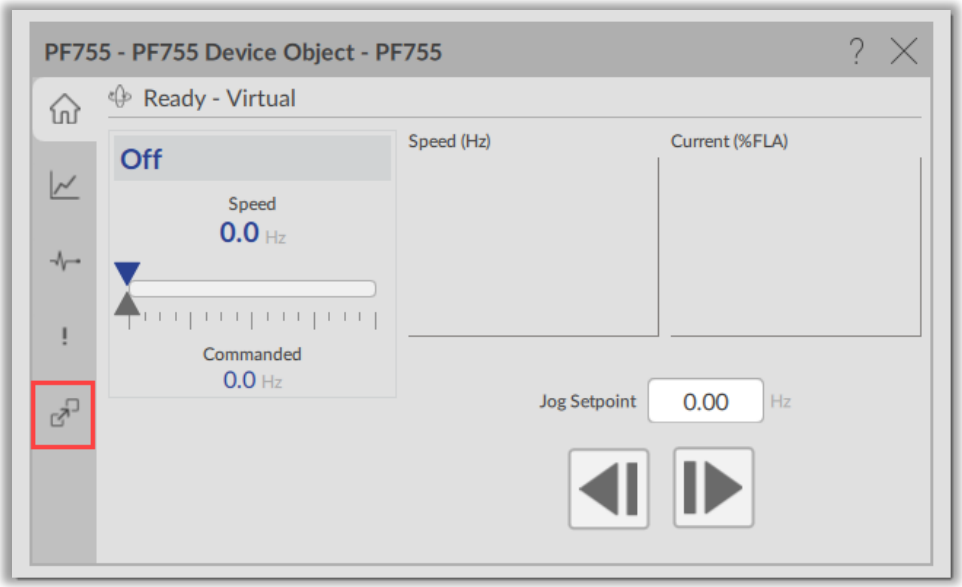
1. In the runtime emulator window, click the **PF755 graphical symbol button**. This will open the Faceplate of the PF755 object with the home tab as the default window. From this initial tab, you can easily observe the current status of the PowerFlex drive. You can also assign a jog setpoint and perform jog forward or reverse commands to commission or verify the functionality of the connected equipment.

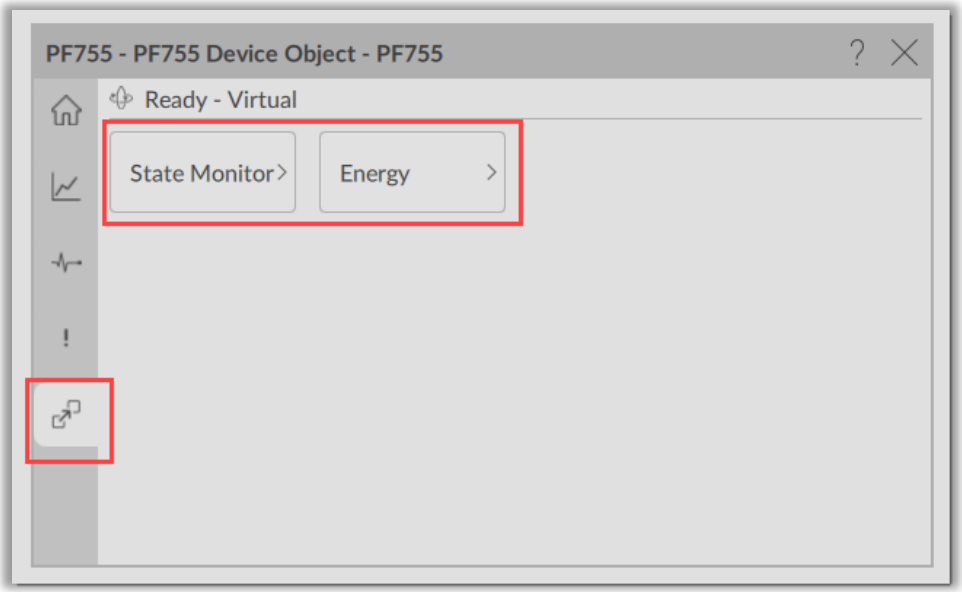


2. At this stage, review the contents found under each tab of the **PF755 device object** faceplate . This will allow you to gain a comprehensive understanding of the functionality and features provided under each tab.

   Note: Also note that along with the **PF755 device object** faceplate, there are additional extension objects (for example, the **Energy Extension** object and **State Monitor** object).

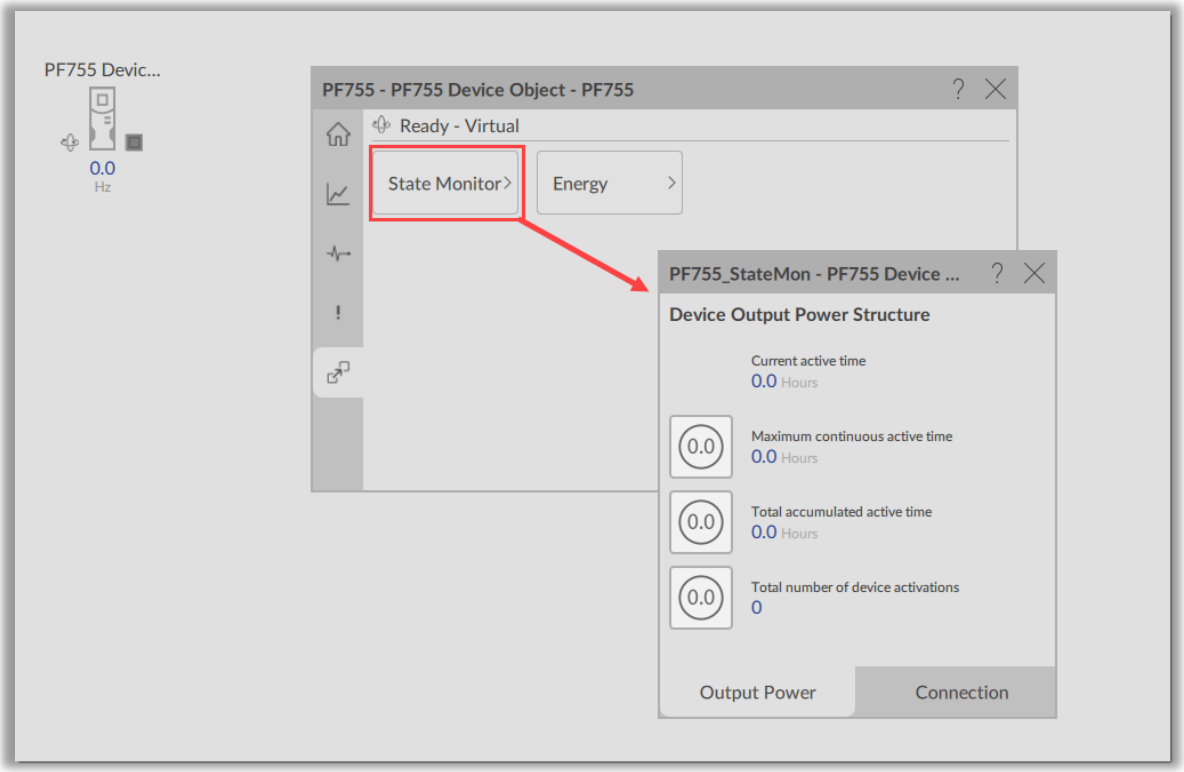
3. To access these extensions, click on the **Extension** tab on the device object faceplate. The available objects will be shown as enabled navigation buttons.





4. Click the **State Monitor** button to open the State Monitor extension object faceplate.

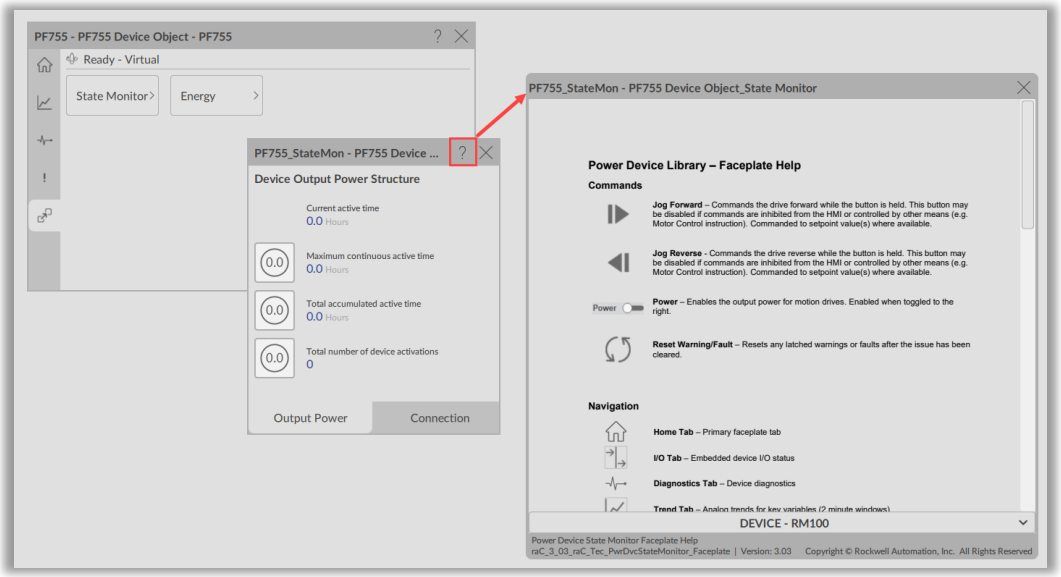
   State Monitor Extensions enable the user to track, view, and reset counters for runtime hours, number of starts, and more. This information is available to be viewed in the HMI or to be programmatically accessed by user applications.



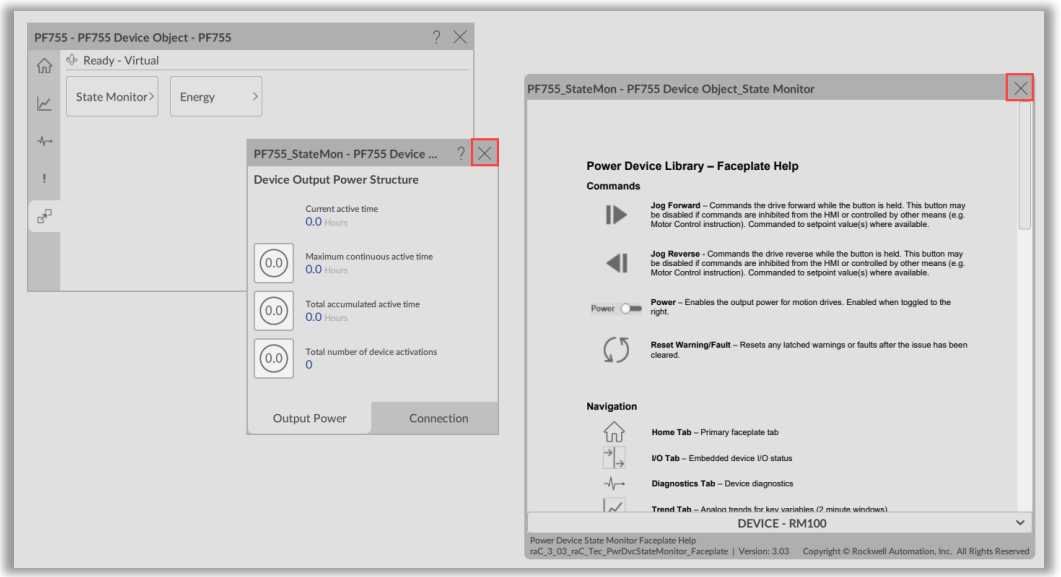
   Examine the contents under each tab of the extension to gain a thorough understanding of the capabilities and features provided by the state monitor extension object.

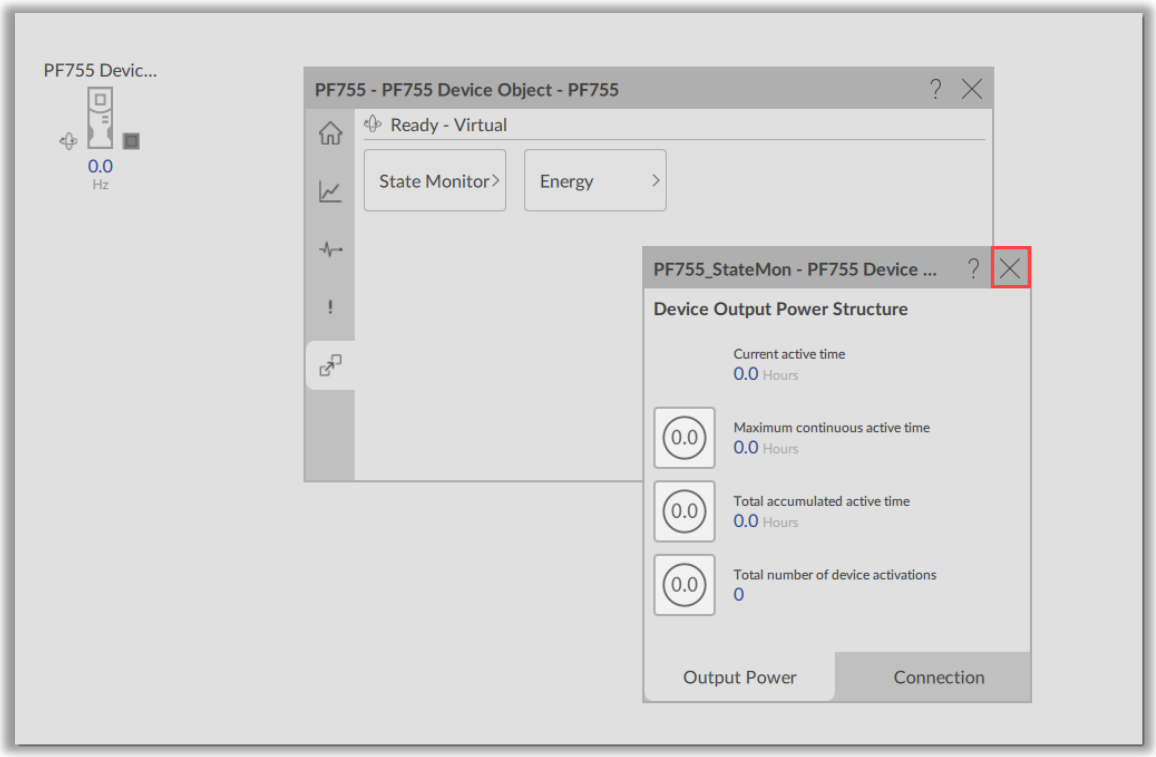
   Additionally, the faceplates are provided with a **Help** button for each faceplate. The Help button is located in the upper-right corner of the faceplate frame.

5. Clicking on the **Help** button will open a pop-up display that includes a Help Document that provides quick help about the contents of the faceplate. There is also an accordion menu that contains an embedded web browser that gives you access to the full Power Device Library reference manual from web.



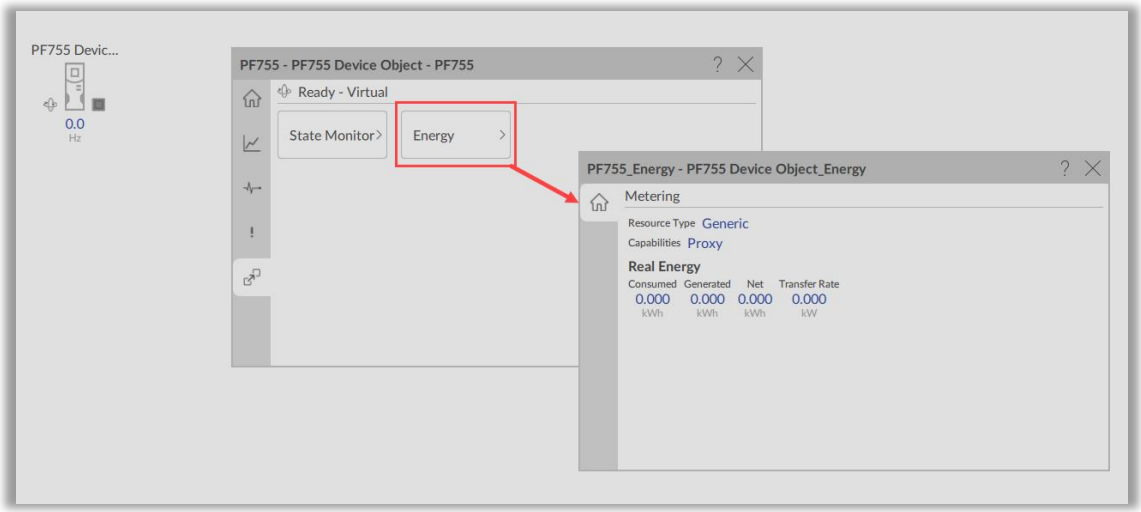
6. Click the **Close** button to close the state monitor extension object and help.



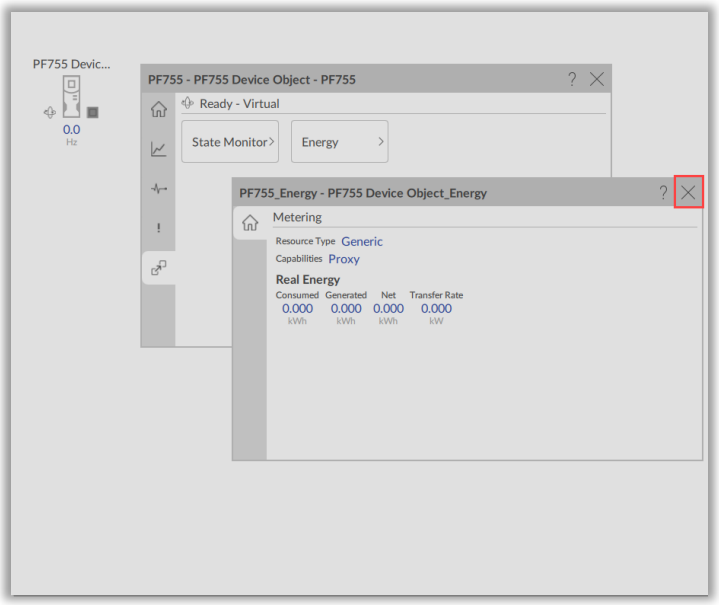


7. Click the **Energy** button to open the Energy extension object faceplate.

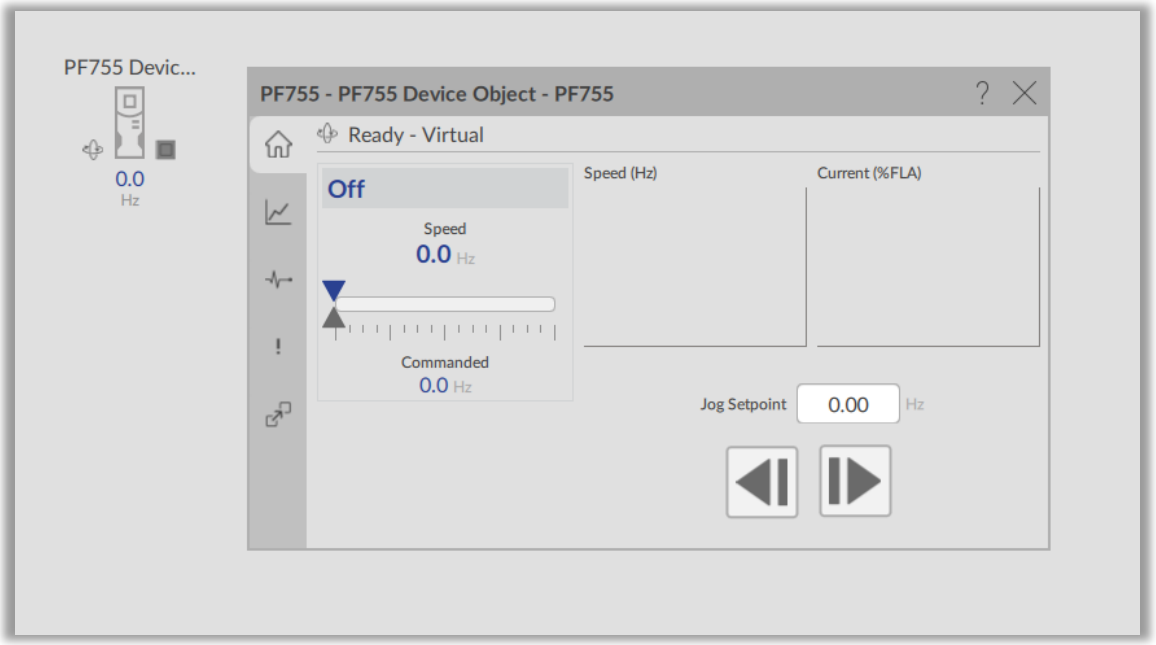
   Energy Extensions enable the user to monitor electrical energy information related to the device including voltage, current, power, energy and frequency.



8. Click the **Close** button to close the energy extension object.



9. Click the **Close** button to close the PF755 object faceplate.



10. If time allows, feel free to continue to review the different tabs and functions provided by the PowerFlex 755 faceplate.

11. When you are finished, close all objects and the Emulator window.

Connect with us.

rockwellautomation.com — expanding **human possibility**™

AMERICAS: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444
EUROPE/MIDDLE EAST/AFRICA: Rockwell Automation NV, Pegasus Park, De Kleetlaan 12a, 1831 Diegem, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640
ASIA PACIFIC: Rockwell Automation, Level 14, Core F, Cyberport 3, 100 Cyberport Road, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846

Publication TSP-00000 - September 2024 | Supersedes
Publication TSP-00000 - September 20204 Copyright © 2024