



FT FactoryTalk



**Rockwell  
Automation**

expanding **human possibility**®



# FactoryTalk Optix Technical Training

ASEM • APPLICATION SUPPORT ENGINEERS • 08/2025

# Table of contents



# Table of contents – Pre requirements

- (Pre-requirement) Getting started
- (Pre-requirement) Be familiar with FactoryTalk Optix Studio
- (Pre-requirement) How to Create a simple HMI project in few steps using wizards
- (Pre-requirement) WebIDE
- (Pre-requirement) Project versions and components
- (Pre-requirements) FactoryTalk Optix Entitlements

# Table of contents – Basic topics

- (Base) Define communication drivers and tags
- (Base) Develop the user interface
- (Base) Dynamic links
- (Base) Work with reusable objects
- (Base) Template Library
- (Base) UI Objects Protection
- (Base) Animations
- (Base) Events
- (Base) DataStores
- (Base) DataLoggers and EventLoggers
- (Base) Represent data by Trends
- (Base) OPC-UA Client
- (Base) OPC-UA Server
- (Base) Transfer and Execute the project

# Table of contents – Basic topics

- [\*\*\(Base\) Manage alarms\*\*](#)
- [\*\*\(Base\) Use recipes\*\*](#)
- [\*\*\(Base\) Configure Users and Groups\*\*](#)
- [\*\*\(Base\) Configure Reports\*\*](#)
- [\*\*\(Base\) Set Locale, Languages and Measurement System\*\*](#)
- [\*\*\(Base\) Develop using version control and collaboration\*\*](#)

# Table of contents – Advanced topics

- [\(Advanced\) Define client-server architectures via OPC-UA](#)
- [\(Advanced\) Domain authentication](#)
- [\(Advanced\) Define UI Session and Session variables](#)
- [\(Advanced\) Work with NetLogic scripts](#)
- [\(Advanced\) Debugging NetLogic scripts](#)
- [\(Advanced\) Use Retentivity](#)
- [\(Advanced\) MQTT](#)
- [\(Advanced\) REST API](#)
- [\(Advanced\) Optix Runtime on Docker containers](#)
- [\(Advanced\) OPC-UA NodeSet](#)
- [\(Advanced\) OPC-UA custom behaviors](#)
- [\(Advanced\) Custom keyboards](#)

# Table of contents - Appendix

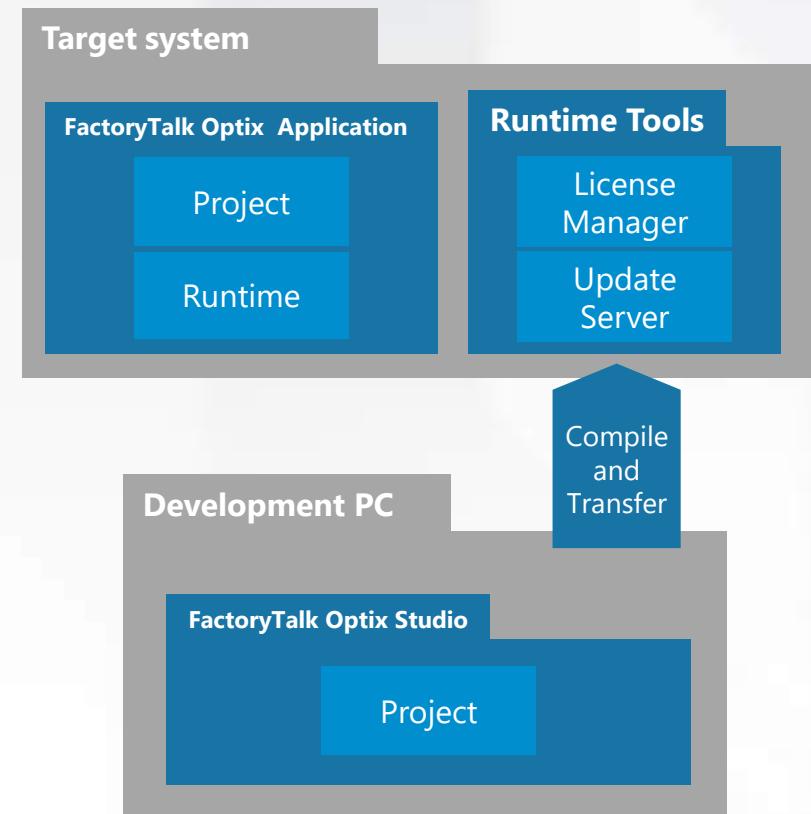
- [Appendix: See the light about OPC-UA Security](#)
- [Appendix: Generating OPC-UA certificates](#)
- [Appendix: Using OPC-UA certificates](#)
- [Appendix: Returning values from NetLogic methods](#)
- [Appendix: Connect to an External SQL Database via ODBC connector](#)
- [Appendix: Connect to an External InfluxDB instance](#)
- [Appendix: Additional resources](#)

# Getting Started



# Architecture

- **FactoryTalk Optix Studio** is the Integrated Development Environment (IDE) used to design and deploy HMI and IIOT applications
- **FactoryTalk Optix Application** is the compiled and "portable" application that combines the Project code and the Runtime executable for the Target (HMI or iPC)
- **Runtime Tools** is the set of tools that includes:
  - **Update Server**: Software to allow the deployment of applications from Studio to the target
  - **License Manager**: Software to manage the Runtime entitlements on the target device



# FactoryTalk Optix key features



## EXTENSIBILITY options

Natively OPC UA

Libraries

Scripting



## DESIGN options

Object-oriented

Software As  
A Service

Version Control



## GRAPHIC & DEPLOYMENT options

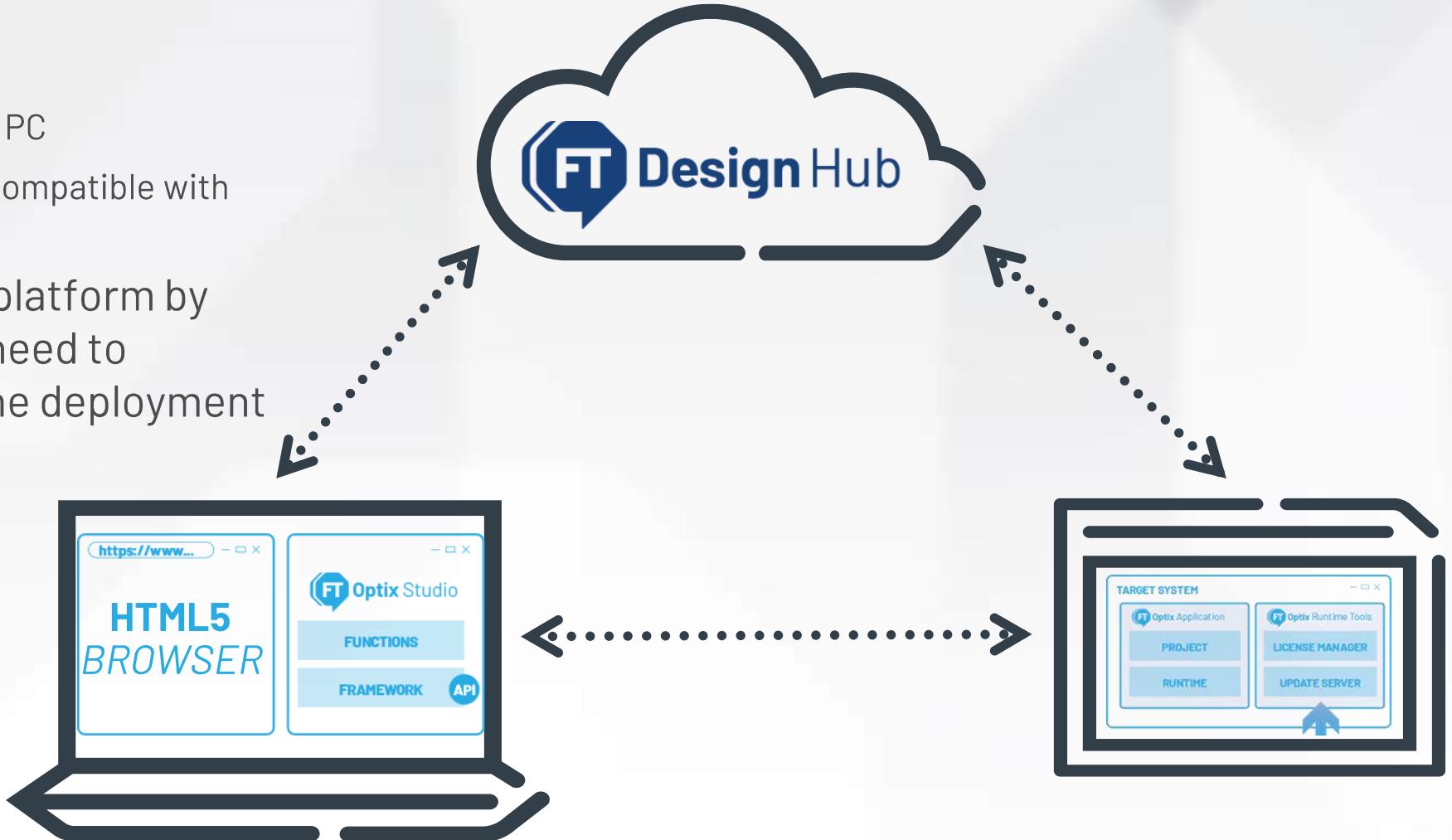
Cross-platform

Universal UI

Audit

# FactoryTalk Optix Studio

- Available both
  - On premise: as a setup on local PC
  - As a service: a cloud instance compatible with any HTML5 browser
- All applications are cross-platform by design, the user does not need to choose or know which is the deployment target



# FactoryTalk Optix Studio flavours

## STANDARD

Installed on local PC

Design and Deploy application  
from your PC

No screens, tags, alarms or other  
Runtime features limitations

---

FREE



## PRO

*STANDARD capabilities plus...*

Use the **Version Control** System  
(both on project and libraries)  
to enable multi-user **collaboration**

---

Design application using  
**web-based** Studio

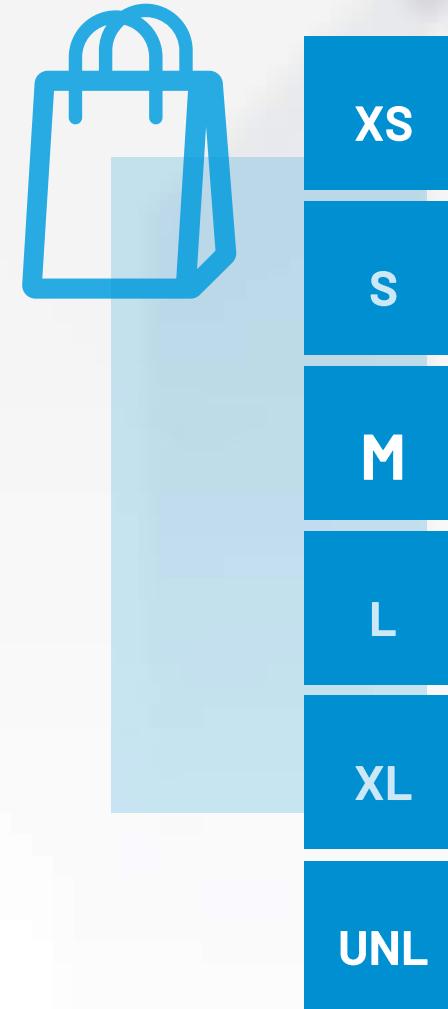
---

Annual subscription  
per User

# FactoryTalk Optix Runtime – Feature tokens overview

- Every **feature** used by the Runtime has a cost in **Tokens**
- License **size** defines the number of Tokens available

SIZE	RUNTIME FOR IPC
XS	5
S	8
M	11
L	15
XL	21
UNL	Unlimited



# FactoryTalk Optix Runtime – Feature tokens example

## SMALL HMI

Native UI	1
Web UI (up to 1 clients)	1
Alarms + Alarms History	1
Data logger	1
Recipes	1
1x Comm Driver (multiple stations)	2
<b>TOTAL</b>	<b>7</b>

## MEDIUM HMI

Native UI	1
Web UI (up to 3 clients)	2
Alarms + Alarms History	1
Data logger	1
Retentivity	1
Recipes	1
OPC UA Server (1 client)	1
1x Comm Driver (multiple station)	2
<b>TOTAL</b>	<b>10</b>



# Ready to start with FactoryTalk Optix?

- Just need to create a «My Rockwell» account then...
- Optix Studio STANDARD can be downloaded for free from  
<https://compatibility.rockwellautomation.com/>
- Optix Studio PRO is available as 90 Days Trial at  
<https://home.cloud.rockwellautomation.com>

A large, abstract graphic of blue dots forms a wave-like pattern across the background, starting from the bottom left and curving upwards towards the top right.

# Be familiar with FactoryTalk Optix Studio



# Development environment: new project wizard

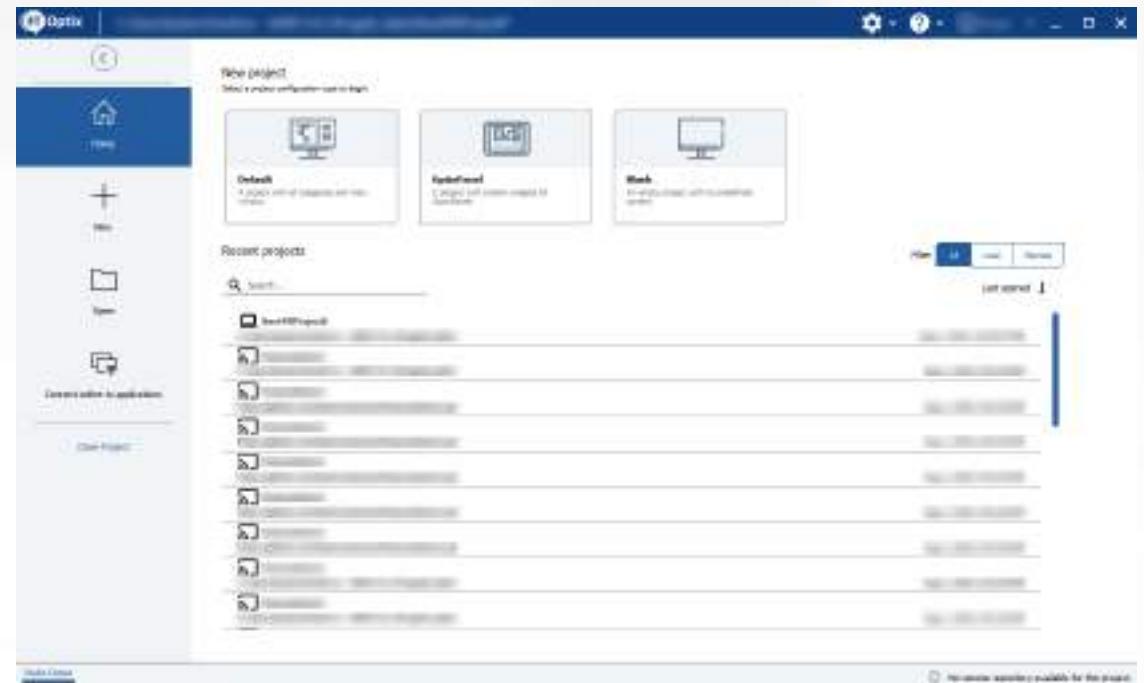
Choose between

- Default: project with default folders/resource structure (recommended)
- OptixPanel: project with Optix Panel's dedicated widgets (network, screen settings, etc)
- Blank: empty project, without folders/resources structure (only if «advanced mode» is enabled)

**Important:** this selection does not limit in the deployment capabilities or available features

## Project configuration

- Name, Location
- Use version control: allow the project versioning and collaboration between developers (GIT)
- Encrypt secrets: passwords are encrypted via a private key stored in the FactoryTalk HUB domain
- Default screens resolution (can be changed later)



# Development environment: new project wizard

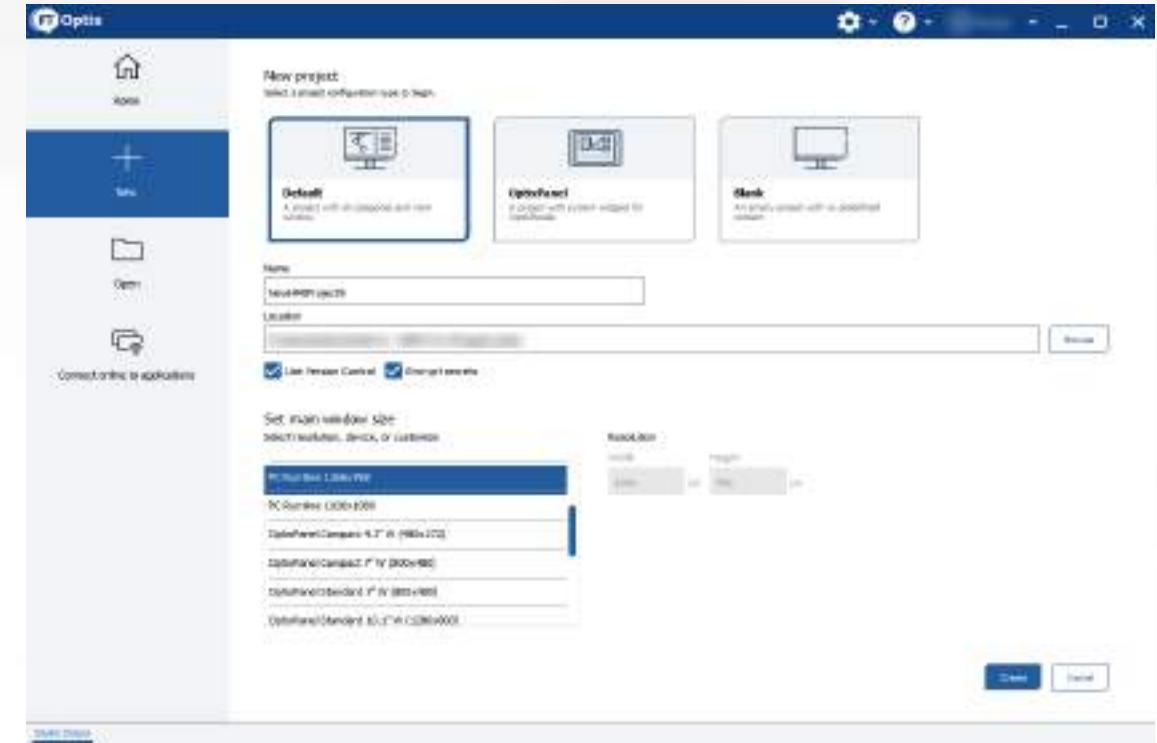
Choose between

- Default: project with default folders/resource structure (recommended)
- OptixPanel: project with Optix Panel's dedicated widgets (network, screen settings, etc)
- Blank: empty project, without folders/resources structure (only if «advanced mode» is enabled)

**Important:** this selection does not limit in the deployment capabilities or available features

## Project configuration

- Name, Location
- Use version control: allow the project versioning and collaboration between developers (GIT)
- Encrypt secrets: passwords are encrypted via a private key stored in the FactoryTalk HUB domain
- Default screens resolution (can be changed later)



# Development environment

1. Wizards

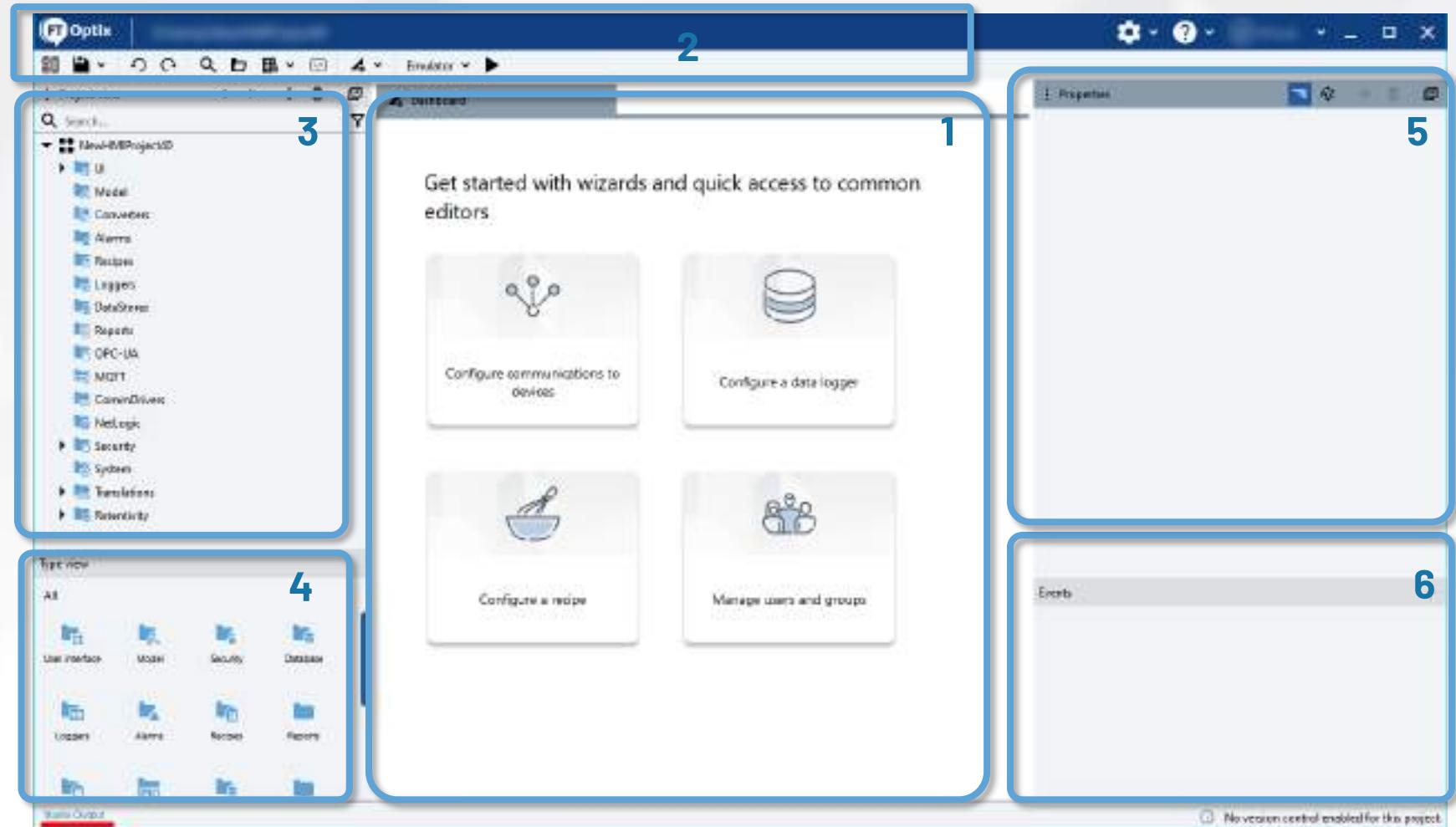
2. Main Toolbar

3. Project View panel

4. Types View panel

5. Properties panel

6. Events panel



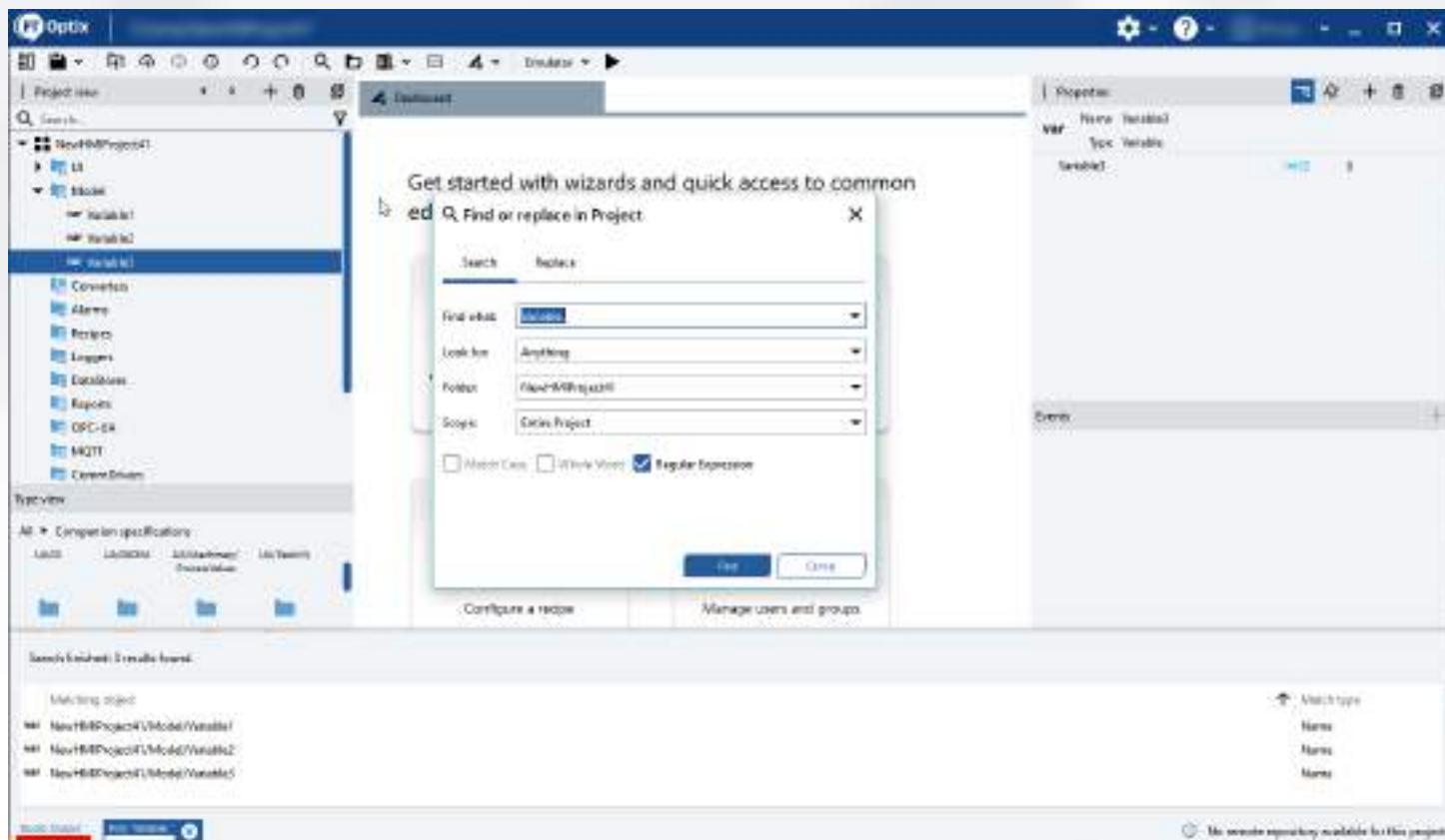
# Main toolbar



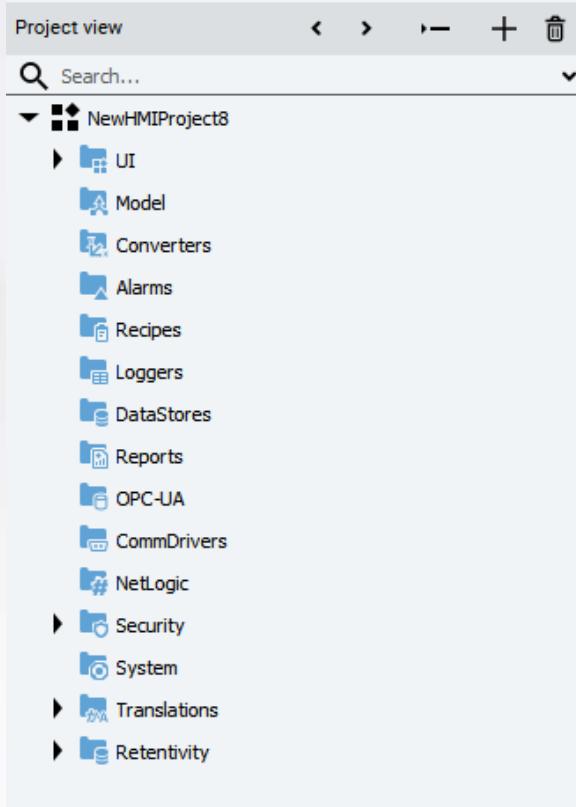
- **Collaboration:** Save & Commit/Push/Pull/History
- **Undo/Redo**
- **Global search**
- **Project Files**
- **Libraries** (%UserProfile%\Documents\Rockwell Automation\FactoryTalk Optix\Libraries)
- **NetSolution**
- **Wizards**
- **Deploy options:** Run the project within Emulator or transfer and run on a Target
- **Studio Options**

# Main toolbar

- **Global Search:** helps finding elements anywhere in the project
  - Supports RegEx (using POSIX syntax, e.g: «.\*» to search for any characters)
  - Can search in NetLogic, project tree and values
  - Output is returned in the a dedicated tab of the IDE
  - Supports replacing text and values



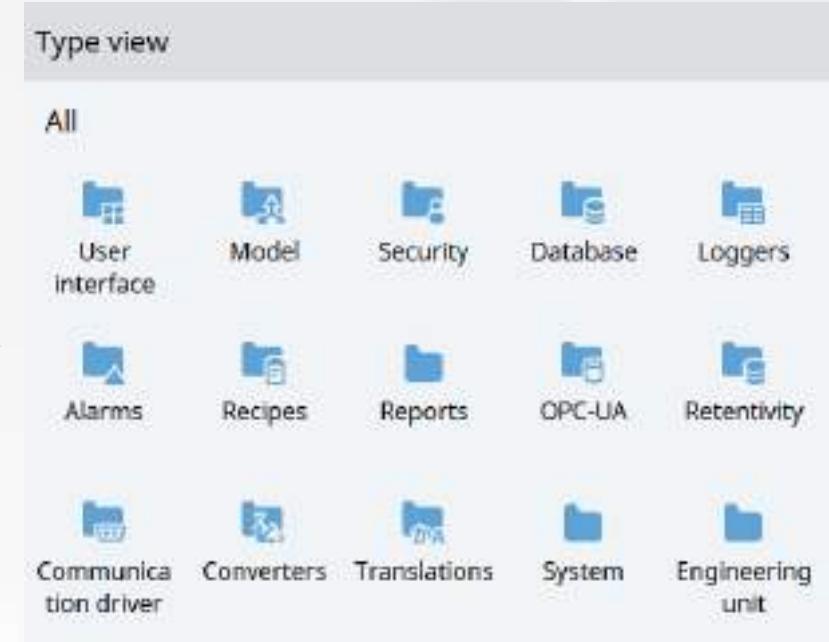
# Project view



- The Project view shows, in a **typical tree architecture**, the project folders
- Folder structure constitutes the foundation of a project and are organized by type and functionality
- The Project view tree supports:
  - Context commands with the mouse (right-click)
  - Multiple selections
  - Drag&Drop
  - Copy/Paste
  - Search and filter
  - Detach
- Please, do not move or rename the default folders

# Type view

- Collection of **Native Types**, grouped in folders according to their purpose:
  - **Basic User Control**: to develop the User Interface
  - **Optix Types**: all the objects to be added to the project folders
  - **Companion specifications**: set of standard and custom types definition which are used to share information between different manufacturers (PackML, DI, TMC, etc)
- Object from Type view can be added into the project Tree by Drag&Drop
- Can be also the repository of **Custom Types**



# Properties panel

- This pane shows properties of the selected node
  - Project
  - Resource
  - Objects
- Depending on the type of property, the editing can be done through:
  - Direct editing by typing
  - by Selecting the pencil icon 
  - Drop-down menu
  - by Selecting the DynamicLink icon 
- Can edit common properties of Multiple selections

Properties	
Name	MyProject
Type	Project folder
Locales	en-US
Translation fallback locales	en-US
Branching enabled	False
Measurement systems map	<i>Default mapping</i> 
Authentication mode	Model only
Default user folder	<a href="#">Users</a> 
- Password policy	
Maximum password age	0
Enforce password history	1
Minimum password age	0
Minimum password length	8

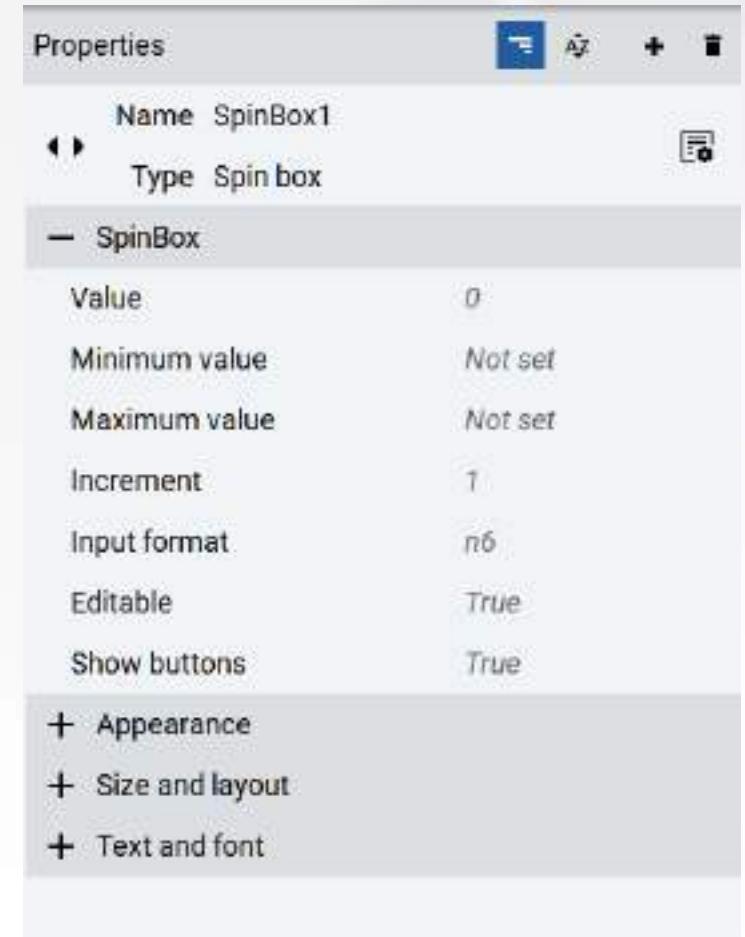
# Properties panel

- This pane shows properties of the selected node
  - Project
  - Resource
  - Objects
- Depending on the type of property, the editing can be done through:
  - Direct editing by typing
  - by Selecting the pencil icon 
  - Drop-down menu
  - by Selecting the DynamicLink icon 
- Can edit common properties of Multiple selections

Properties	
Name	Screen1 (type)
Type	Panel
Visible	True
Enabled	True
Opacity	100
Rotation	0
Hit test visible	False
— Size and layout	
Horizontal alignment	Left
Vertical alignment	Top
Width	300
Height	300
Left margin	0
Top margin	0

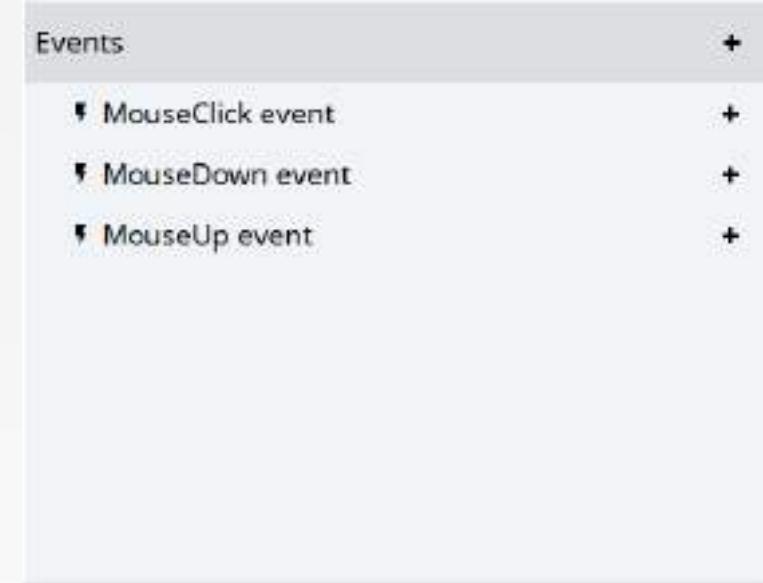
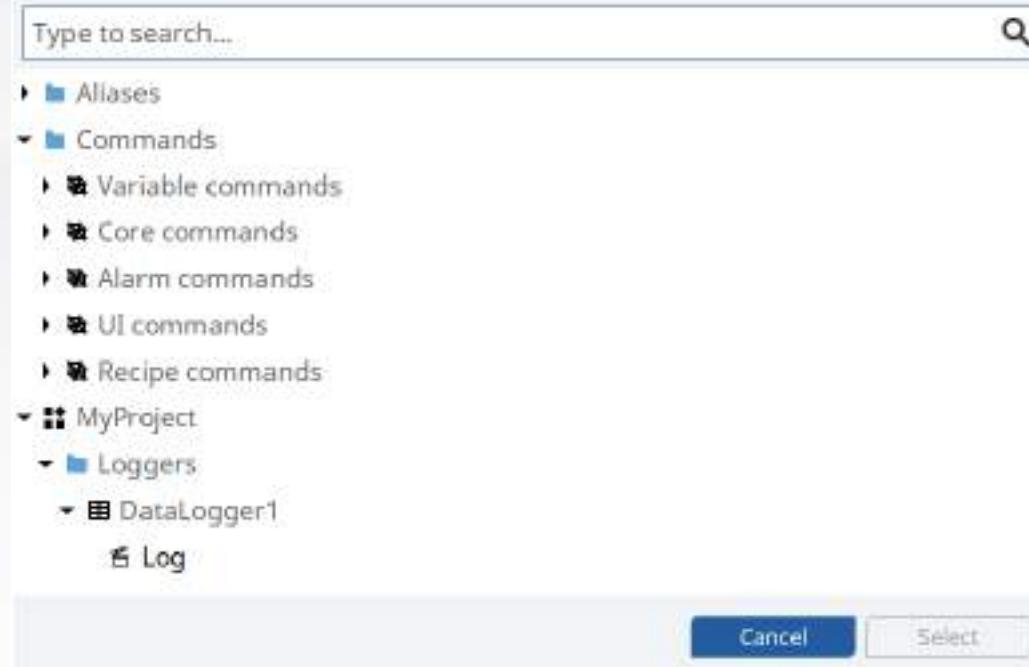
# Properties panel

- This pane shows properties of the selected node
  - Project
  - Resource
  - Objects
- Depending on the type of property, the editing can be done through:
  - Direct editing by typing
  - by Selecting the pencil icon 
  - Drop-down menu
  - by Selecting the DynamicLink icon 
- Can edit common properties of Multiple selections



# Events panel

- Used to configure Commands or Methods to be executed when events are triggered by the selected object
- Different objects have different events



# How to create a simple HMI project in few steps



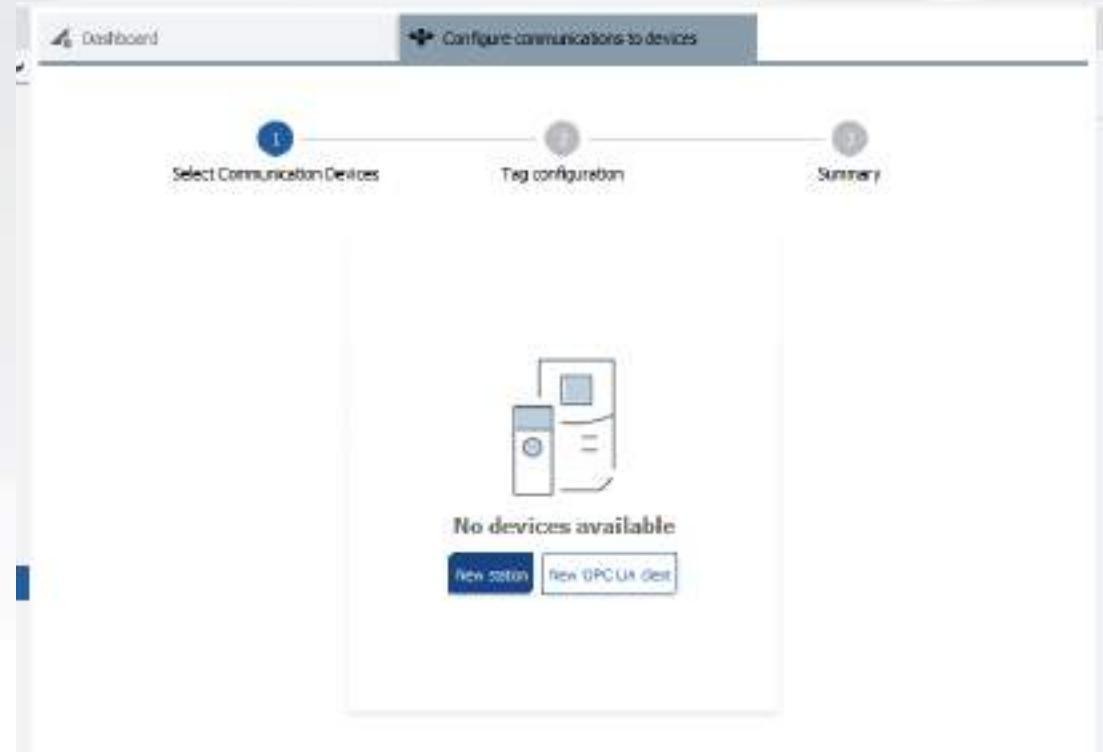
# Create a simple HMI project in few steps

## 1. Configure connected device



# Create a simple HMI project in few steps

## 1. Configure connected device



# Create a simple HMI project in few steps

## 1. Configure connected device

The screenshot shows a software interface for configuring communication devices. At the top, there's a navigation bar with 'Dashboard' and 'Configure communications to devices'. Below that, a progress bar indicates three steps: 'Select Communication Devices' (step 1), 'Tag configuration' (step 2), and 'Summary' (step 3). The main area is titled 'Select protocol' and shows two options: 'ComDriver' (selected) and 'OPC UA Client'. A section labeled 'New station' contains a 'Name' field with the value 'RAEtherNet\_IPStation1'. Below this, a 'Type' section lists several station types with radio buttons, and 'RA EtherNet/IP Station' is selected.

Select Communication Devices

Select protocol

ComDriver    OPC UA Client

New station

Name: RAEtherNet\_IPStation1

Type:

CODESYS Station

MELSEC FX3U Station

MELSEC Q station

Modbus Station

OMRON EtherNet/IP station

OMRON FXn Station

RA EtherNet/IP Station

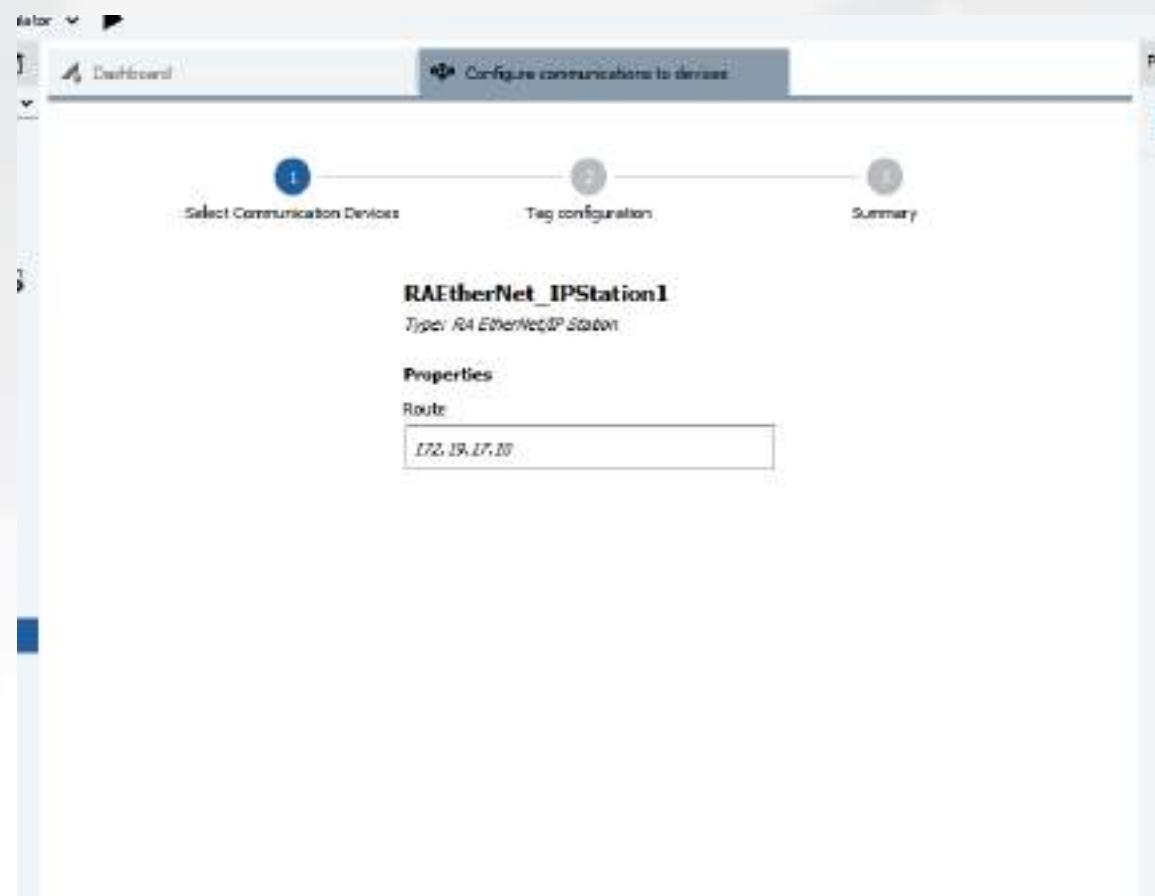
S7TOP Station

S7 TIA PROFINET station

TwinCAT station

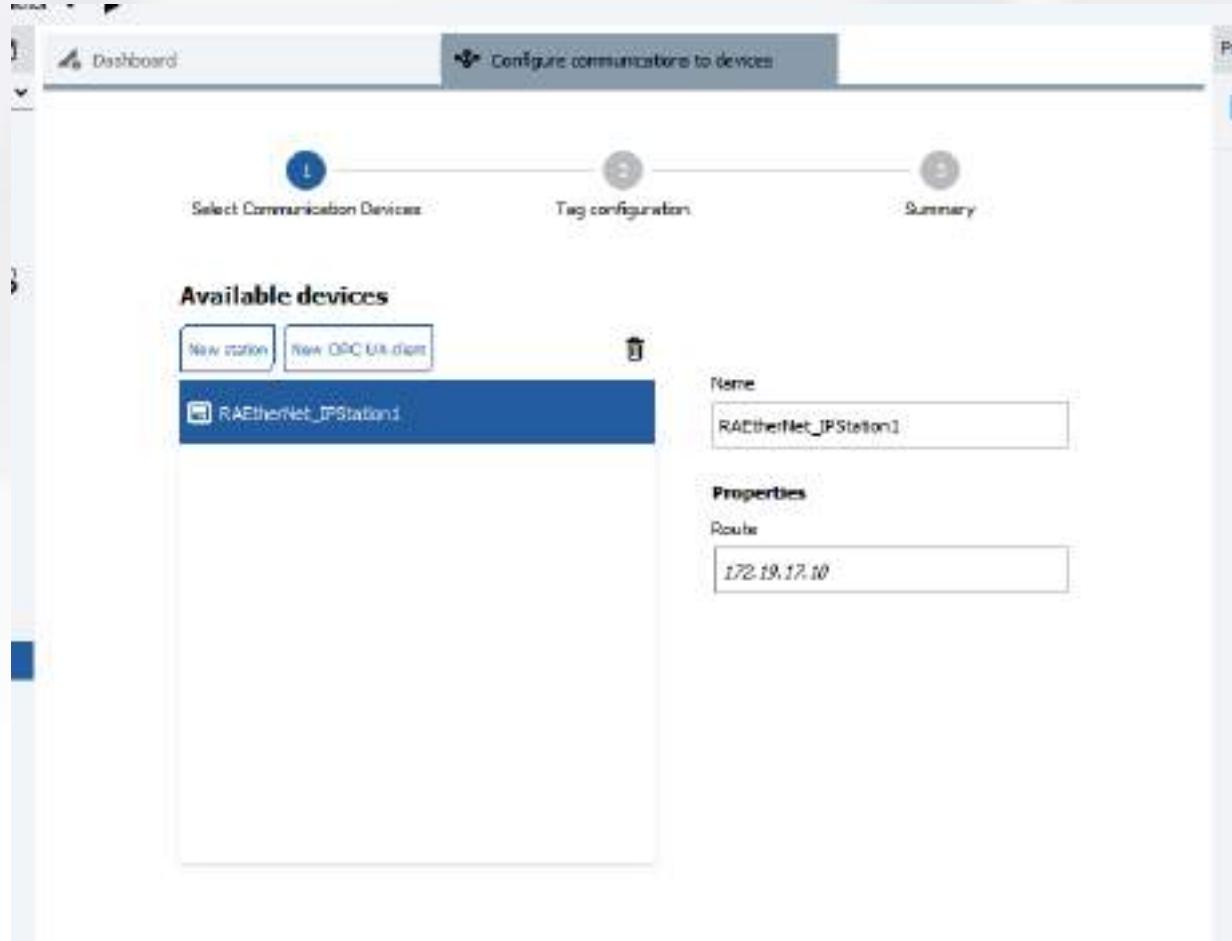
# Create a simple HMI project in few steps

## 1. Configure connected device



# Create a simple HMI project in few steps

## 1. Configure connected device



# Create a simple HMI project in few steps

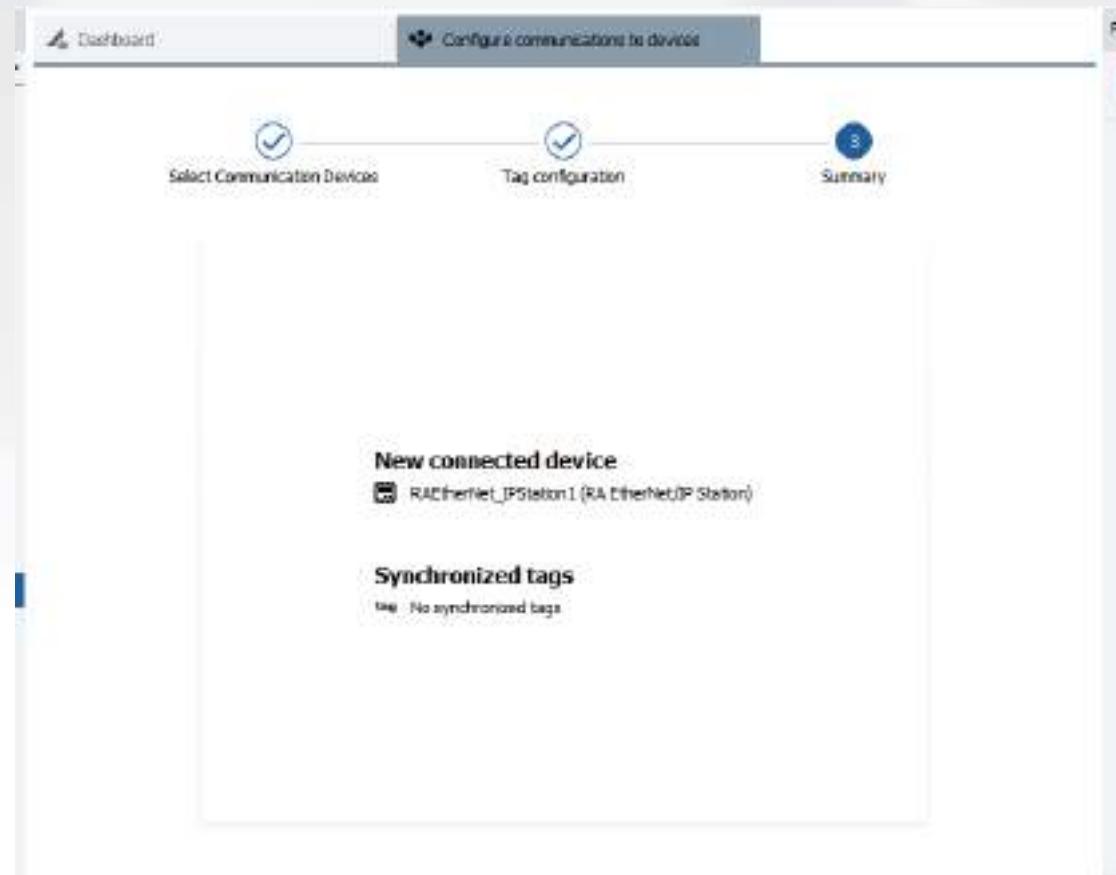
## 1. Configure connected device

The screenshot shows the 'Tag configuration' step of a project setup process. At the top, there are three tabs: 'Select Communication Devices' (highlighted with a checkmark), 'Tag configuration' (highlighted with a blue circle and the letter 'z'), and 'Summary'. Below the tabs, the title 'RAEtherNet\_IPStation1' is followed by a link 'Select in property view'. A status indicator shows 'Online' with a green light. A search bar contains the placeholder 'Search...'. The main area displays a table of tags:

Tag	Type	Description
Tag Motors	Motor	Motor [5]
Tag Tank1	TankUDT	TankUDT
Tag Tank2	TankUDT	TankUDT
Tag TestBoolTag	RA EtherNet/IP Tag	Boolean
Tag TestIncrementTag	RA EtherNet/IP Tag	Int32
Program:ACI_LocalTagsProgram	Folder	

# Create a simple HMI project in few steps

## 1. Configure connected device



# Create a simple HMI project in few steps

## 2. Configure a datalogger

Let's begin building your user interface



I want to configure  
connected devices



I want to configure a data  
logger



I want to configure a recipe



I want to manage users and  
groups

# Create a simple HMI project in few steps

## 2. Configure a datalogger

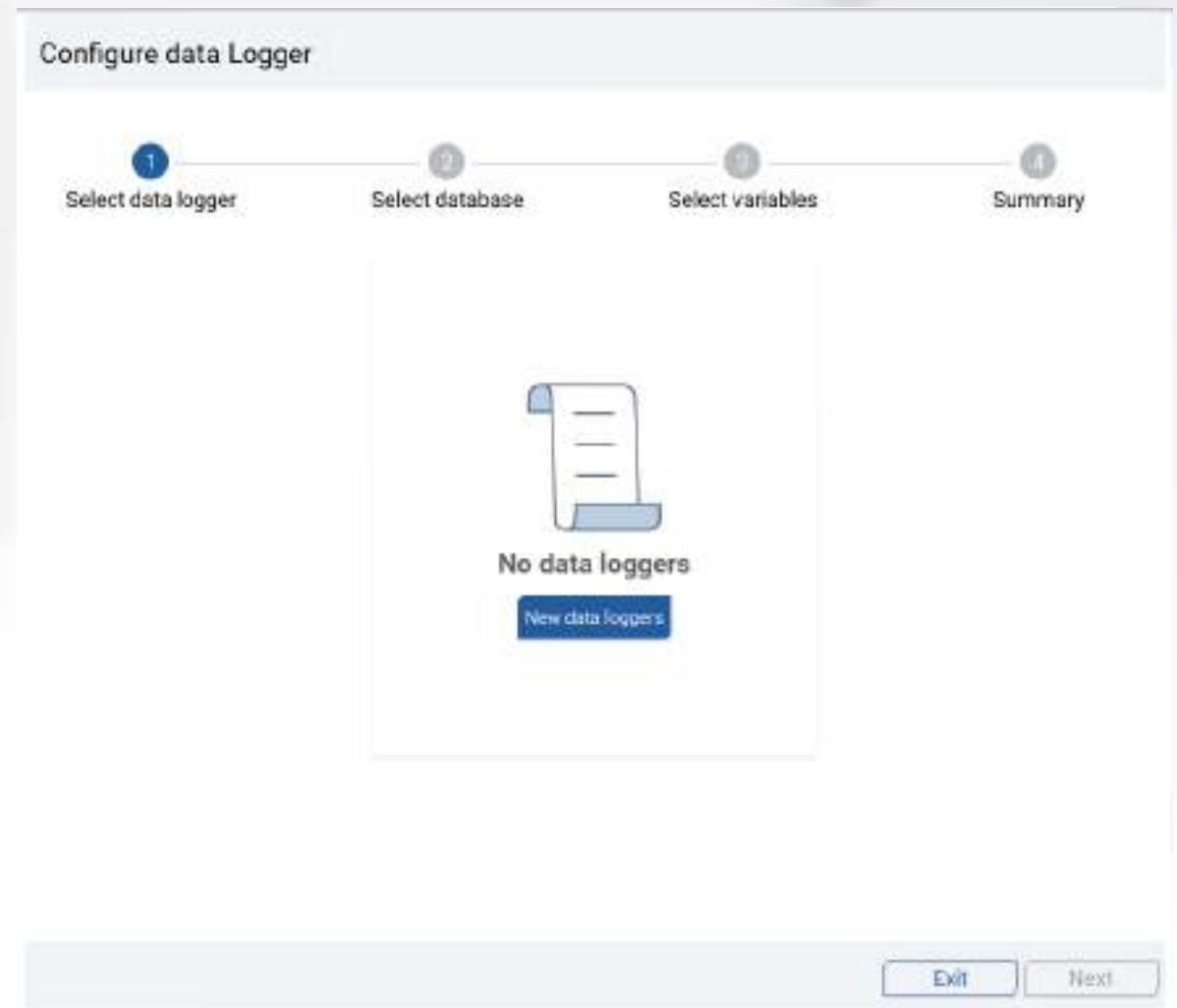
Configure data Logger

1 Select data logger    2 Select database    3 Select variables    4 Summary

No data loggers

New data loggers

Exit    Next



# Create a simple HMI project in few steps

## 2. Configure a datalogger

Configure data Logger

1 Select data logger    2 Select database    3 Select variables    4 Summary

New data logger

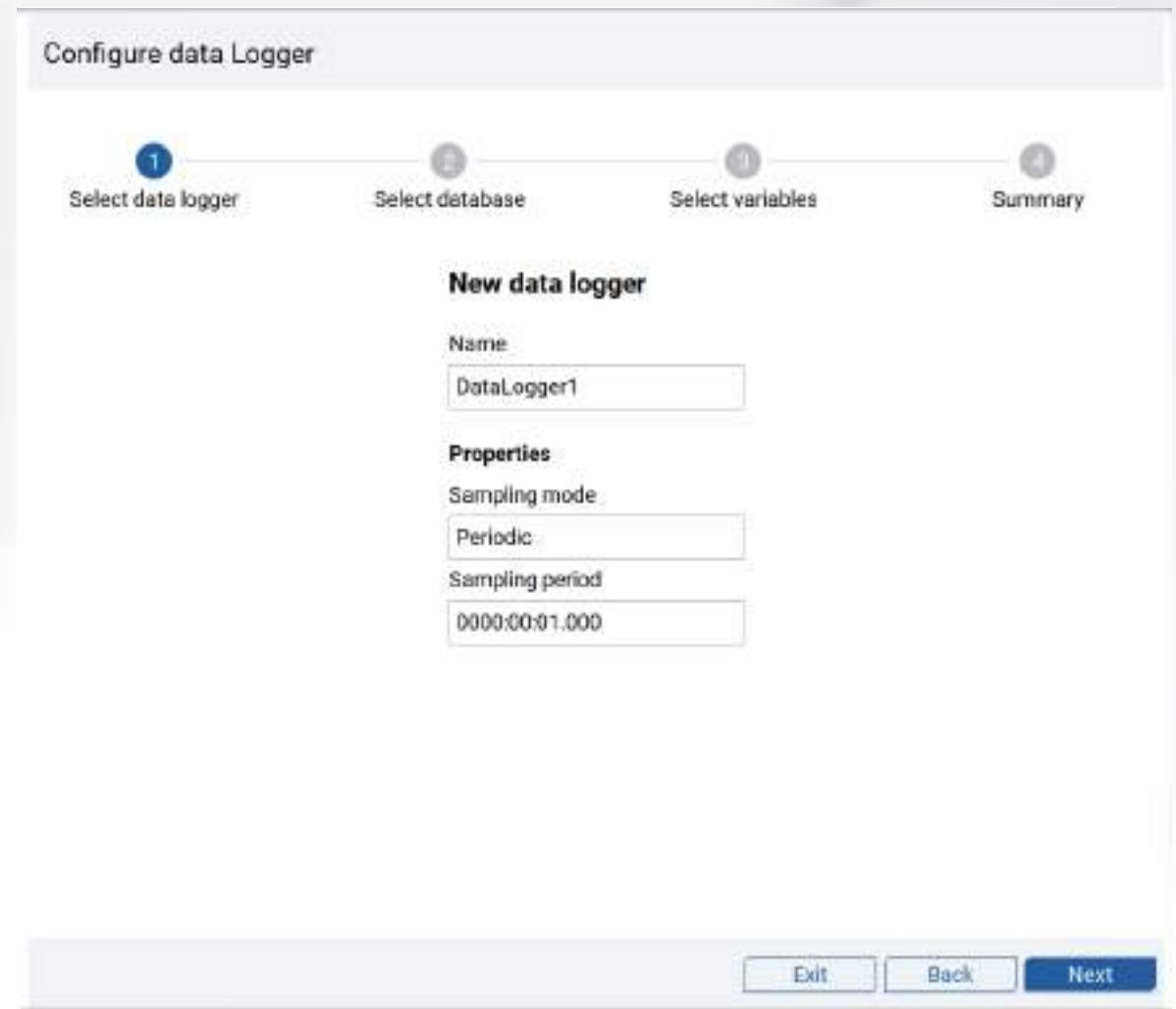
Name: DataLogger1

Properties

Sampling mode: Periodic

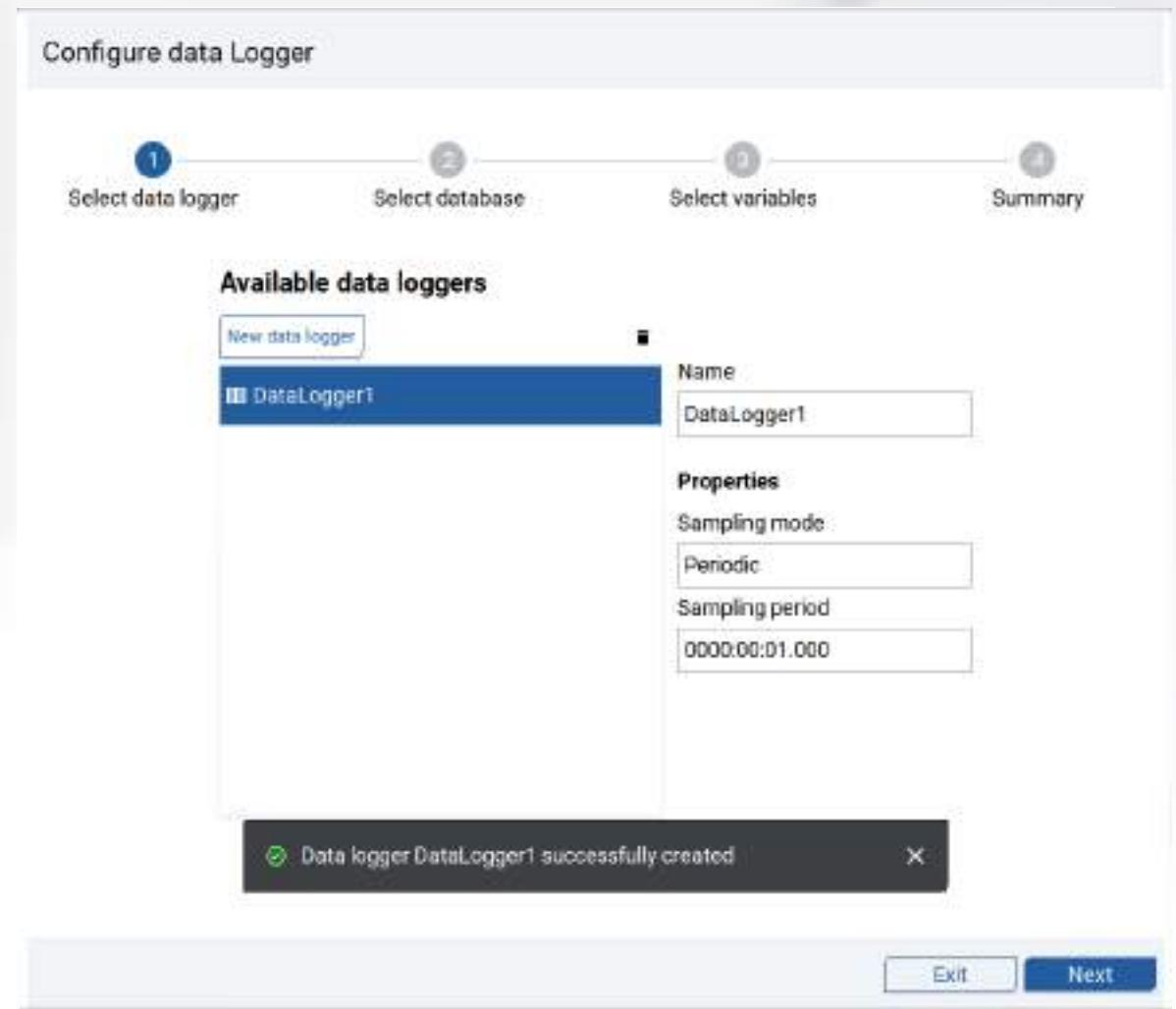
Sampling period: 0000:00:01.000

Exit    Back    Next



# Create a simple HMI project in few steps

## 2. Configure a datalogger



# Create a simple HMI project in few steps

## 2. Configure a datalogger

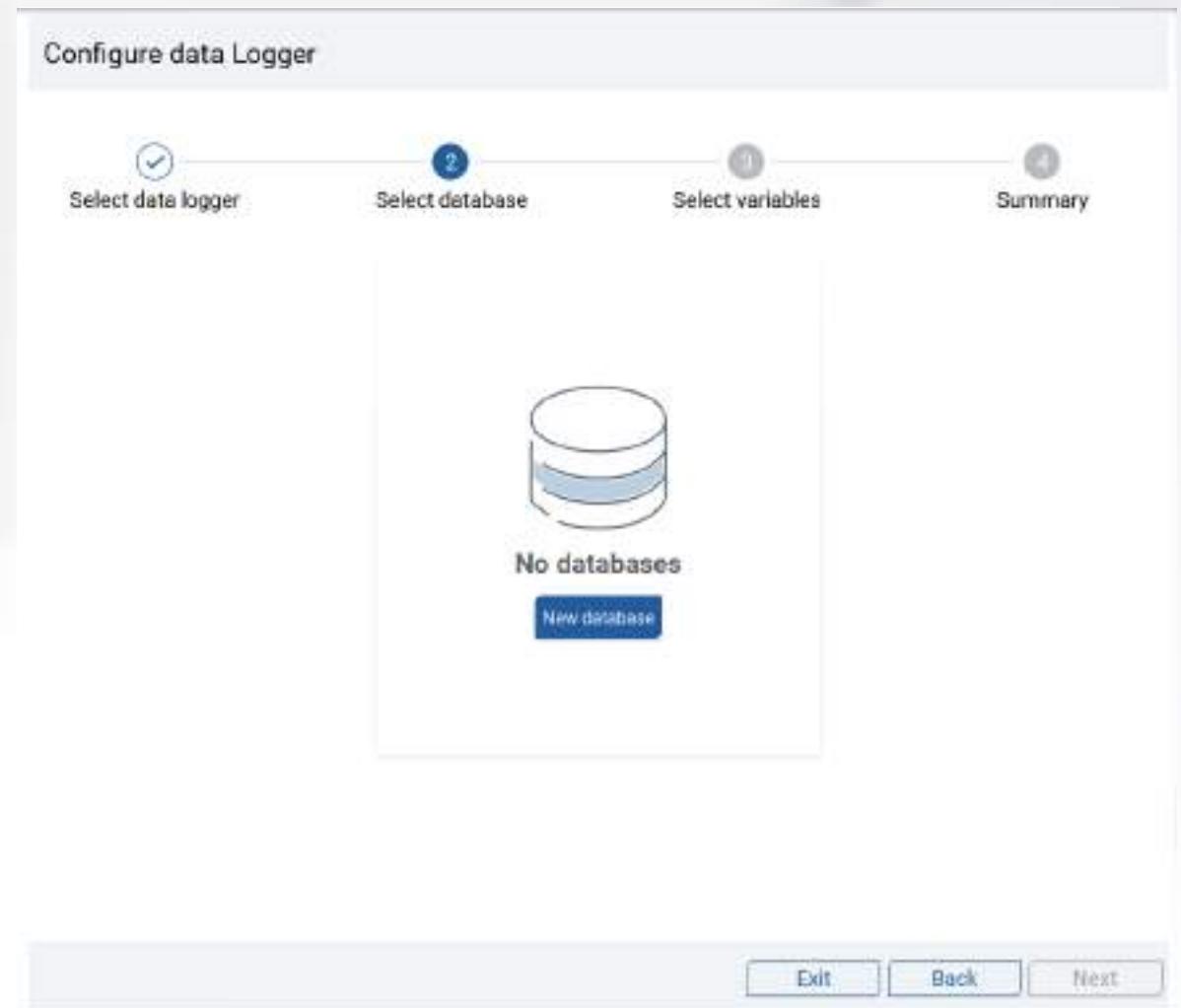
Configure data Logger

1 Select data logger    2 Select database    3 Select variables    4 Summary

No databases

New database

Exit    Back    Next



# Create a simple HMI project in few steps

## 2. Configure a datalogger

Configure data Logger

1 Select data logger    2 Select database    3 Select variables    4 Summary

New database

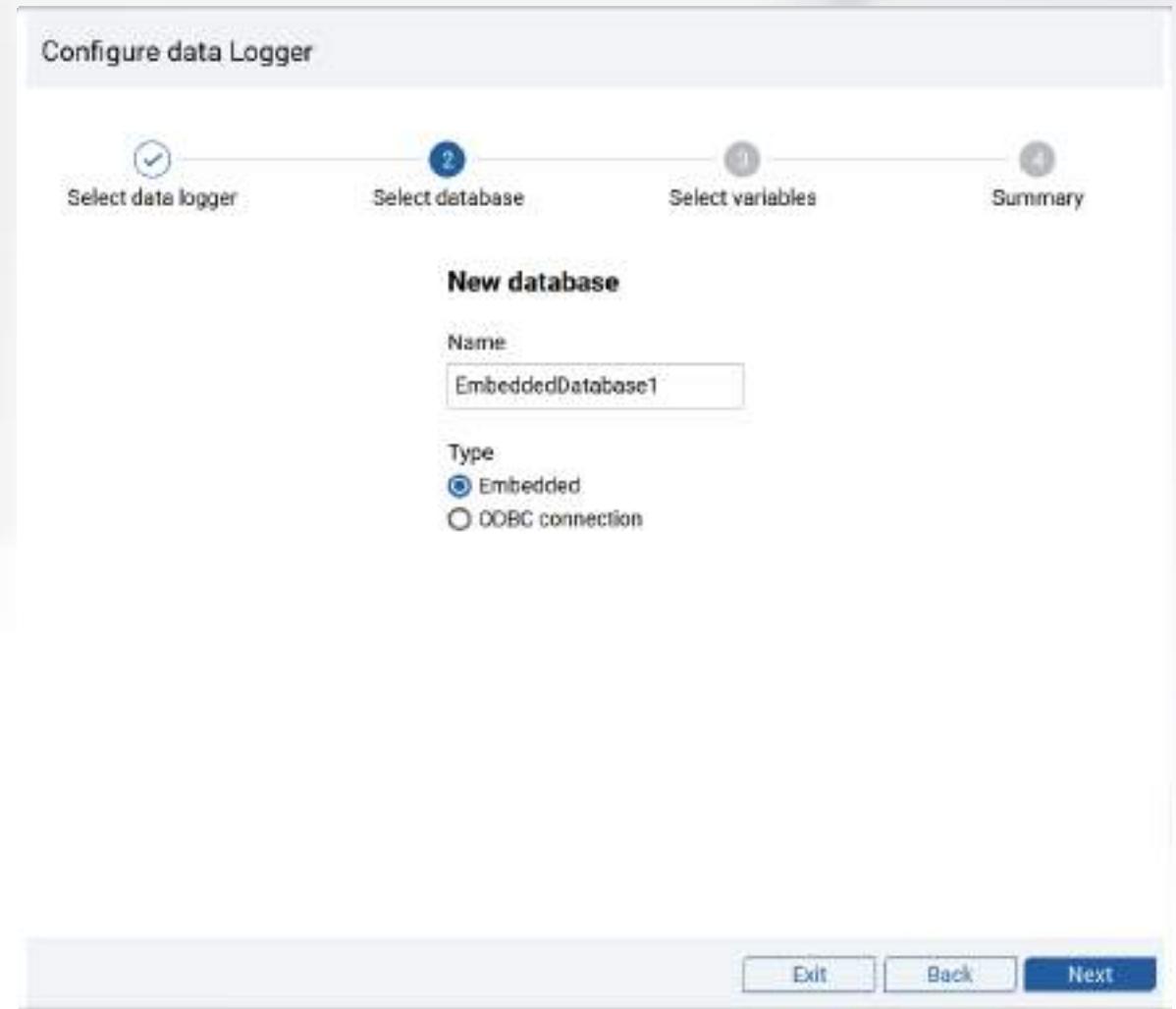
Name:

Type:

Embedded

ODBC connection

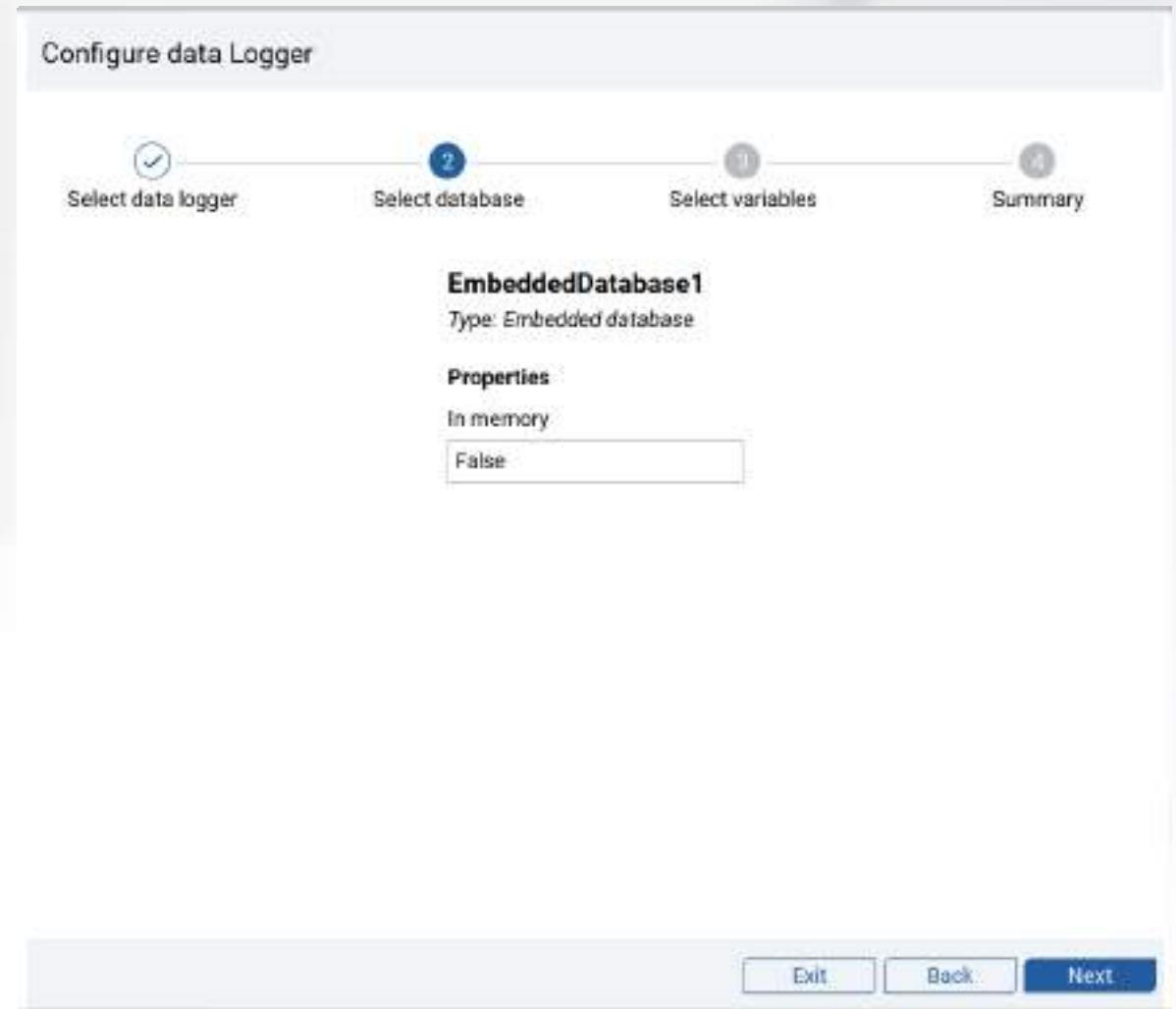
Exit    Back    Next



# Create a simple HMI project in few steps

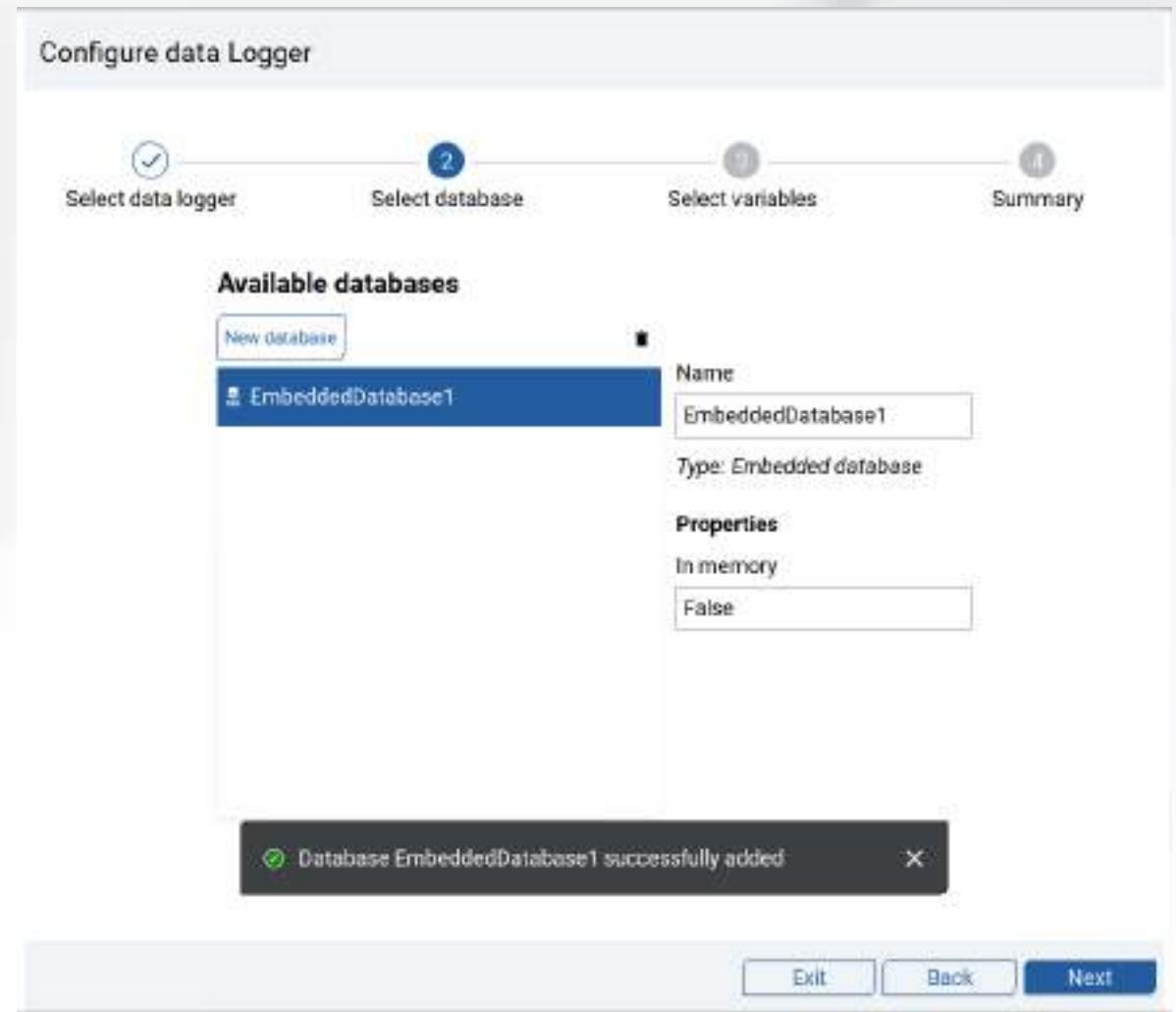
## 2. Configure a datalogger

- In memory: data are put in RAM, content of the DB is lost if the application is restarted



# Create a simple HMI project in few steps

## 2. Configure a datalogger



# Create a simple HMI project in few steps

## 2. Configure a datalogger

Configure data Logger

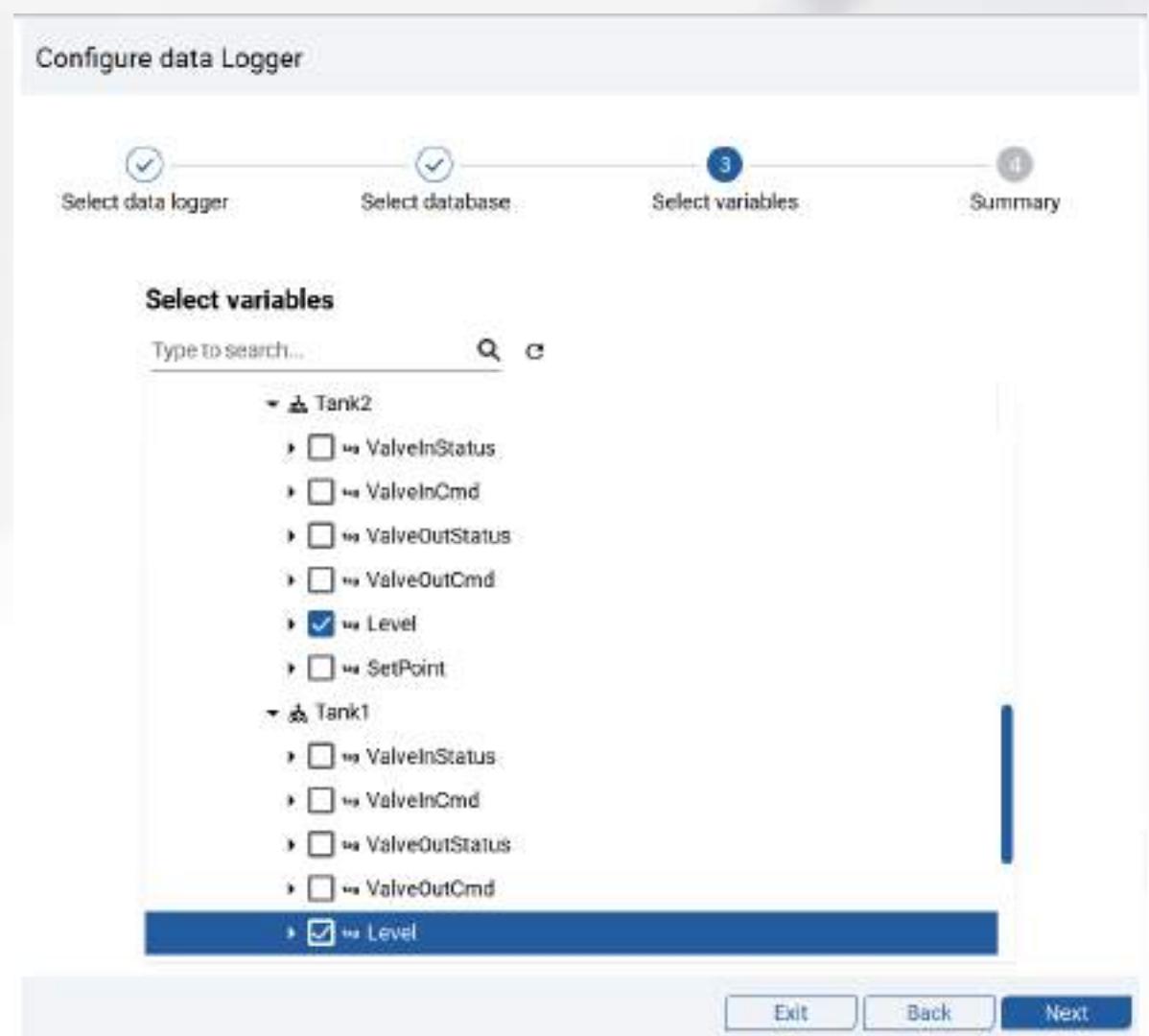
1 Select data logger   2 Select database   3 Select variables   4 Summary

Select variables

Type to search...  🔍 ✖

- ▾ Tank2
  - >  ↳ ValveInStatus
  - >  ↳ ValveInCmd
  - >  ↳ ValveOutStatus
  - >  ↳ ValveOutCmd
  - >  ↳ Level
  - >  ↳ SetPoint
- ▾ Tank1
  - >  ↳ ValveInStatus
  - >  ↳ ValveInCmd
  - >  ↳ ValveOutStatus
  - >  ↳ ValveOutCmd
  - >  ↳ Level

Exit Back Next



# Create a simple HMI project in few steps

## 2. Configure a datalogger

Configure data Logger

Select data logger    Select database    Select variables    Summary 4

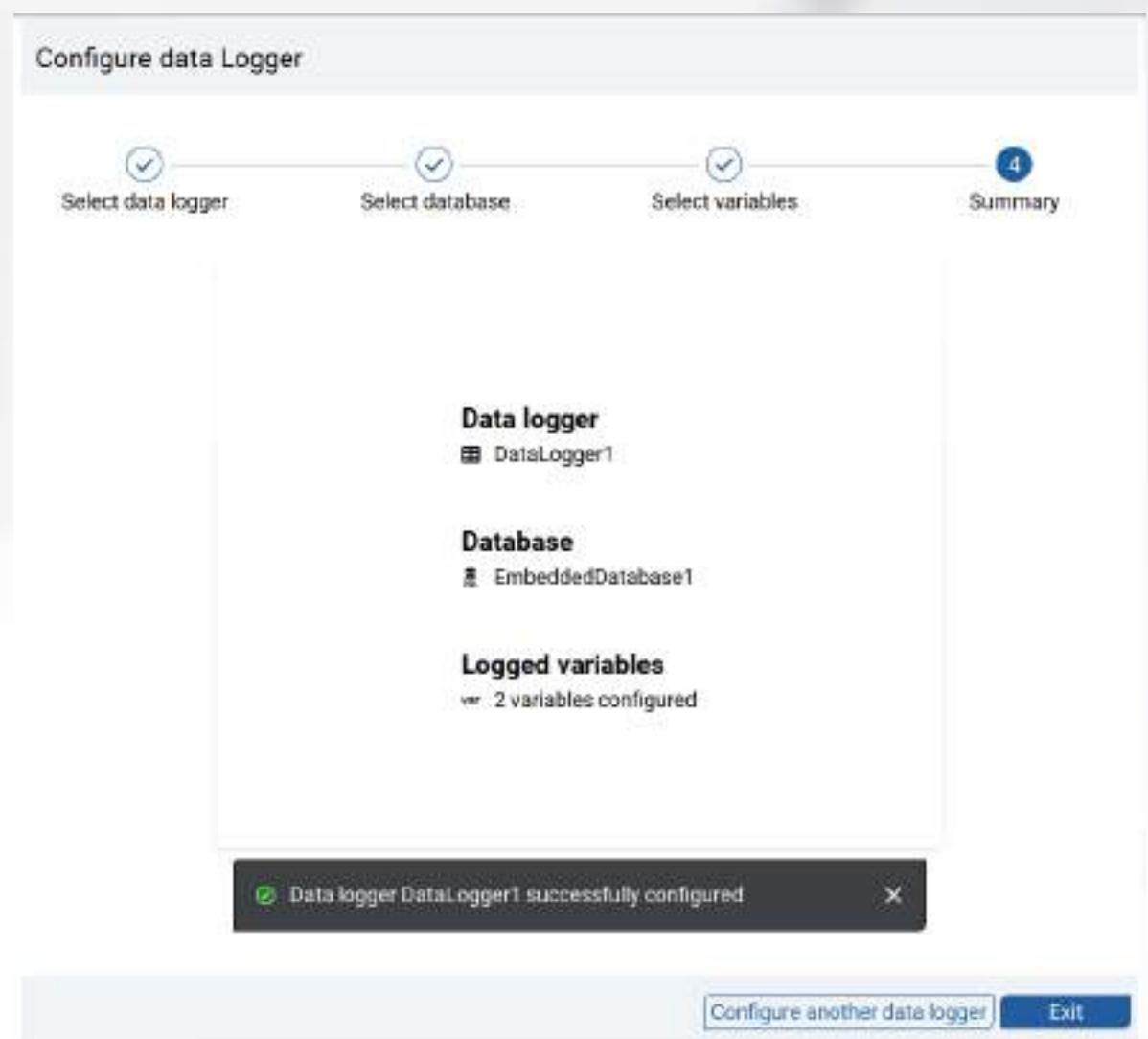
**Data logger**  
DataLogger1

**Database**  
EmbeddedDatabase1

**Logged variables**  
2 variables configured

**Success message:** Data logger DataLogger1 successfully configured

Configure another data logger    Exit



# Create a simple HMI project in few steps

## 3. Configure the user interface

Let's begin building your user interface



I want to configure  
connected devices



I want to configure a data  
logger



Preview

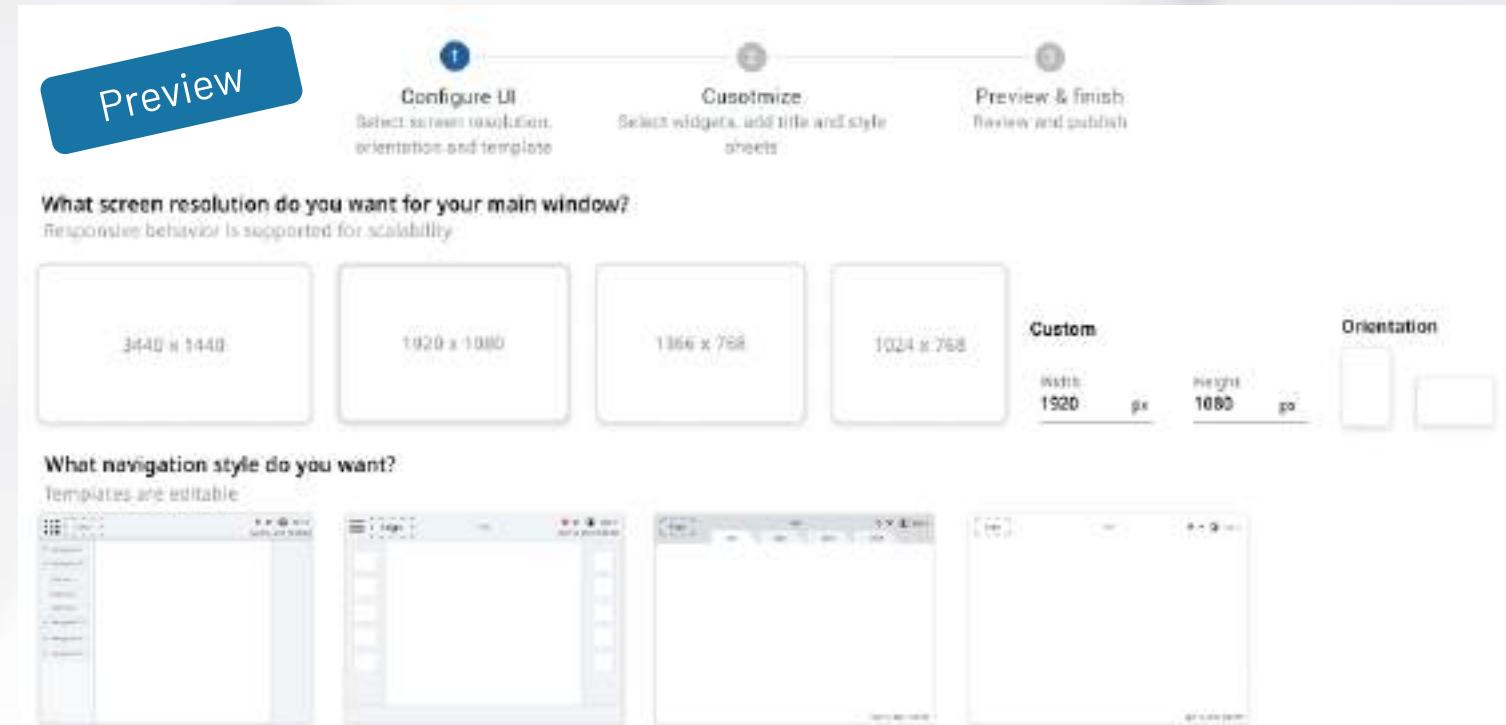
Configure my UI displays  
starting with main window



I want to manage users and  
groups

# Create a simple HMI project in few steps

## 3. Configure the user interface



**How many parent pages would you like to create?**

Three children pages are included with each parent page.

Number of pages:

- 000 +

**How many tabs would you like to create?**

Tabs are editable.

Number of tabs:

- 000 +

# Web IDE



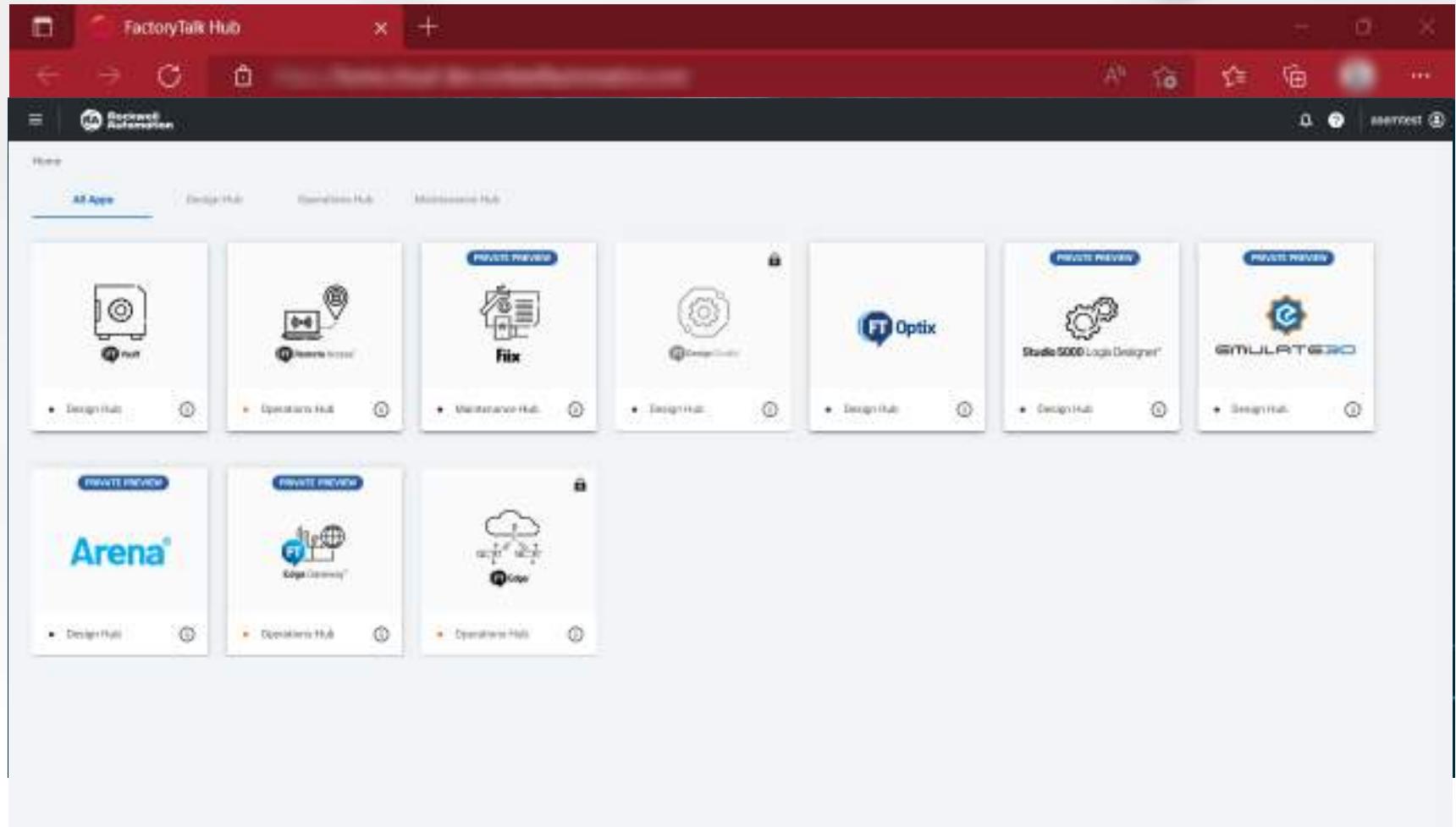
# Web IDE using FactoryTalk Hub

- Sign in to FactoryTalk Hub
- Click on the FactoryTalk Optix tile
- Select which build to launch on Web IDE or continue with the last used version



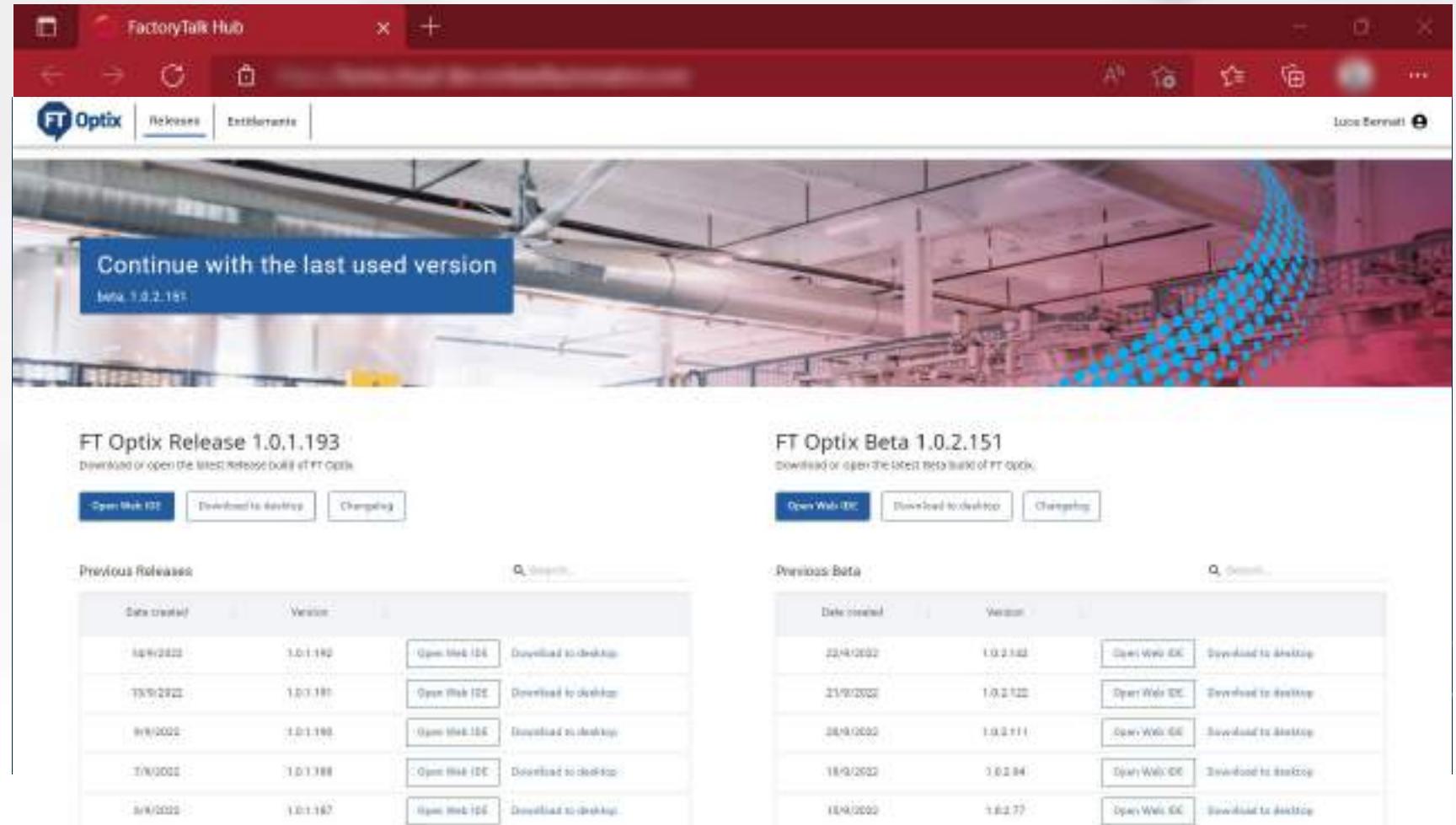
# Web IDE using FactoryTalk Hub

- Sign in to FactoryTalk Hub
- Click on the FactoryTalk Optix tile
- Select which build to launch on Web IDE or continue with the last used version



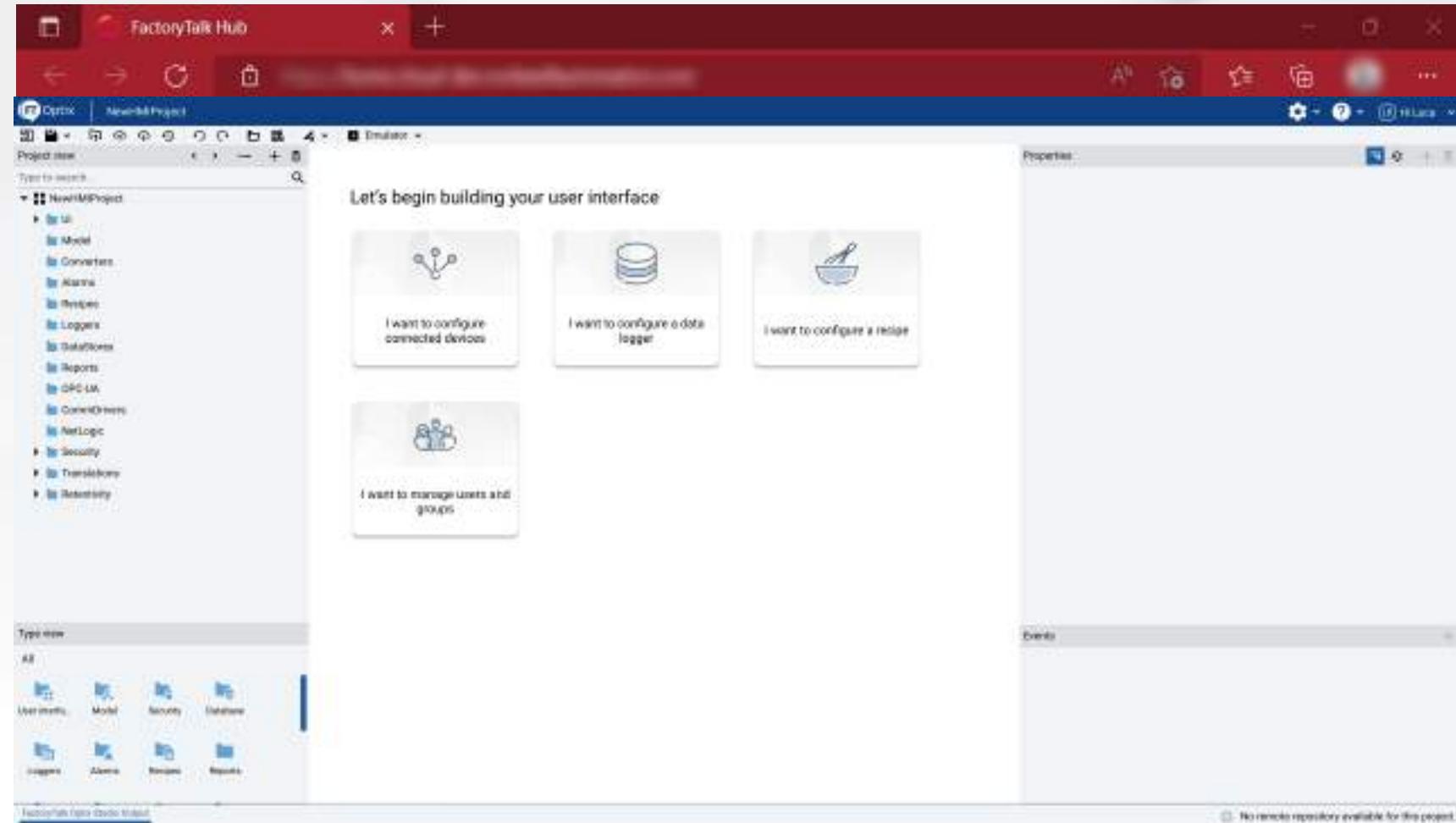
# Web IDE using FactoryTalk Hub

- Sign in to FactoryTalk Hub
- Click on the FactoryTalk Optix tile
- Select which build to launch on Web IDE or continue with the last used version



# Web IDE using FactoryTalk Hub

- Sign in to FactoryTalk Hub
- Click on the FactoryTalk Optix tile
- Select which build to launch on Web IDE or continue with the last used version



# Project versions and components



# Project compatibility and backup

- **Upgrade:**

- Projects created with an old version of Studio **can be opened with later versions**
- Studio automatically updates the project and all its modules to the current version

- **Downgrade:**

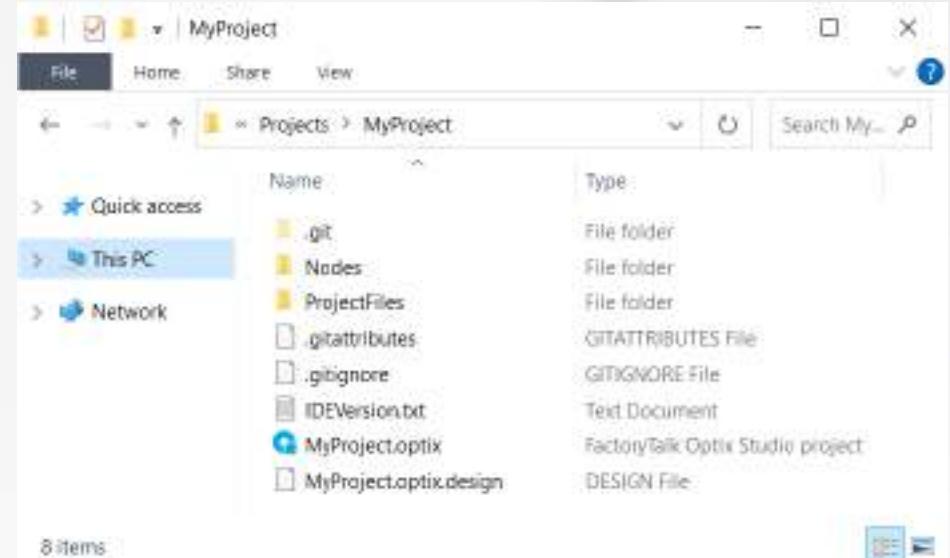
- Projects created with newer versions of Studio **cannot be opened with older versions**

- **Backups:** %localappdata%\Rockwell Automation\FactoryTalk Optix\FTOptixStudio\Backups\

- Automatic: contain the last three backups, performed when saving the project
- Upgrade: performed when updating the project to make it compatible with a newer version
- Recovery: performed when a corrupt project is opened, and repair is attempted

# Project structure

- Contents of the project folder
  - **.optix** file: information on the project's elements
  - **.optix.design** file: information on accessory elements only required at design time.
  - **Nodes** folder: contains a YAML file for each node that describes its related information model
  - **ProjectFiles** folder: containing all files used in the project, including images, fonts and documents.
    - NetSolution subfolder contains all files used to create customized C# scripts
    - PKI subfolder: Public Key Infrastructure folder, used to manage security certificates for OPC-UA

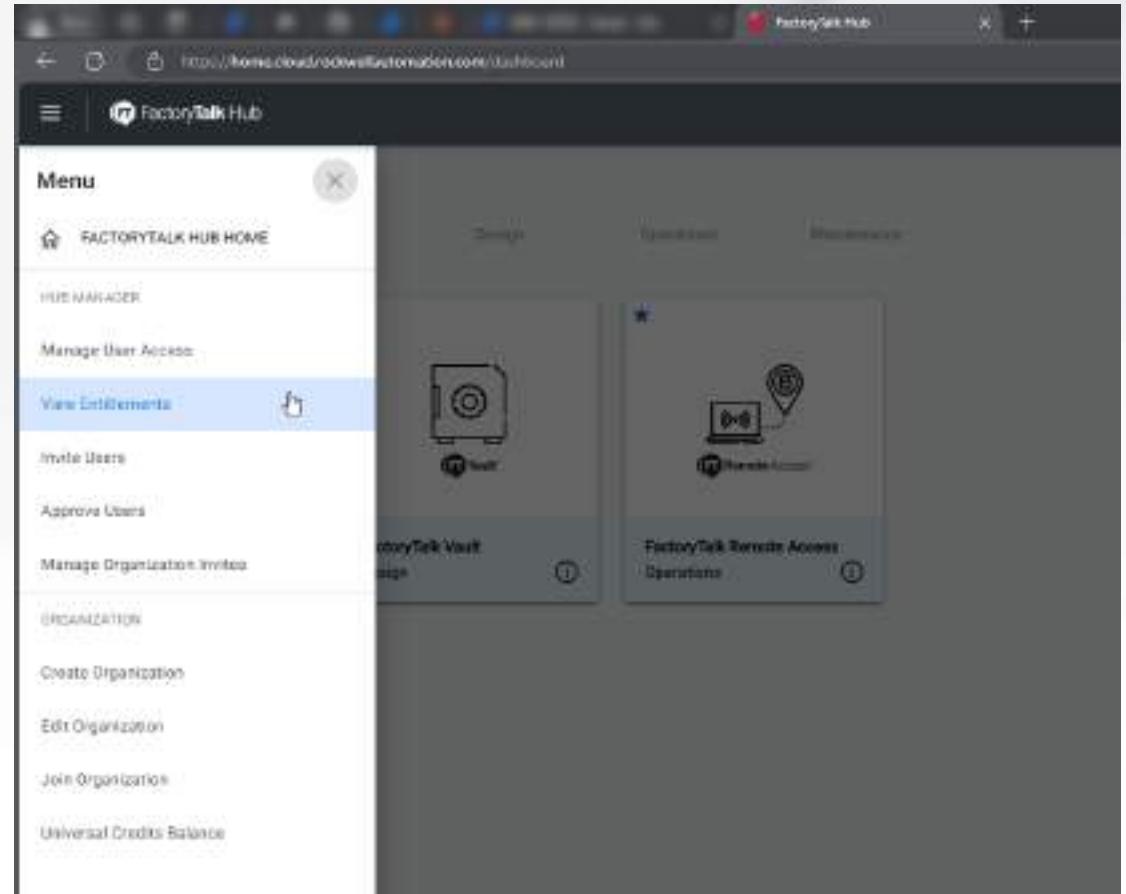


# Factory Talk Optix Entitlements



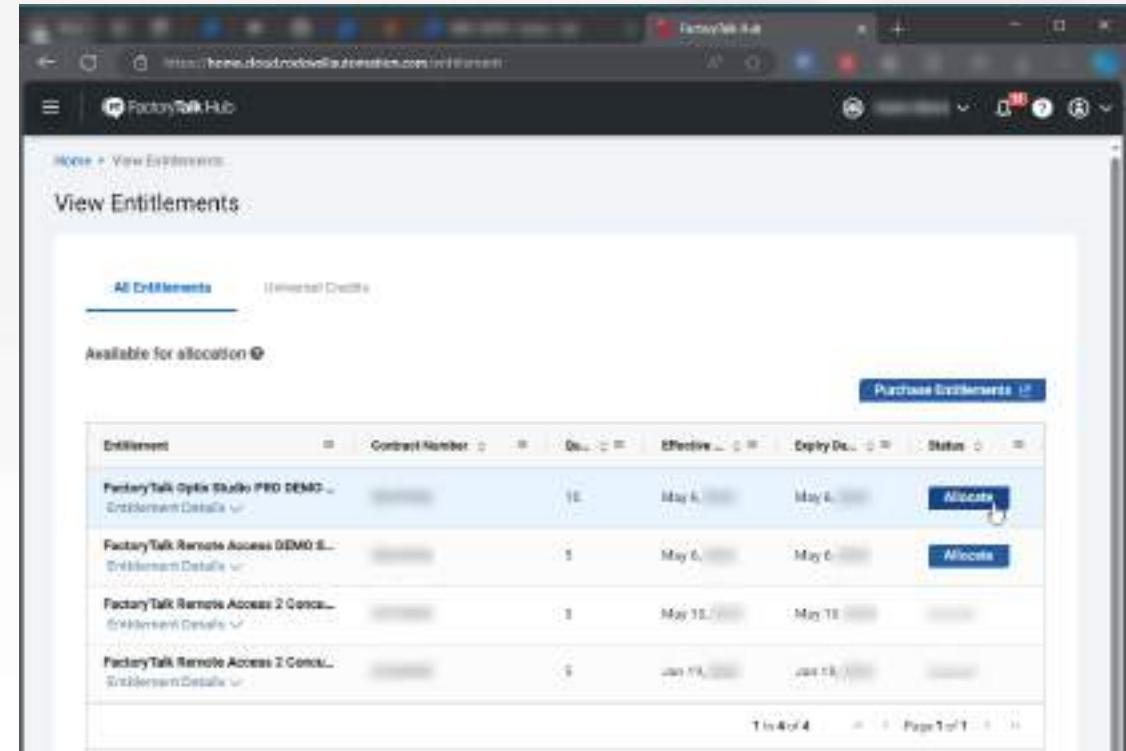
# FactoryTalk Optix Entitlements

- Licenses are called «Entitlements»
- Entitlements are assigned to the user that bought the license by accessing the «Entitlements» menu of FactoryTalk Hub
- Entitlements have to be assigned to a FactoryTalk Optix organization



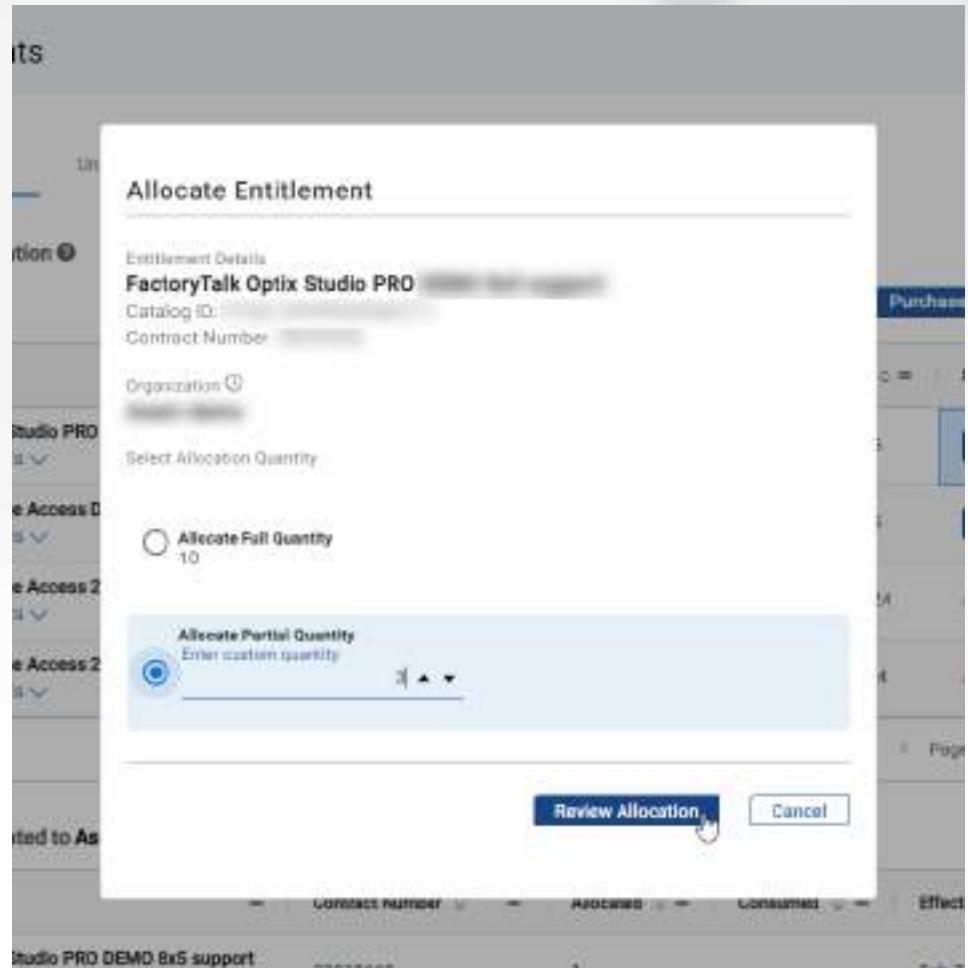
# FactoryTalk Optix Entitlements

- Licenses are called «Entitlements»
- Entitlements are assigned to the user that bought the license by accessing the «Entitlements» menu of FactoryTalk Hub
- Entitlements have to be assigned to a FactoryTalk Optix organization



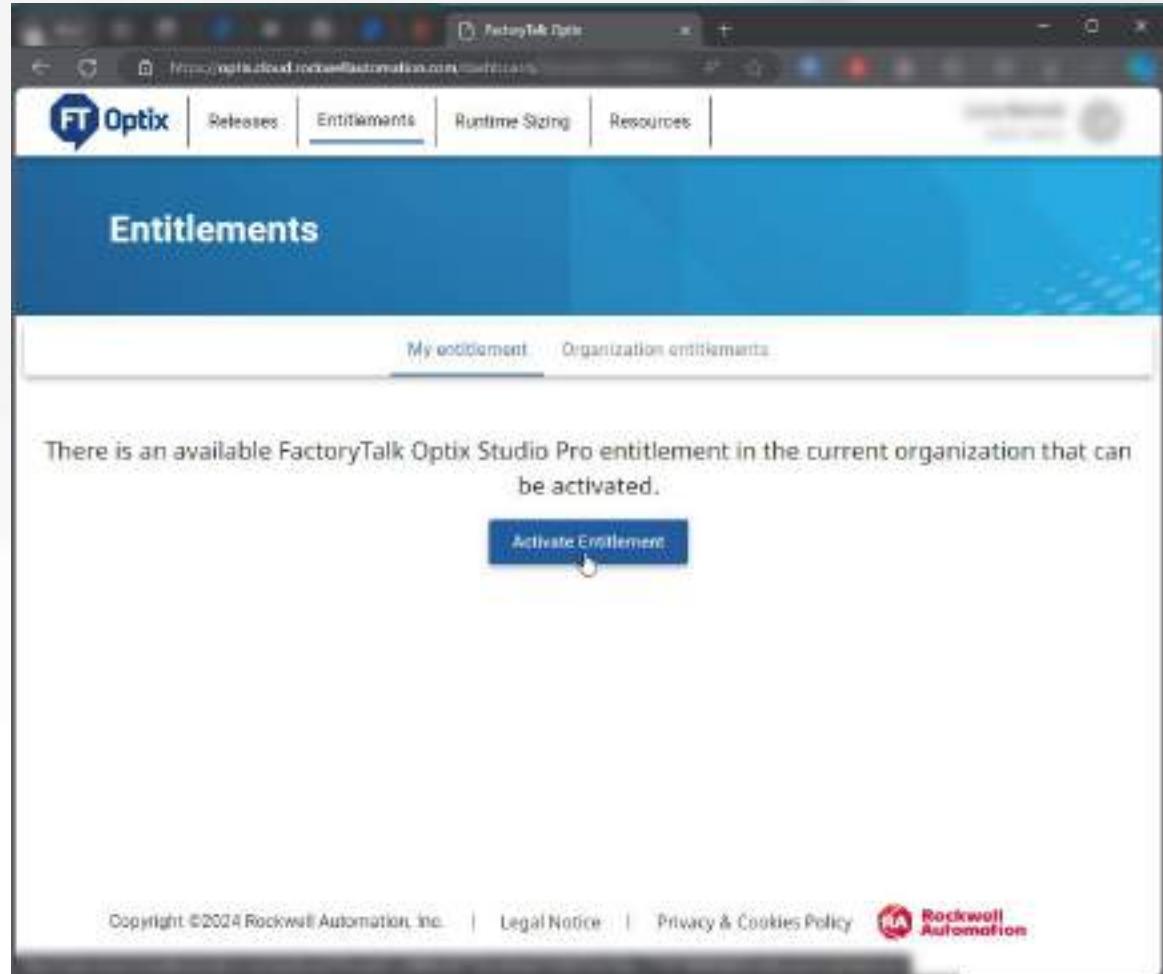
# FactoryTalk Optix Entitlements

- Licenses are called «Entitlements»
- Entitlements are assigned to the user that bought the license by accessing the «Entitlements» menu of FactoryTalk Hub
- Entitlements have to be assigned to a FactoryTalk Optix organization



# FactoryTalk Optix Studio Entitlements

- Users of an organization can self-assign any available Studio entitlement from FactoryTalk Optix web page
- Studio Entitlements can be released once they are not needed anymore and used by a different user in the same organization either from FactoryTalk Optix Studio or the web dashboard
- Studio Entitlements can only be rehosted by the user that has is using the license or by an administrator of the organization



# FactoryTalk Optix Studio Entitlements

- Users of an organization can self-assign any available Studio entitlement from FactoryTalk Optix web page
- Studio Entitlements can be released once they are not needed anymore and used by a different user in the same organization either from FactoryTalk Optix Studio or the web dashboard
- Studio Entitlements can only be rehosted by the user that has is using the license or by an administrator of the organization

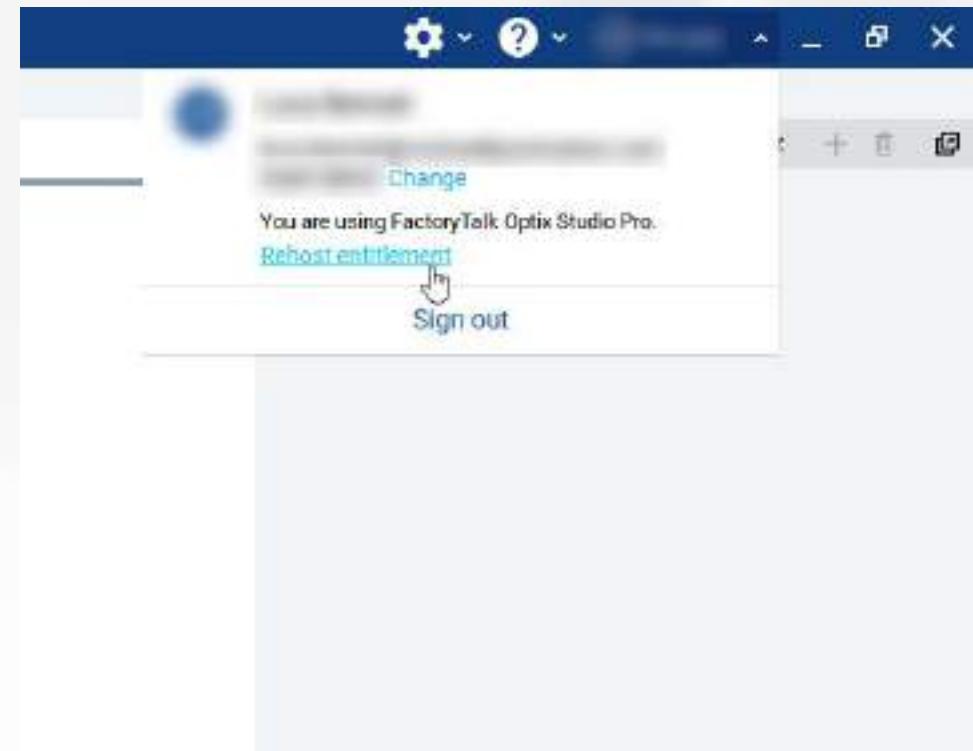
The screenshot shows the 'Entitlements' page of the FactoryTalk Optix web interface. At the top, there's a navigation bar with tabs: 'Releases', 'Entitlements' (which is active), 'Runtime Sizing', and 'Resources'. Below the navigation is a blue header bar with the title 'Entitlements'. Underneath, there are two tabs: 'My entitlement' (selected) and 'Organization entitlements'. A search bar with placeholder text 'Search' and a date range selector ('Purchase Date' and 'From') are also present. The main content area displays a table of entitlements:

User Number	Entitlement	Purchase Date	Expiration Date	Activation Date	Action Buttons
PRODEMOT11	optia-de-pro	[redacted]	[redacted]	[redacted]	<button>View Activations</button> <button>Re-host</button>
PRODEMOT11	optia-de-pro	[redacted]	[redacted]	[redacted]	<button>View Activations</button> <button>Re-host</button>

At the bottom of the page, there are links for 'Copyright ©2024 Rockwell Automation, Inc.', 'Legal Notice', 'Privacy & Cookies Policy', and the 'Rockwell Automation' logo.

# FactoryTalk Optix Studio Entitlements

- Users of an organization can self-assign any available Studio entitlement from FactoryTalk Optix web page
- Studio Entitlements can be released once they are not needed anymore and used by a different user in the same organization either from FactoryTalk Optix Studio or the web dashboard
- Studio Entitlements can only be rehosted by the user that has is using the license or by an administrator of the organization



# FactoryTalk Optix Runtime Entitlements

- Embedded devices comes with a preinstalled entitlements (size varies depending on the device)
- «Open» devices (like iPC) should be manually activated using the License Manager which is installed with the Runtime Tools
- Runtime licenses can be upgraded by buying a bigger size
  - Runtime licenses upgrade on Embedded devices (like OptixPanels) may be limited to specific sizes

The screenshot shows a web-based interface titled "Entitlements". At the top, there are two tabs: "My entitlement" and "Organization entitlements", with "Organization entitlements" being the active tab. Below the tabs is a search bar with placeholder text "Search entitlements" and a dropdown menu set to "Purchase Date". There are also "From" and "To" date pickers. The main area displays a table with the following columns: "Entitlement Key", "User", "Contract Number", and "Catalog number". The table lists seven rows of data, each containing a different entitlement key, user name, contract number (all starting with "I702M-OPXRTD1"), and catalog number (all ending in "catalog number").

Entitlement Key	User	Contract Number	Catalog number
220HP-K4FEX	[REDACTED]	I702M-OPXRTD1	[REDACTED] catalog number
22T0F-VVWHW	[REDACTED]	I702M-OPXRTD1	[REDACTED] catalog number
266LE-DOKD?	[REDACTED]	I702M-OPXRTD1	[REDACTED] catalog number
3PCER-7VMJH	[REDACTED]	I702M-OPXRTD1	[REDACTED] catalog number
3RYOU-7Vv92	[REDACTED]	I702M-OPXRTD1	[REDACTED] catalog number
2WK37-HL9TF	[REDACTED]	I702M-OPXRTD1	[REDACTED] catalog number
2WY9T-WFPKX	[REDACTED]	I702M-OPXRTD1	[REDACTED] catalog number

# FactoryTalk Optix Entitlements

- A dedicated tutorial is available

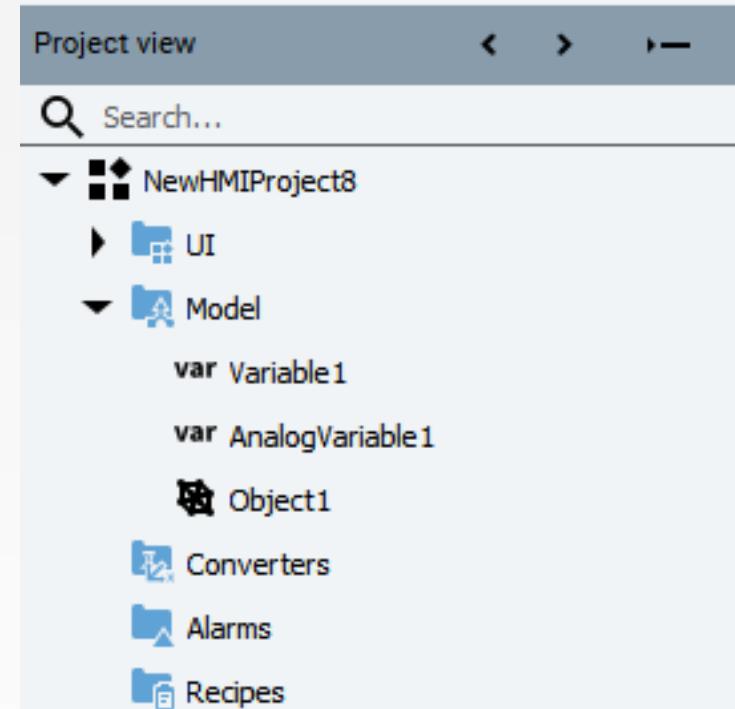


# Variables and tags



# Model's variables

- What are the purposes of these Tags?
  - as "**internal variables**" used only on the HMI project (such as parameters or global variables)
  - as "**bridge**" between the Tag connected to the Comm Driver and the User Interface (such as using same UI for different PLC vendors)
- Under the "Model" node, it's possible to define different types of tags:
  - **Variable**: Standard Tag with desired data type
  - **Analog Variable**: Tag with Engineering Unit (used in combination with Measurement System)
  - **Object**: Structured Tag (a.k.a. User Defined Type)



# Communication drivers

- Used to:
  - communicate with a PLC exchanging Tag values
  - import Tags from the PLC application (if supported by the protocol)
- To configure the communication,
  1. Add a Communication Driver object that represents the communication protocol. Some drivers (like Modbus) have a choice between serial and eth.
  2. Add one (or more) Station object that represents PLC and configure communication parameters

Communication driver	Online tags import	Offline tags import
Modbus driver		
MELSEC FX3U Driver		✓
S7TCP driver	✓	✓
OMRON EtherNet/IP driver		✓
MELSEC Q driver		✓
S7 TIA PROFINET driver	✓	✓
OMRON Fins Driver		✓
CODESYS Driver	✓	✓
TwinCAT driver	✓	✓
Serial port		
RA EtherNet/IP Driver	✓	✓
Micro Controller Driver	✓	

# Communication drivers

- Some drivers also supports runtime import of PLC Tags
  - A custom logic must be added to the project (NetLogic examples are available on GitHub)
  - Only supported for:
    - RA Ethernet/IP
    - Twincat
    - Siemens S7 TCP

```
[FTOptix.NetLogic.ExportMethod]
public static void ImportFromRAEthernetIP()
{
    Log.Info("RuntimeTagsImport.ImportFromRAEthernetIP", "Starting runtime tags import");

    var station = Project.Current.Get<FTOptix.RAEtherNetIP.Station>("CommDrivers/RAEtherNet_IPDriver1/RAEtherNet_IPStation1");

    // Call Browse method to retrieve data from PLC
    station.Browse(out var plcItems, out var prototypes);

    Log.Info("RuntimeTagsImport.ImportFromRAEthernetIP", "Fetched " + plcItems.Length + " tags structures from the PLC");

    // Filter tag to import by tag name
    var tagInstances = FilterTagsToImport(new List<string>() { "Controller Tags/Tank1/Level", "Controller Tags/Tank2/Level" }, plcItems);

    Log.Info("RuntimeTagsImport.ImportFromRAEthernetIP", "Importing " + tagInstances.Length + " tags");

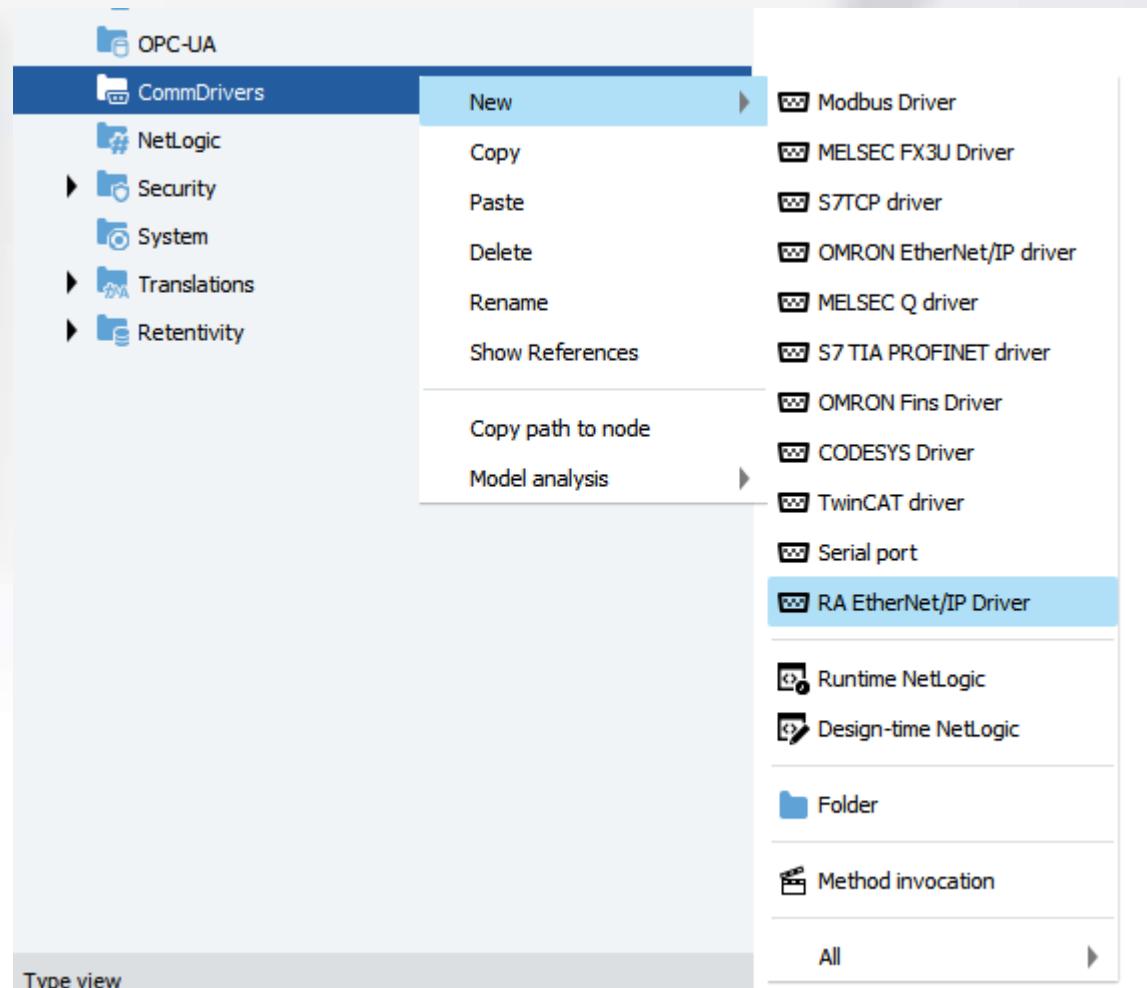
    if (tagInstances.Length == 0)
    {
        Log.Warning("RuntimeTagsImport.ImportFromRAEthernetIP", "No tags to import. Returning.");
        return;
    }

    //Filter tag to import by prototypeID
    // var tagInstances = GetTagsOfType("MachineParameter", plcItems);

    station.Import(tagInstances, prototypes);
}
```

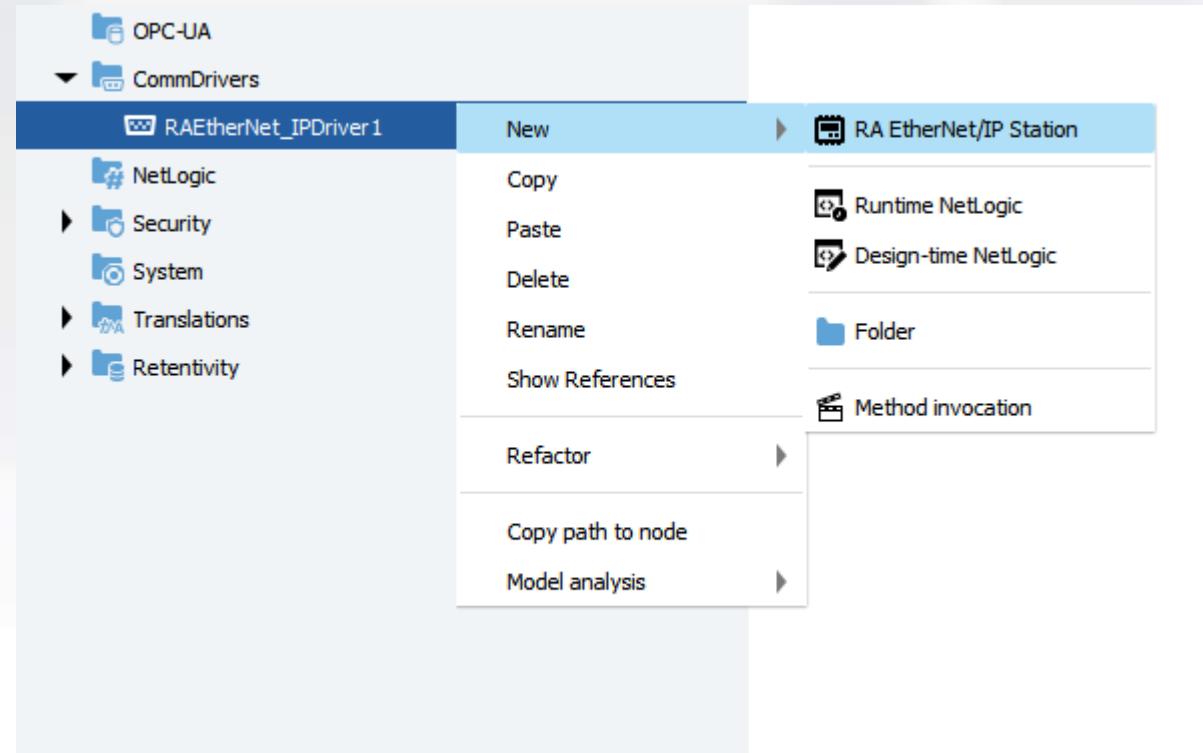
# Communication drivers

- Used to:
  - communicate with a PLC exchanging Tag values
  - import Tags from the PLC application (if supported by the protocol)
- To configure the communication,
  1. Add a Communication Driver object that represents the communication protocol. Some drivers (like Modbus) have a choice between serial and eth.
  2. Add one (or more) Station object that represents PLC and configure communication parameters



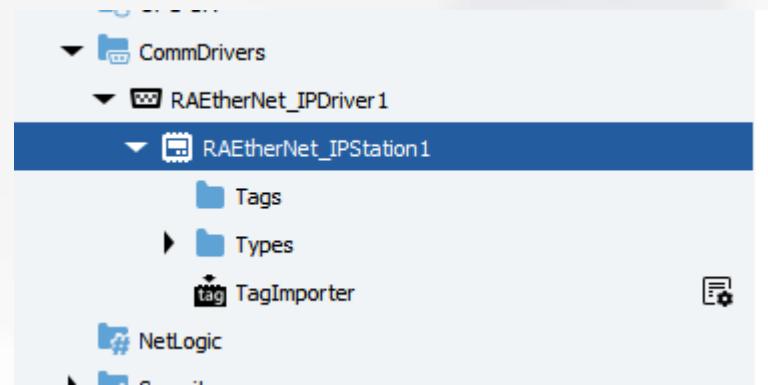
# Communication drivers

- Used to:
  - communicate with a PLC exchanging Tag values
  - import Tags from the PLC application (if supported by the protocol)
- To configure the communication,
  1. Add a Communication Driver object that represents the communication protocol. Some drivers (like Modbus) have a choice between serial and eth.
  2. Add one (or more) Station object that represents PLC and configure communication parameters



# Communication drivers

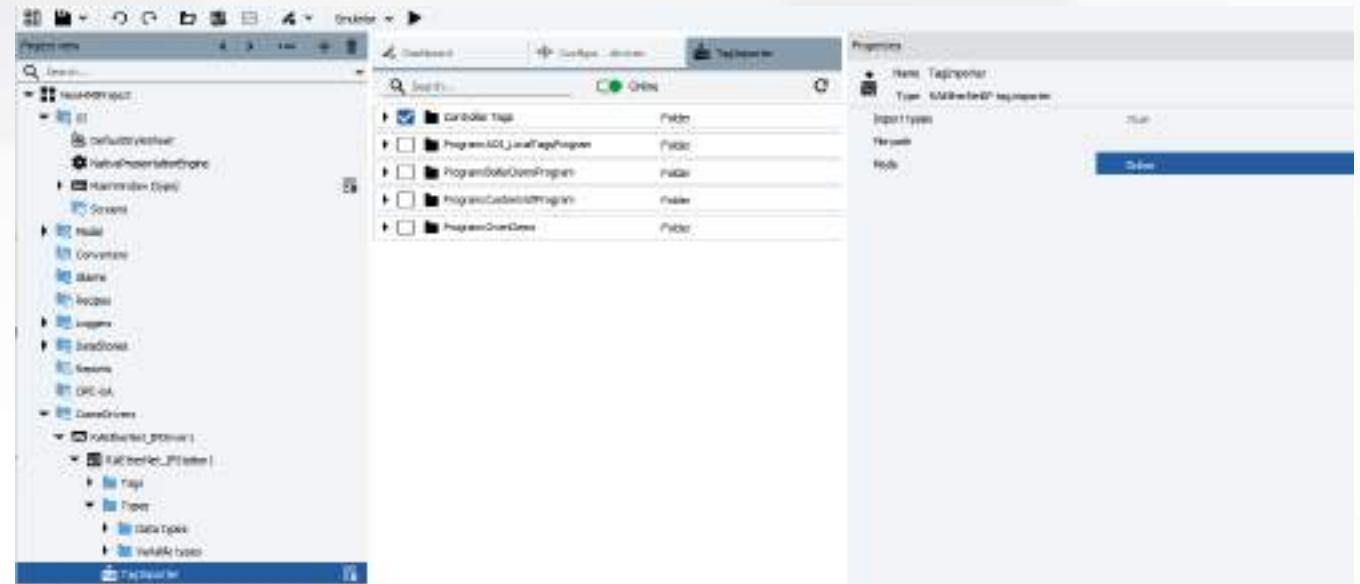
- Used to:
  - communicate with a PLC exchanging Tag values
  - import Tags from the PLC application (if supported by the protocol)
- To configure the communication,
  1. Add a Communication Driver object that represents the communication protocol. Some drivers (like Modbus) have a choice between serial and eth.
  2. Add one (or more) Station object that represents PLC and configure communication parameters



# Add tags exchanged with plc: tag importer

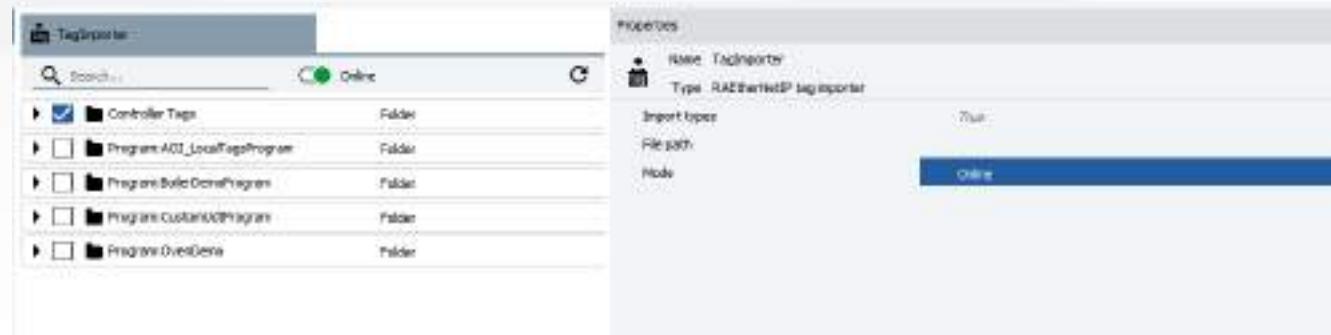
- Tag Importer
  - *Offline*: from an export file generated by PLC programming software
  - *Online*: reading tags directly from an online PLC (if supported by the comm driver)

- Remind!
  - the number of tags exchanged with PLC are not counted by the license manager



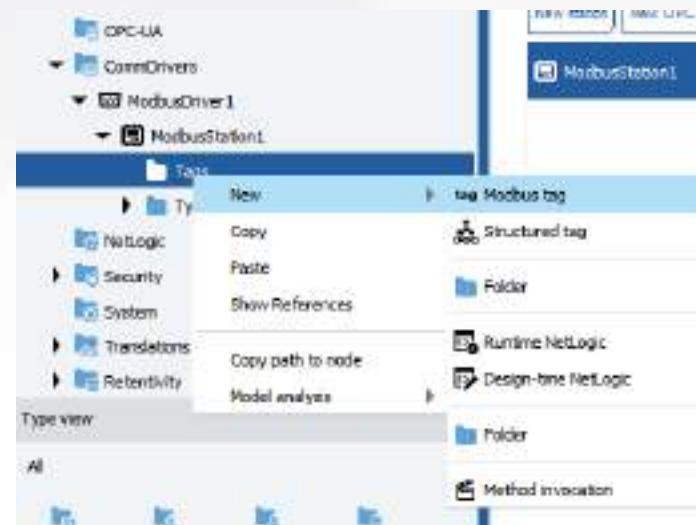
# Add tags exchanged with plc: tag importer

- Tag Importer
  - *Offline*: from an export file generated by PLC programming software
  - *Online*: reading tags directly from an online PLC (if supported by the comm driver)
- Remind!
  - the number of tags exchanged with PLC are not counted by the license manager



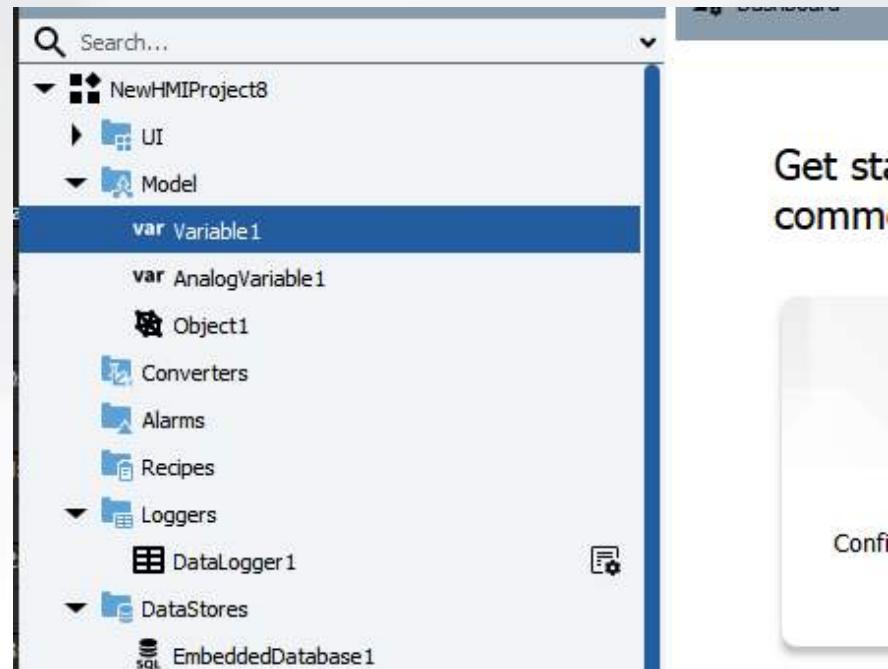
# Add tags exchanged with plc: manually

- for Communication Drivers not using symbolic names (i.e. Modbus TCP)
- tags can be created “manually”
  - Right-click on the Tags node and select the "New Variable" command
  - on the Properties panel, configure the tag



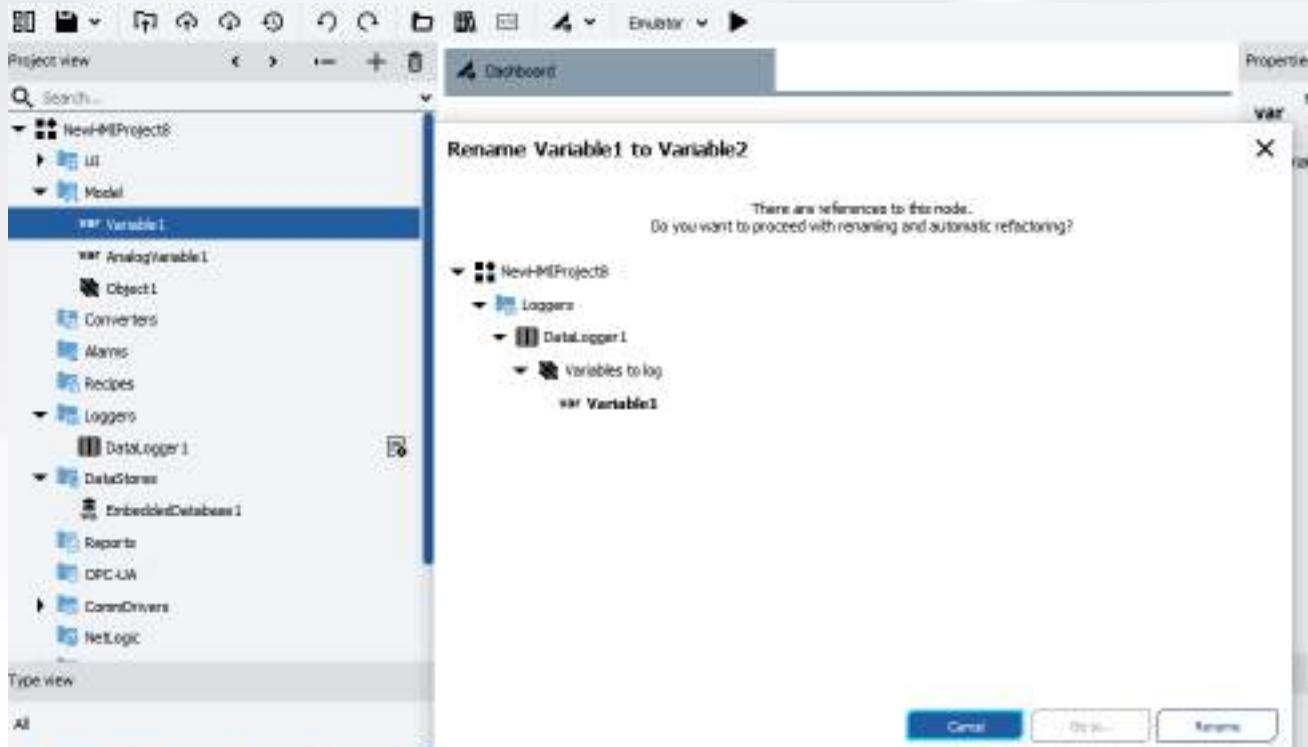
# Tag rename

- Tags can be renamed
  - Applies to Model's Tags
  - Applies to Tags exchanged with PLC created manually and also imported!
  - If the Tag is referenced by a resource that requires refactoring (example: a datalogger), a popup will appear automatically
  - If it has been imported, the Tag will be marked as "Renamed"



# Tag rename

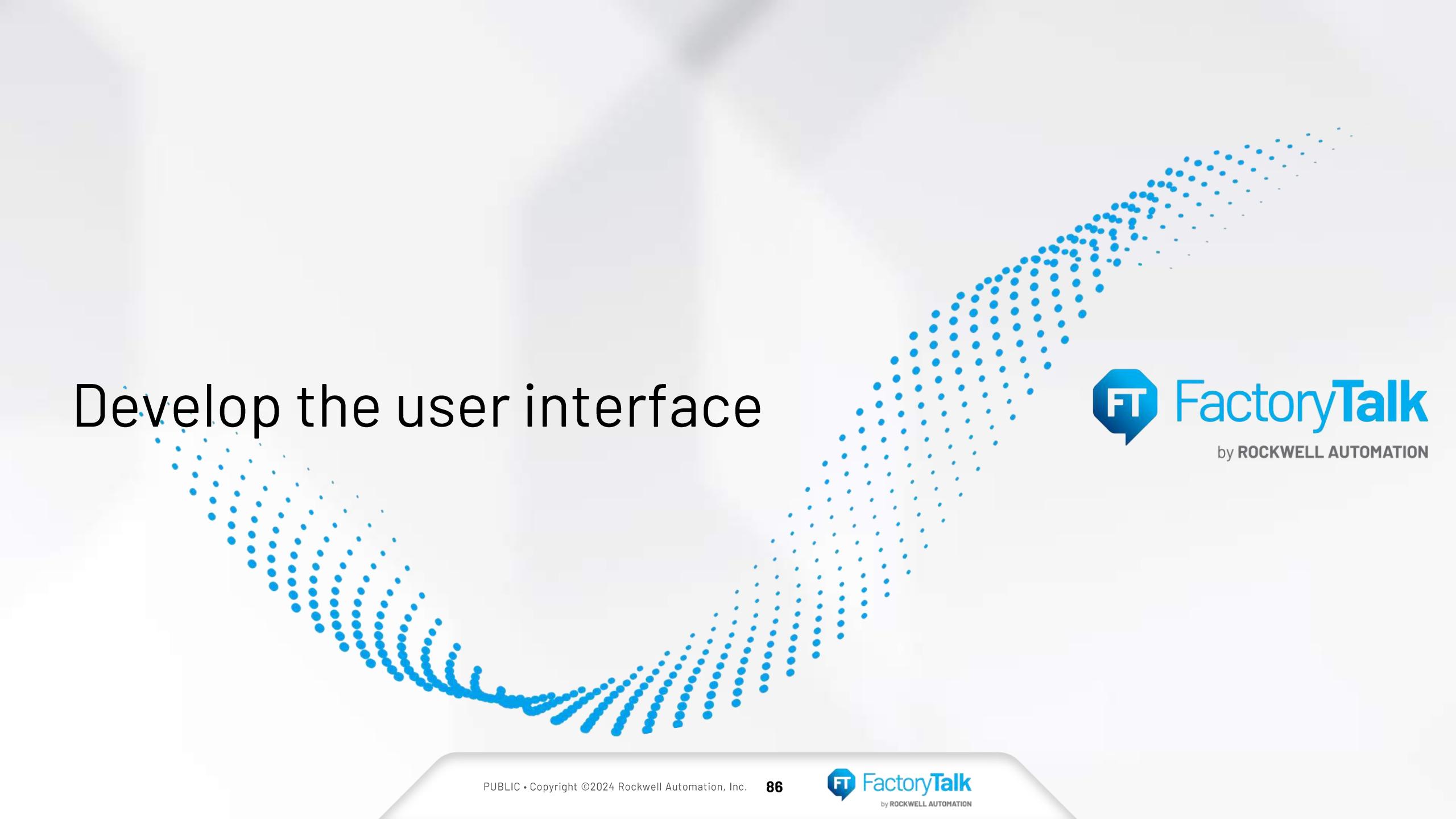
- Tags can be renamed
  - Applies to Model's Tags
  - Applies to Tags exchanged with PLC created manually and also imported!
  - If the Tag is referenced by a resource that requires refactoring (example: a datalogger), a popup will appear automatically
  - If it has been imported, the Tag will be marked as "Renamed"



# Hands-on session

- Create some Model tags
- Add the RA Ethernet/IP communication driver
- Import tags from FactoryTalk Logix Echo



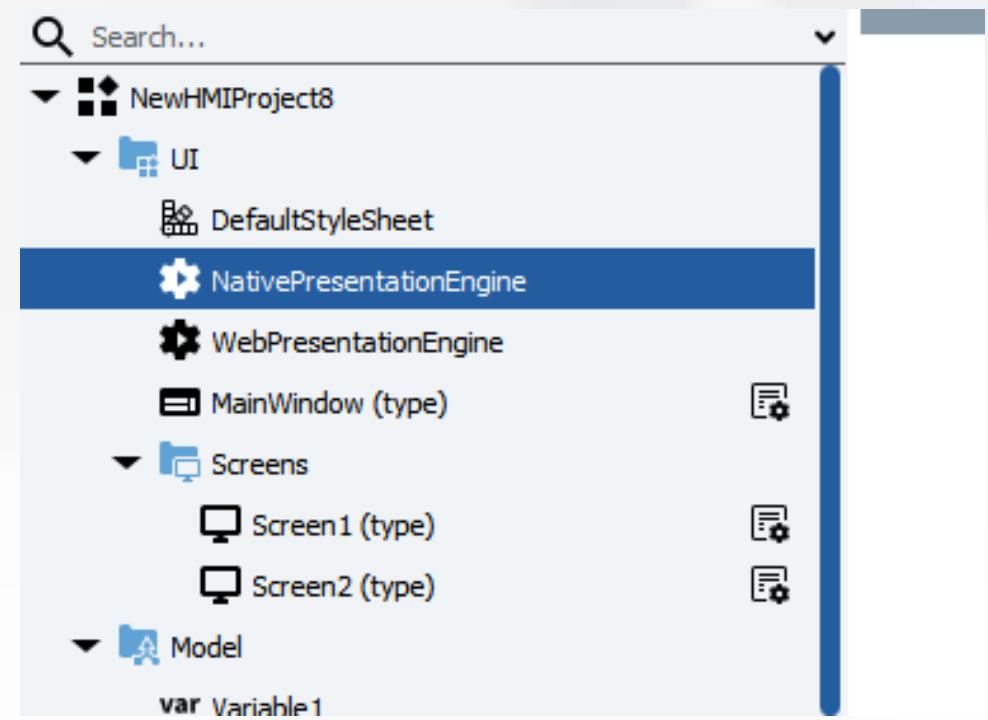
A large, abstract graphic composed of numerous small blue dots arranged in a wave-like, undulating pattern across the entire slide.

# Develop the user interface



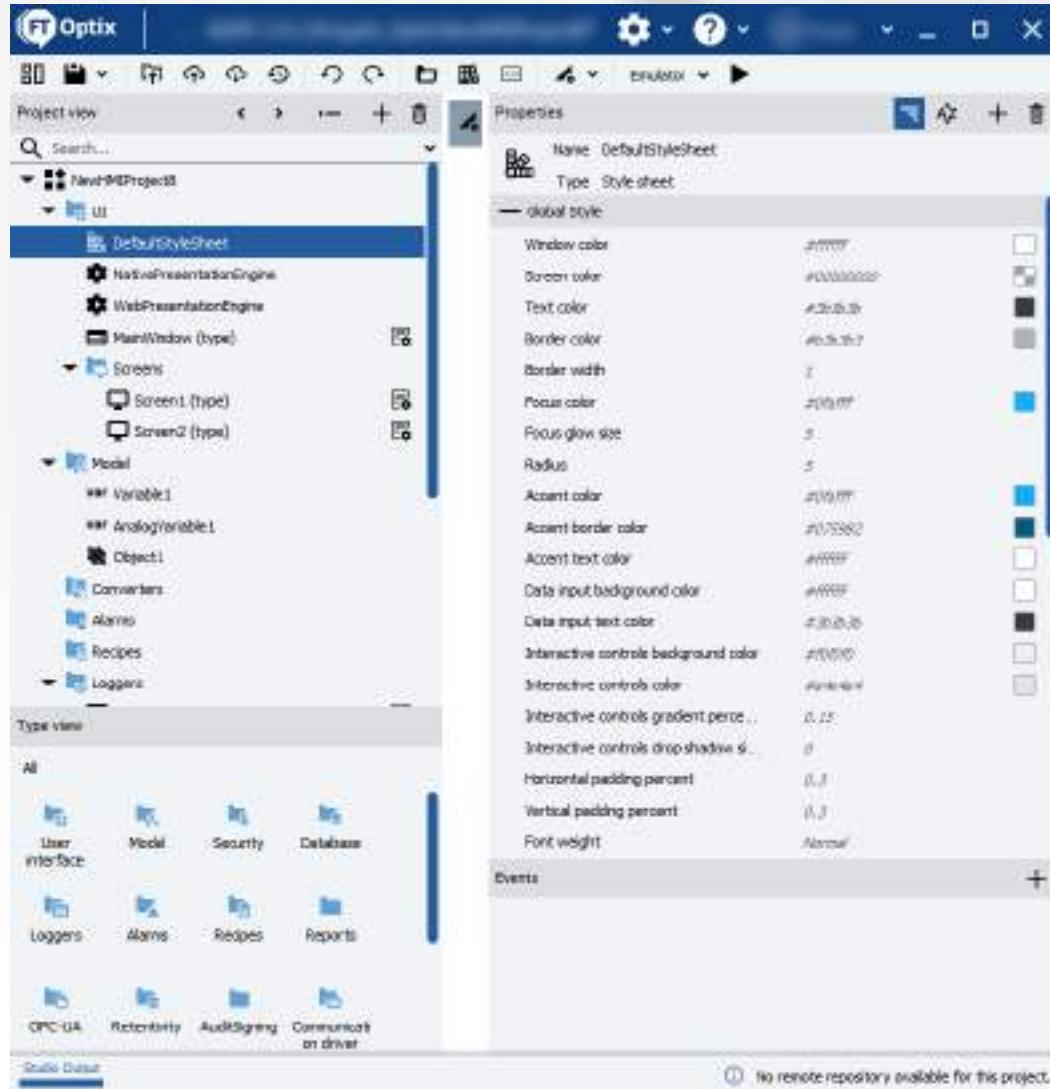
# User interface (UI)

- Default Style Sheet
  - makes it possible to globally set some style properties of all graphical objects in the project
- Presentation Engine
  - **Native** Presentation Engine it's the object that create the user interface of the Application
  - **Web** Presentation Engine, it's the object that create the user interface for Web access
- Main Window
  - This object is the root container of graphical objects



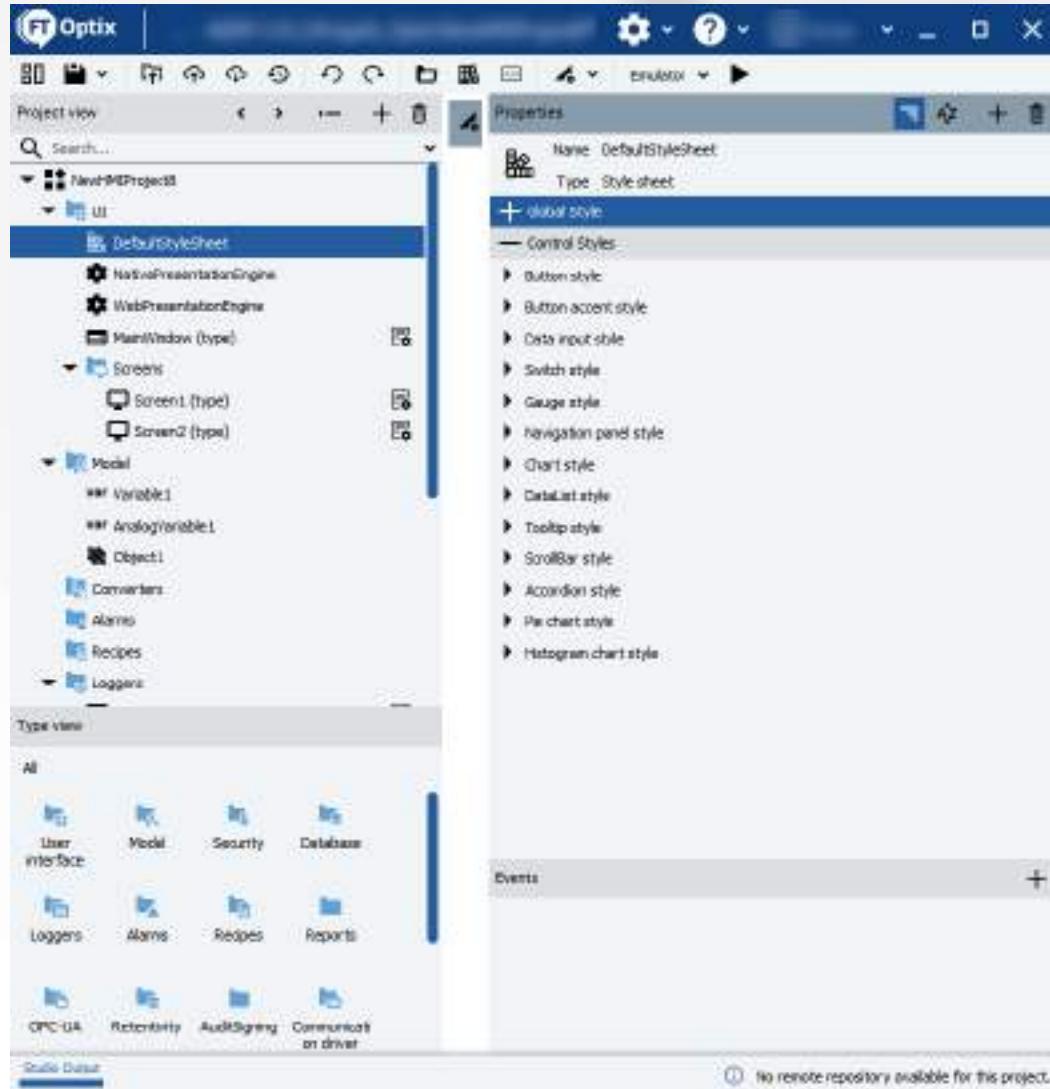
# Default style sheet

- Properties
  - Global Style: style properties in common with all objects (like radius)
  - Control Styles: style properties for a specific type of object (Button, Switch...)
- Multiple style sheets
  - into the same project can be defined multiple style sheets for example, to allow to switch between a light/dark interface at Runtime
- Template Libraries
  - contain some pre-defined and “ready to use” Style sheets



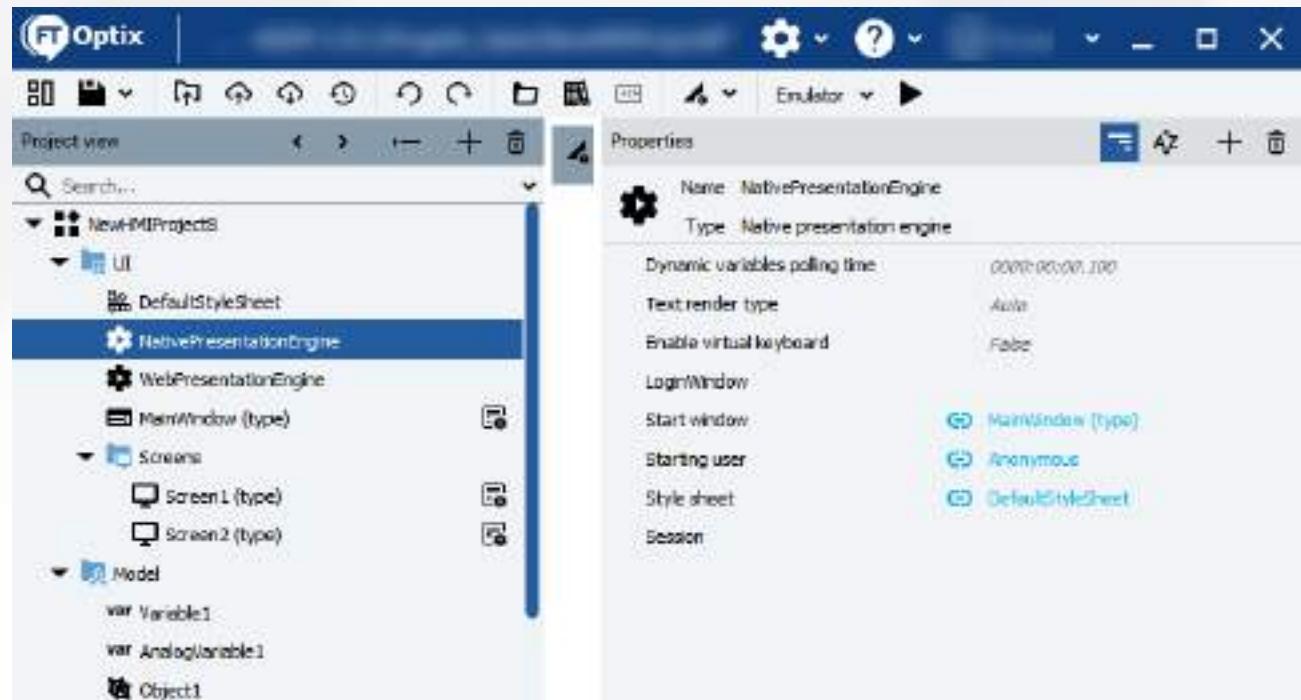
# Default style sheet

- Properties
  - Global Style: style properties in common with all objects (like radius)
  - Control Styles: style properties for a specific type of object (Button, Switch...)
- Multiple style sheets
  - into the same project can be defined multiple style sheets for example, to allow to switch between a light/dark interface at Runtime
- Template Libraries
  - contain some pre-defined and “ready to use” Style sheets



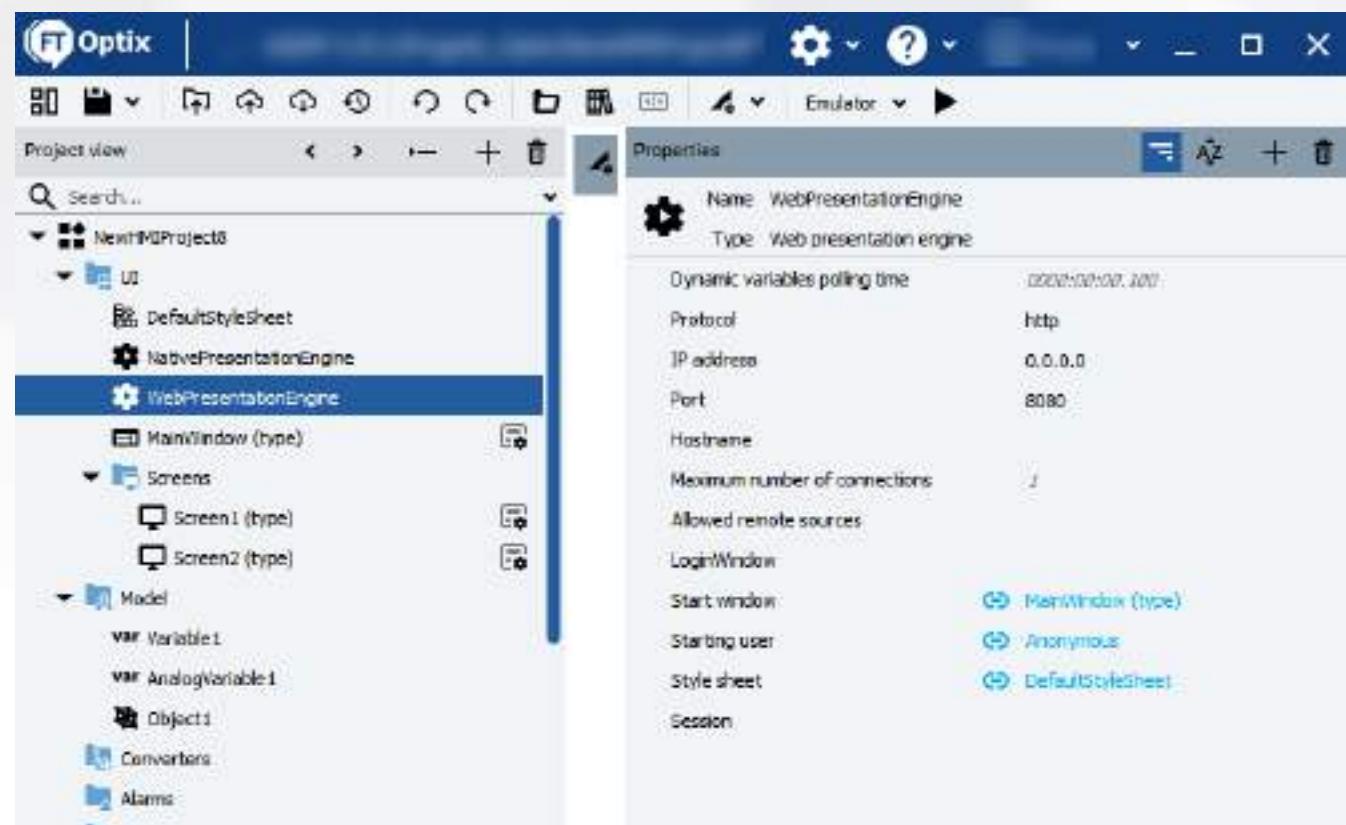
# Native presentation engine

- Properties
  - Dynamic Variables polling time: used to refresh Tags used in Presentation Engine.  
NOTE: this will also set the polling time used by comm drivers (for active tags)
- Start Window
- Starting User
- Style Sheet
- Enable Virtual Keyboard



# Web presentation engine

- Allow publishing the UI through the integrated Web Server
- Properties:
  - Start Window, Starting User, Style Sheet...
  - **Protocol:** http/https
  - **IP Address:** this will be the IP address used by Web Server  
0.0.0.0 = listen on all addresses
  - **Port:** define the listening port
  - Hostname
  - Maximum Number of connections
  - Allowed remote sources: external sources that can be embedded (websites)

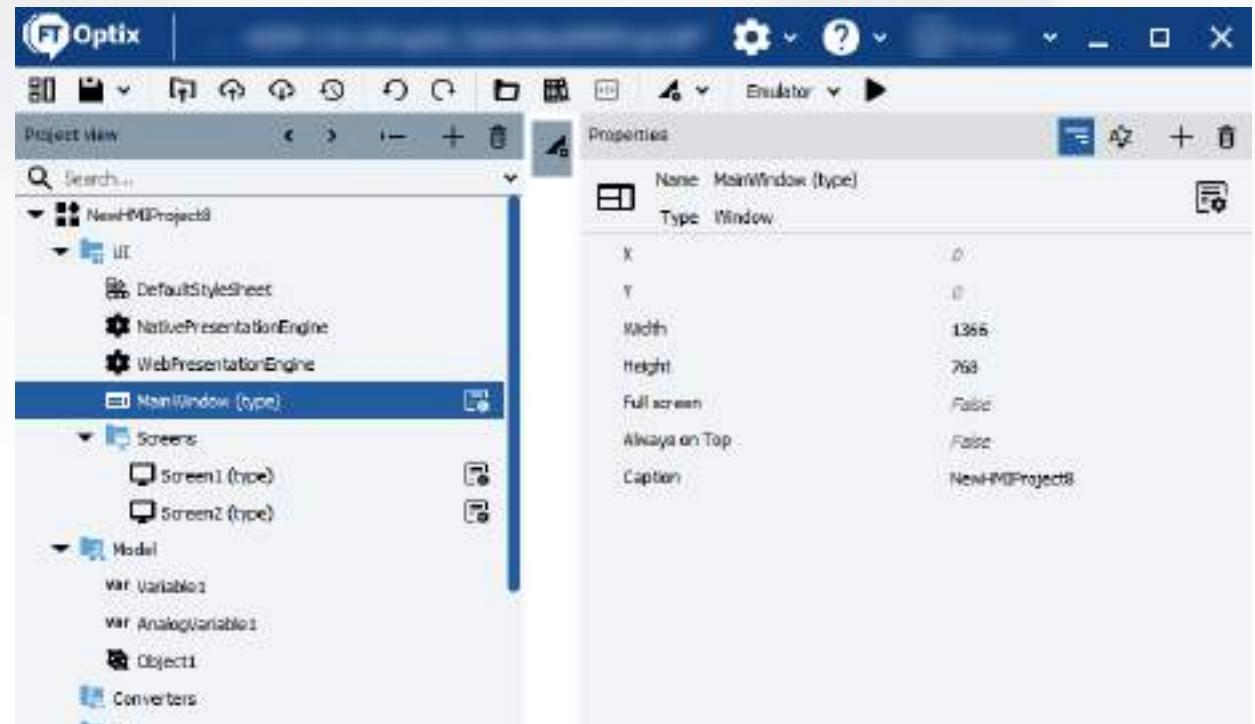


# Main window

- Define the window where all graphical objects are showed
  - width/height: defines (manually) the size of the entire User Interface
  - Full Screen: set the size of UI equal to the current screen resolution hiding the title bar
  - Always on Top: force the window to be always on top of other applications
  - Caption: title of the window

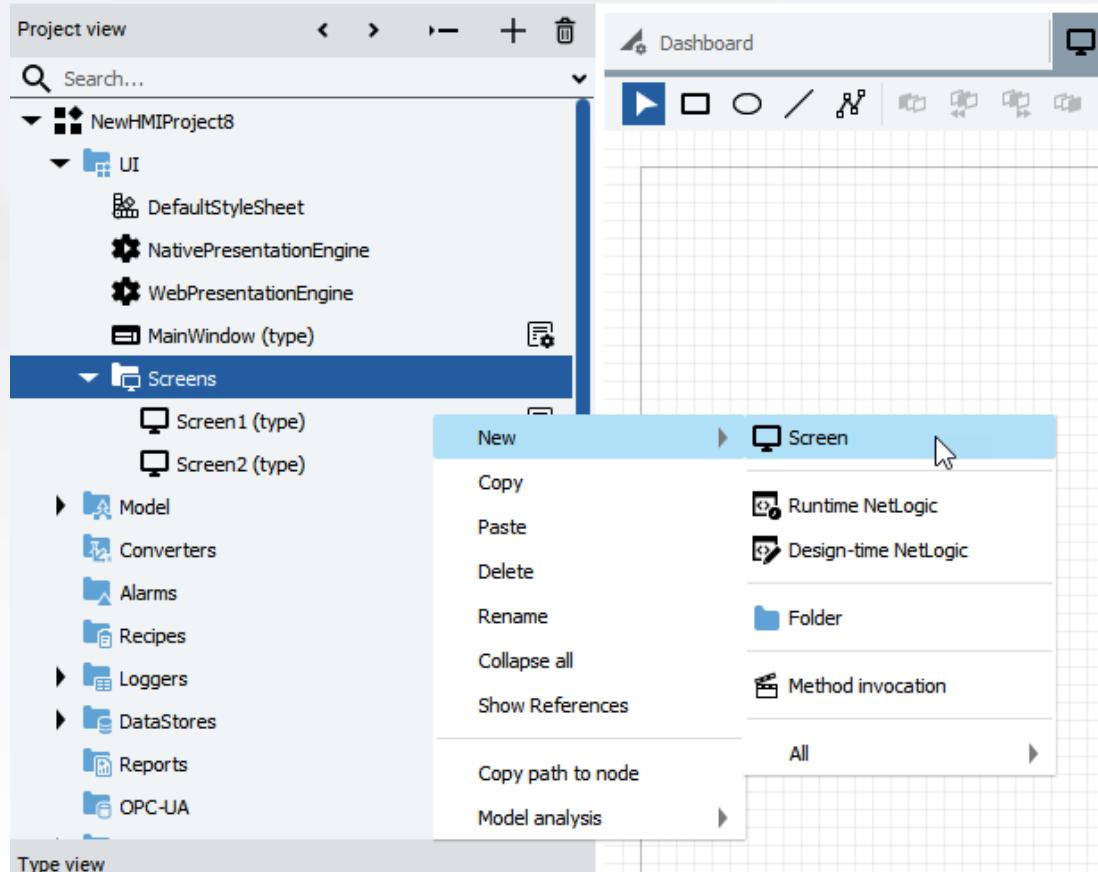
- NOTE:

- Application is a single-window application
- Switching between a page to another will not close/open a window, but will only change its content (containers)



# Screens

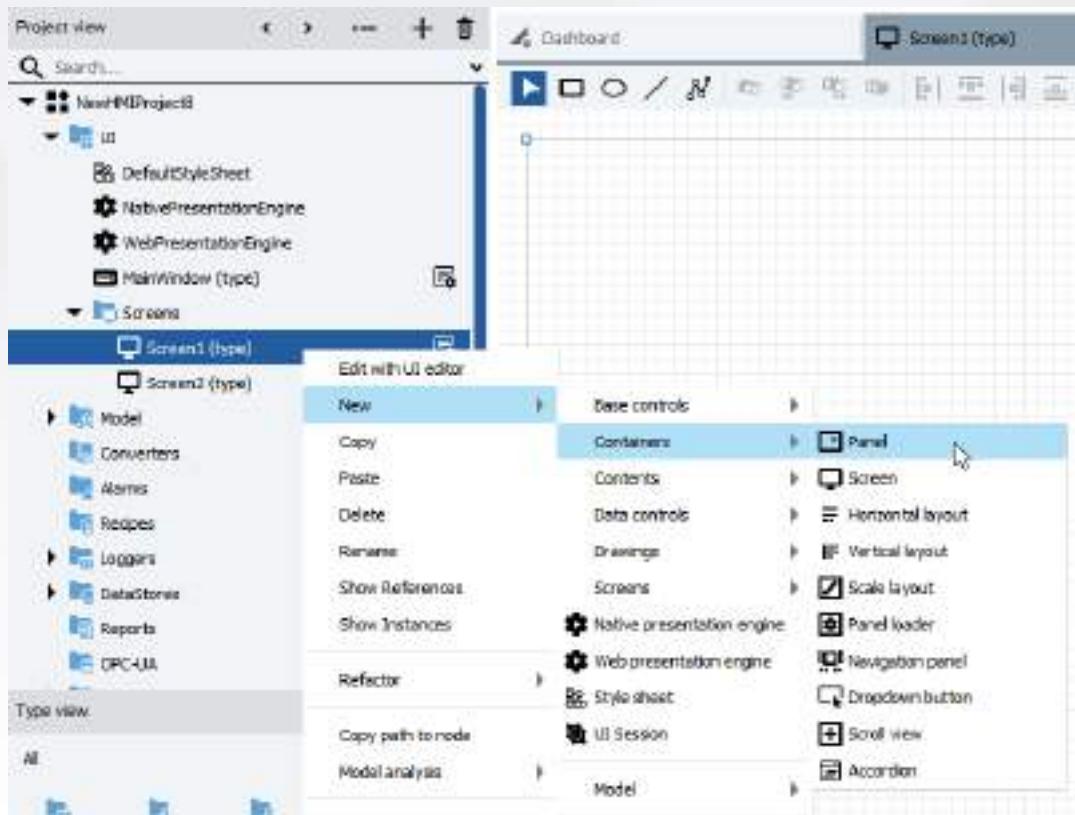
- Screens are the pages that are loaded to the running applications



- Screen sizes are stretched to the parent container (PanelLoader or NavigationPanel)
- Background can be customized with StyleSheet

# Panels

- Panel is a Container that aggregates graphical objects (i.e. widgets)



- create a new Panel:
  - Right-click on the Page
  - select New > Containers > Panel
- Panels are transparent only
  - Can be used to group elements to trigger specific behaviors on all child components (like visibility, enablement, etc)

# Navigation

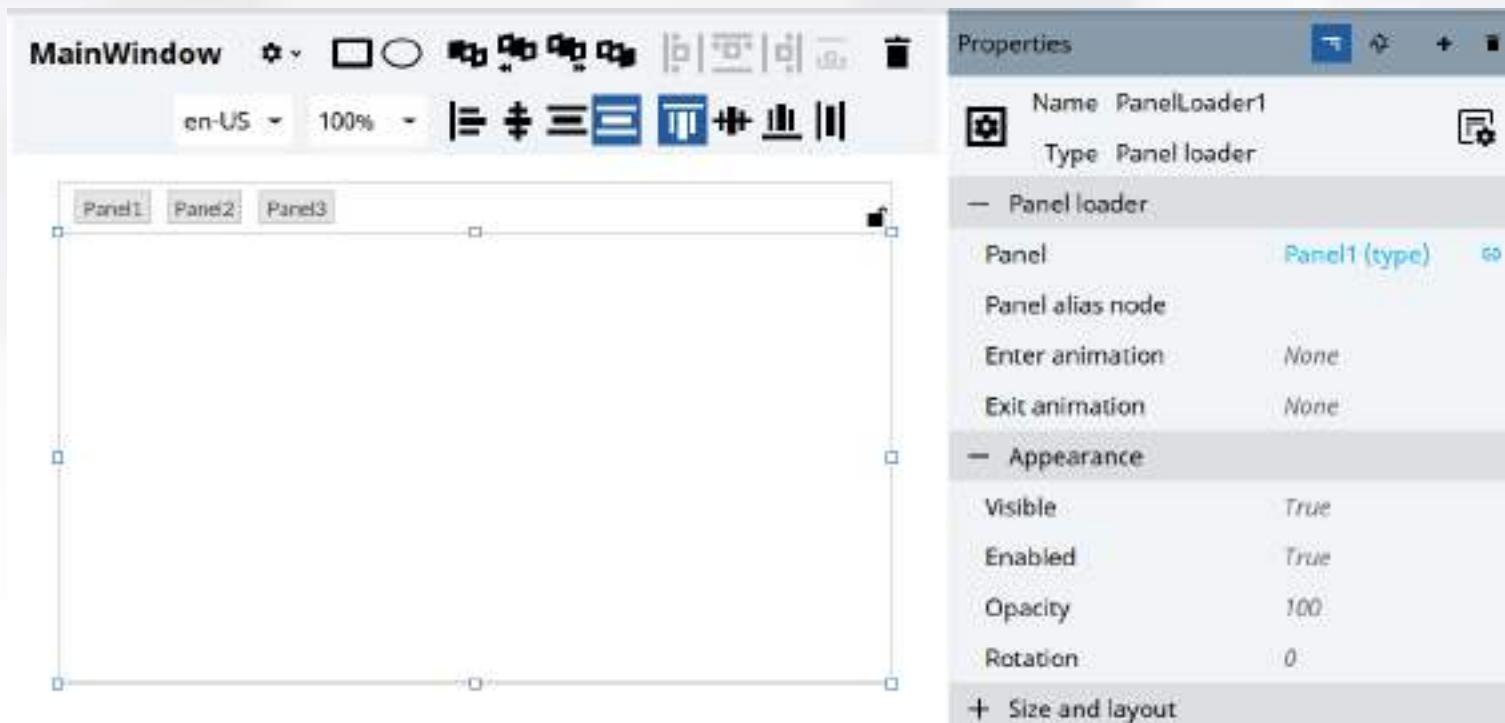
- Navigation can take place using two different ways

## 1. Panel Loader

- Container to display panels or screens
- "Change panel" method allow to load a different panel

## 2. Navigation Panel

- Container that automatically organizes Panels into navigable tabs.
- Just Drag&Drop Screens into the object
- Tabs can be hidden or disabled



# Navigation

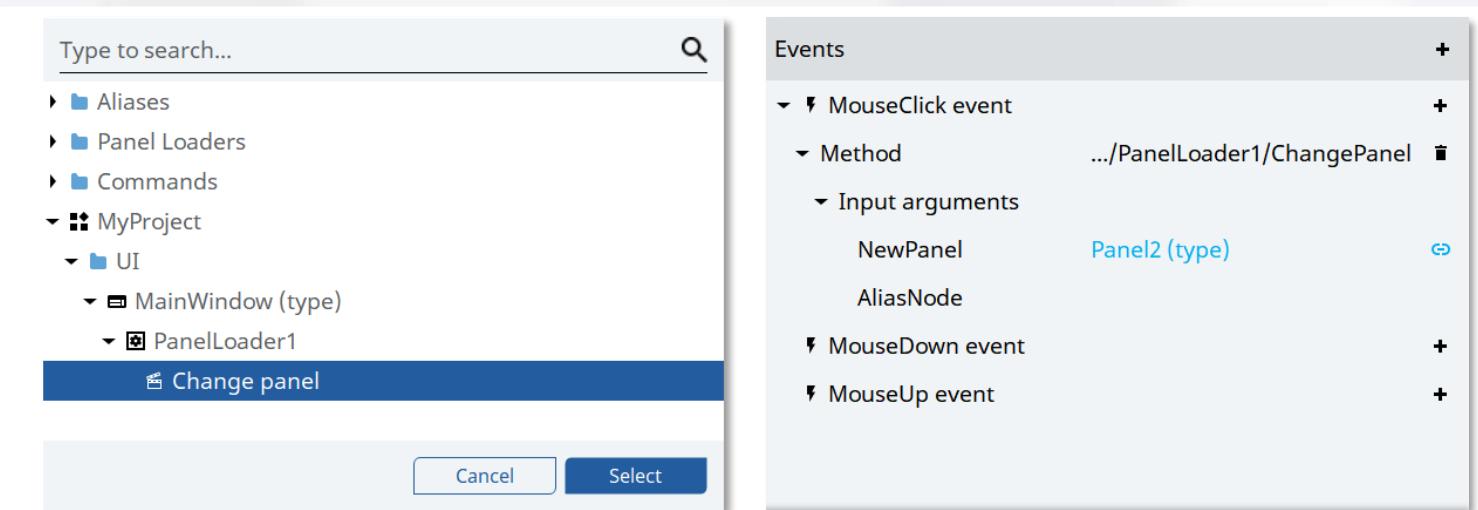
- Navigation can take place using two different ways

## 1. Panel Loader

- Container to display panels or screens
- "Change panel" method allow to load a different panel

## 2. Navigation Panel

- Container that automatically organizes Panels into navigable tabs.
- Just Drag&Drop Screens into the object
- Tabs can be hidden or disabled



# Navigation

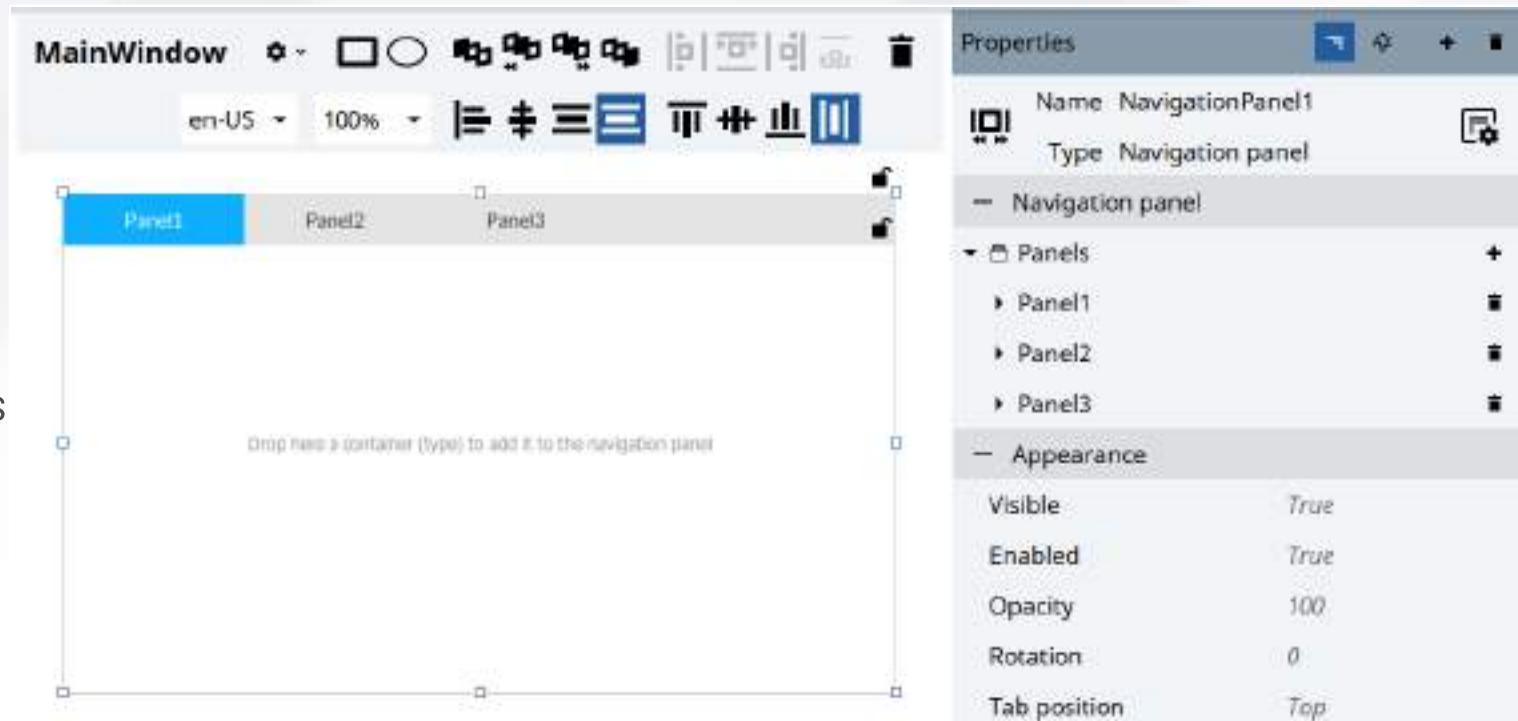
- Navigation can take place using two different ways

## 1. Panel Loader

- Container to display panels or screens
- "Change panel" method allow to load a different panel

## 2. Navigation Panel

- Container that automatically organizes Panels into navigable tabs.
- Just Drag&Drop Screens into the object
- Tabs can be hidden or disabled



# Responsive design

- This can be achieved by using the **relative positioning** of objects or **scaling** them to be bigger or smaller

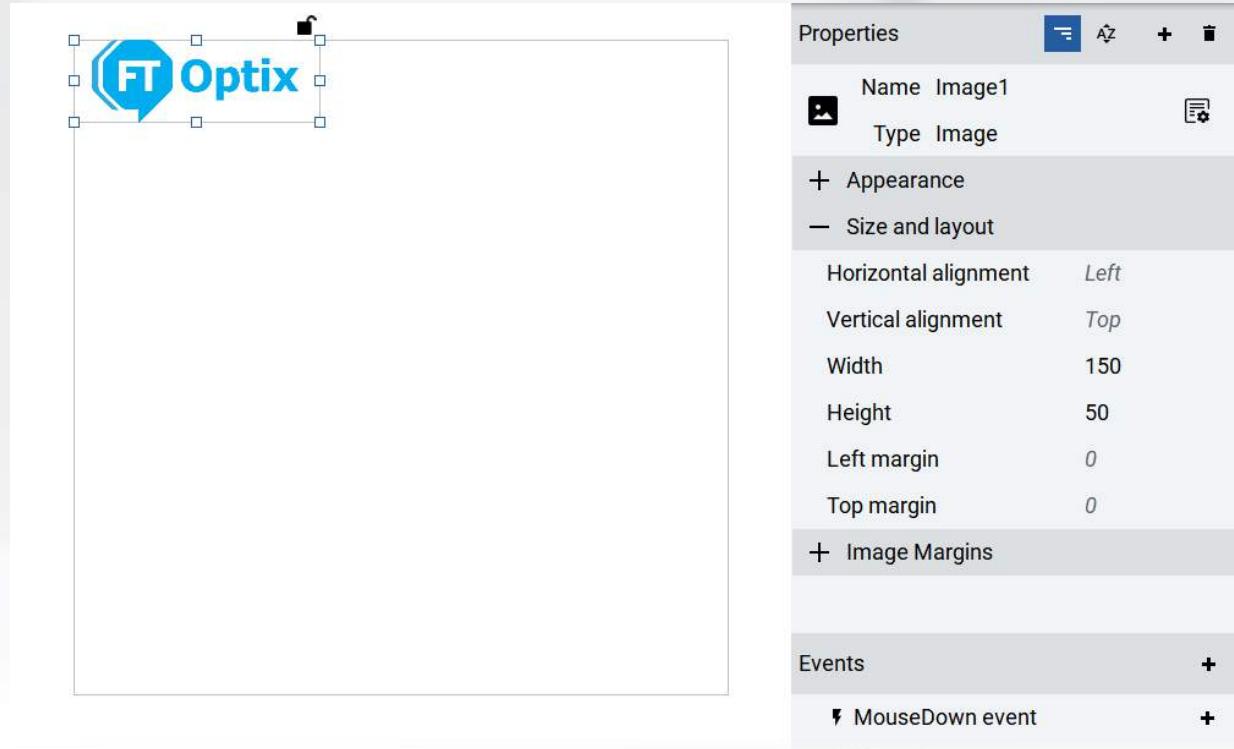
## 1. Alignment \*: manage the reference

- Left/Right and Top/Bottom
- Center
- Stretch

## 2. Margins: manage the distance from the reference

- Horizontal Layout and Vertical Layout can be used for structured layouts

\* Remember: to set Alignment on objects and also on all its containers: Panel, Navigation Panel...



# Responsive design

- This can be achieved by using the **relative positioning** of objects or **scaling** them to be bigger or smaller

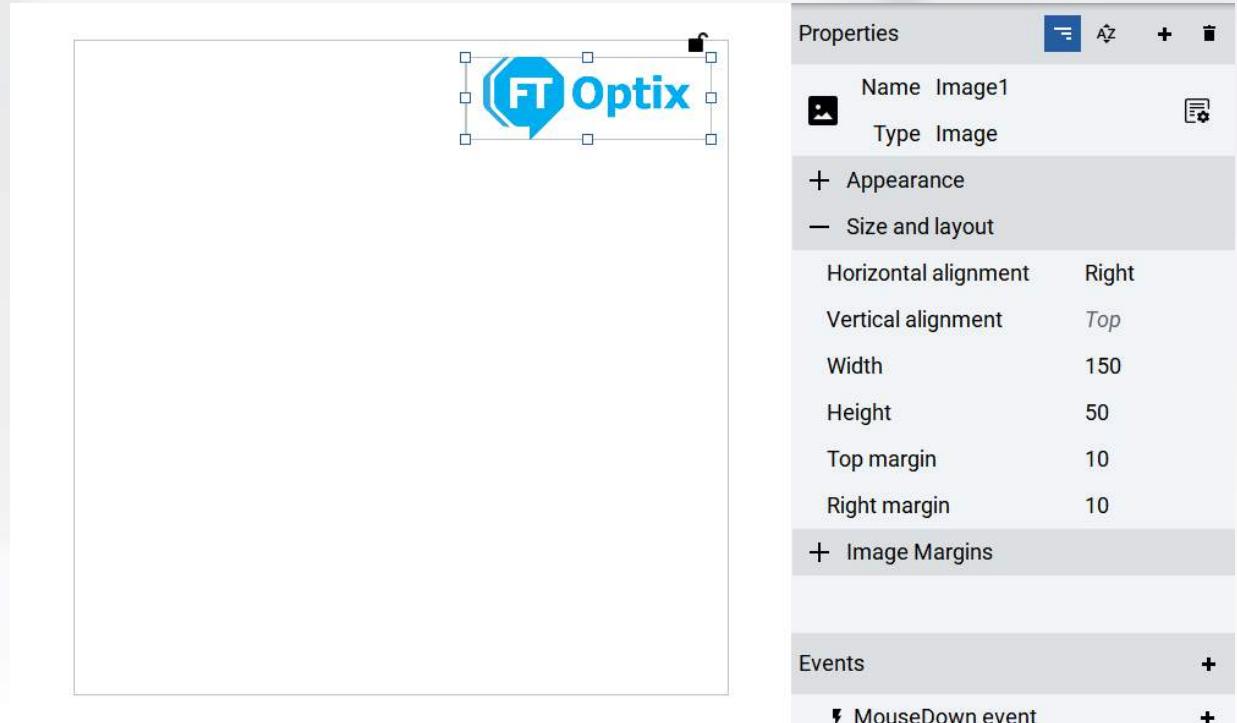
## 1. Alignment \*: manage the reference

- Left/Right and Top/Bottom
- Center
- Stretch

## 2. Margins: manage the distance from the reference

- Horizontal Layout and Vertical Layout can be used for structured layouts

\* Remember: to set Alignment on objects and also on all its containers: Panel, Navigation Panel...



# Responsive design

- This can be achieved by using the **relative positioning** of objects or **scaling** them to be bigger or smaller

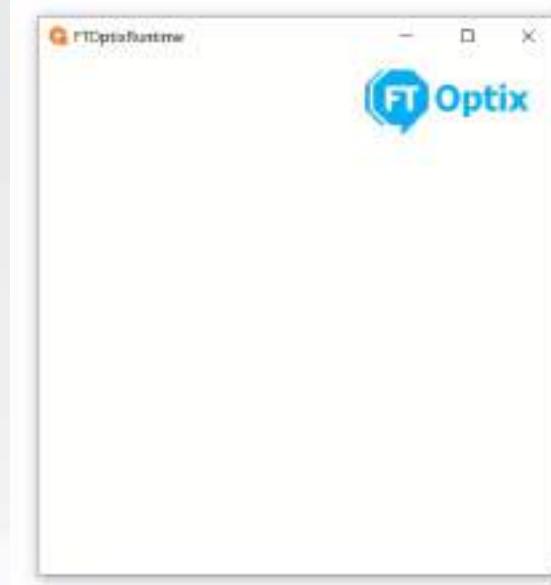
## 1. Alignment \*: manage the reference

- Left/Right and Top/Bottom
- Center
- Stretch

## 2. Margins: manage the distance from the reference

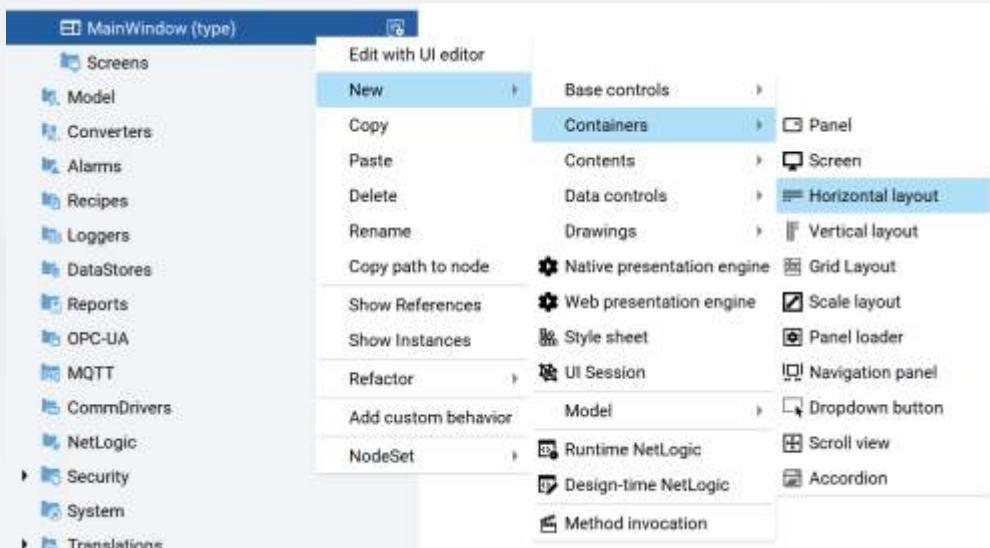
- Horizontal Layout and Vertical Layout can be used for structured layouts

\* Remember: to set Alignment on objects and also on all its containers: Panel, Navigation Panel...



# Horizontal Layout

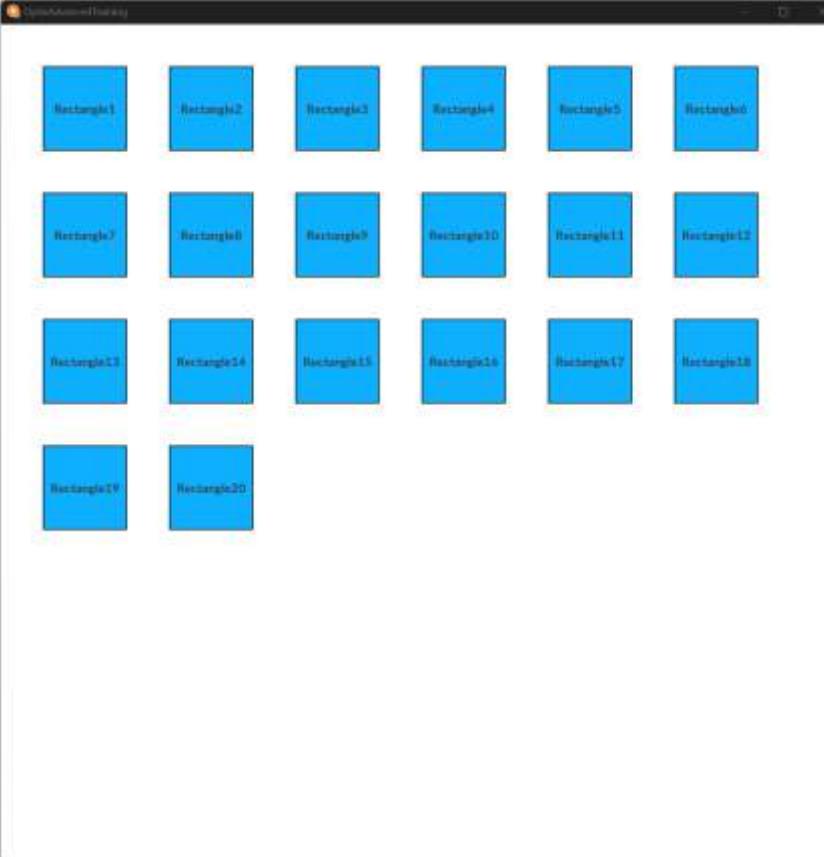
- Automatically lay out content horizontally inside a container



Content alignment	Left aligned
Wrap	False
Horizontal gap	0
Vertical gap	0

- Content alignment: Where to place content inside the container (Left aligned, Center aligned, Right aligned)
- Wrap: Start a new row when content exceeds the horizontal bounds of the container
- Horizontal gap: How much horizontal space between content
- Vertical gap: How much vertical space between content

# Horizontal Layout

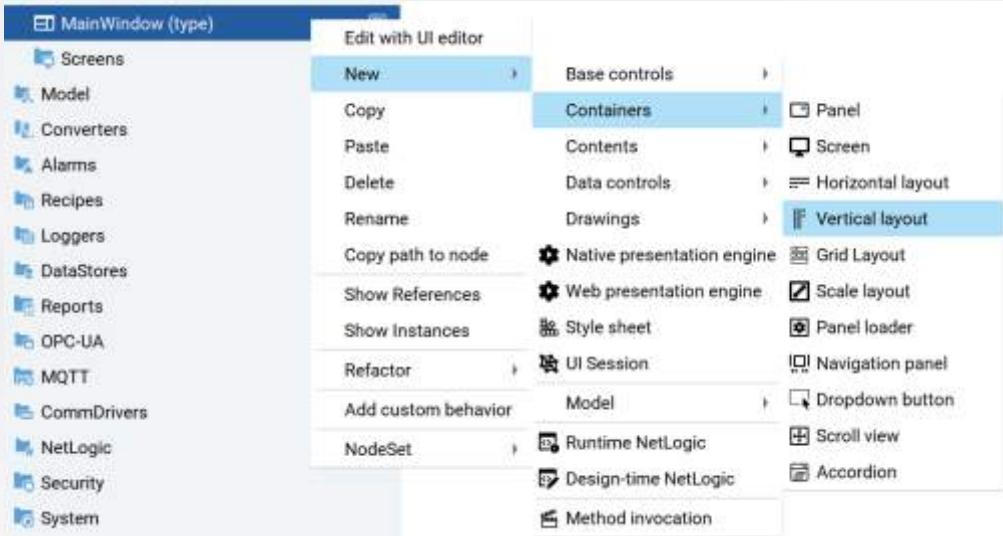


The screenshot shows a Windows application window titled "OpenStackHorizontal4.v1". Inside the window, there is a horizontal layout containing 20 blue rectangular controls, each labeled with a name such as Rectangle1, Rectangle2, etc. The layout is organized into four rows: the first row has 6 rectangles, the second row has 6 rectangles, the third row has 6 rectangles, and the fourth row has 2 rectangles. On the left side of the window, there is a hierarchical tree view of the application's structure. The root node is "MainWindow (type)". Under "MainWindow", there is a "HorizontalLayout1" node, which contains "Rectangle1" through "Rectangle20". Each rectangle in the tree view has a small icon next to it, likely indicating its properties or state.

— Horizontal Layout	
Content alignment	Left aligned
Wrap	True
Horizontal gap	50
Vertical gap	50
— Appearance	
Visible	True
Enabled	True
Opacity	100
Rotation	0
Move target	None
— Size and layout	
Horizontal alignment	Stretch
Vertical alignment	Stretch
Left margin	50
Top margin	50
Right margin	50
Bottom margin	50

# Vertical Layout

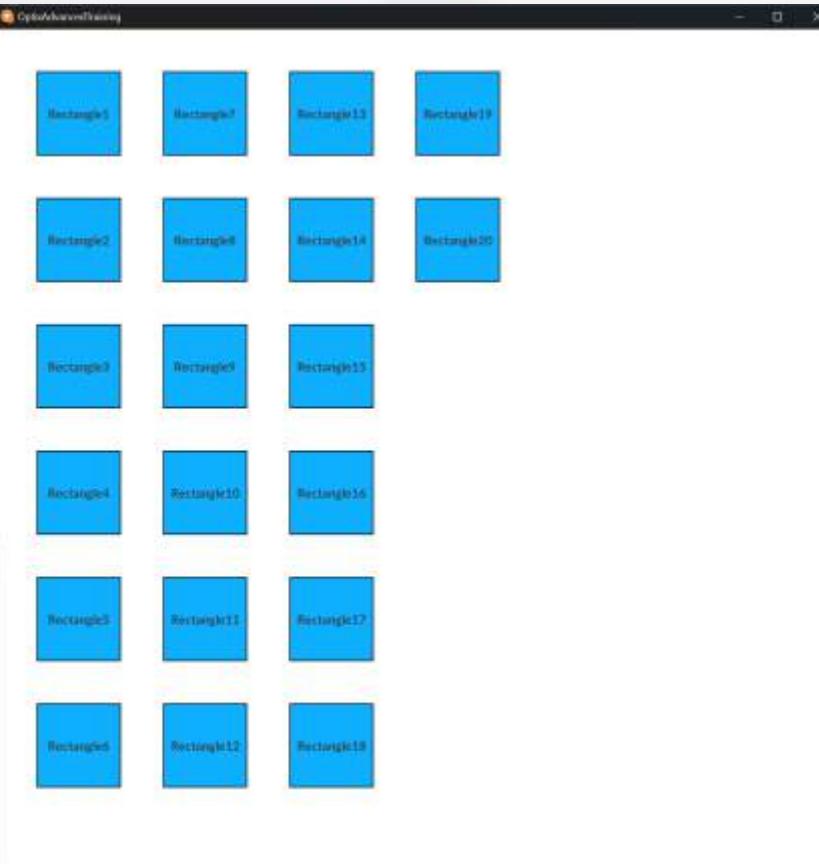
- Automatically lay out content vertically inside a container



Content alignment	Top aligned
Wrap	False
Horizontal gap	0
Vertical gap	0

- Content alignment: Where to place content inside the container (Left aligned, Center aligned, Right aligned)
- Wrap: Start a new column when content exceeds the vertical bounds of the container
- Horizontal gap: How much horizontal space between content
- Vertical gap: How much vertical space between content

# Vertical Layout



The screenshot shows a Windows application window titled "OptionsAndTraining". Inside the window, there is a 5x4 grid of 20 blue rectangles, labeled Rectangle1 through Rectangle20. To the left of the window, a file explorer-style tree view displays the project structure:

- MainWindow (type)
  - VerticalLayout1
    - Rectangle1
    - Rectangle2
    - Rectangle3
    - Rectangle4
    - Rectangle5
    - Rectangle6
    - Rectangle7
    - Rectangle8
    - Rectangle9
    - Rectangle10
    - Rectangle11
    - Rectangle12
    - Rectangle13
    - Rectangle14
    - Rectangle15
    - Rectangle16
    - Rectangle17
    - Rectangle18
    - Rectangle19
    - Rectangle20

## — Horizontal Layout

Content alignment	Top aligned
Wrap	True
Horizontal gap	50
Vertical gap	50

## — Appearance

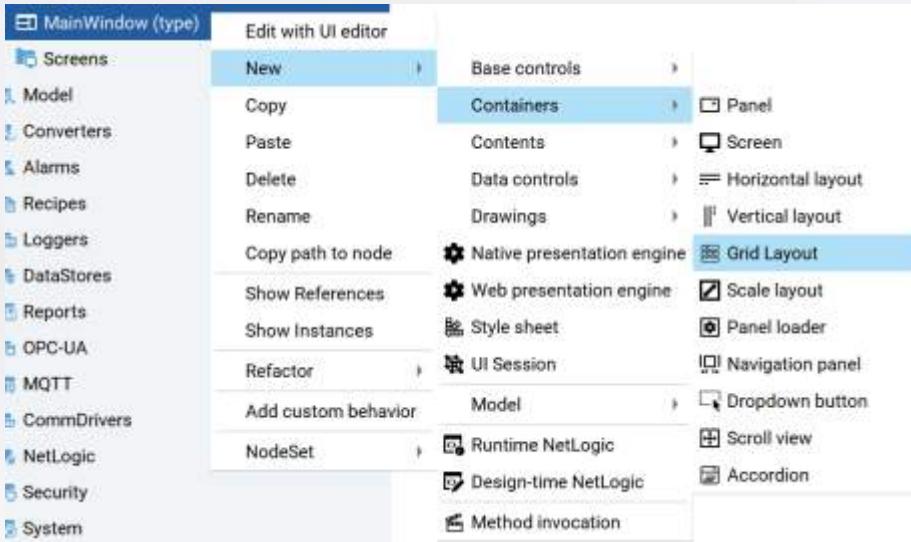
Visible	True
Enabled	True
Opacity	100
Rotation	0
Move target	None

## — Size and layout

Horizontal alignment	Stretch
Vertical alignment	Stretch
Left margin	50
Top margin	50
Right margin	50
Bottom margin	50

# Grid Layout

- Automatically lay out content based on configured grid pattern

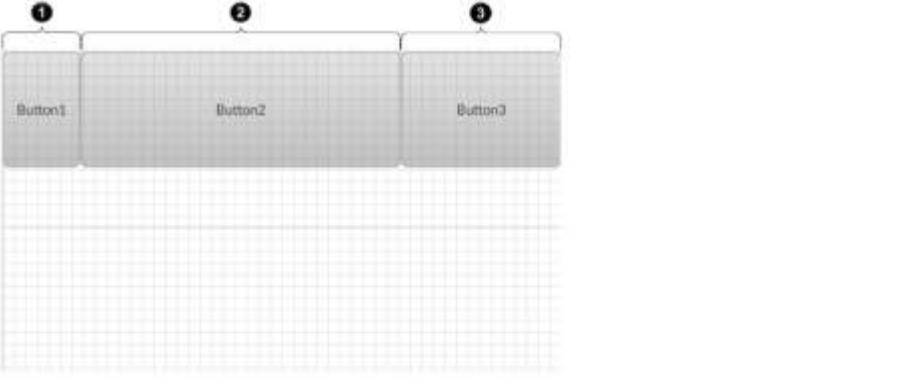


— Grid Layout	
Columns	1fr,2fr,3fr
Rows	1fr,2fr,3fr
Horizontal gap	0
Vertical gap	0
Items Horizontal Alignment	Stretch
Items Vertical Alignment	Stretch

- Columns: Specify column width in units of static pixels or relative frame units ( $1\text{fr} = 1\text{ unit}$ ,  $2\text{fr} = 2\text{ frame units}$ )
- Rows: Specify column width in units of static pixels or relative frame units ( $1\text{fr} = 1\text{ unit}$ ,  $2\text{fr} = 2\text{ frame units}$ )
- Horizontal gap: Space between columns
- Vertical gap: Space between rows
- Items Horizontal Alignment: Column content alignment
- Items Vertical Alignment: Row content alignment

# Grid Layout

- Grid Pattern Configuration



The diagram shows a 3x3 grid layout on a grid background. Three callouts point to the first column:

- Callout 1:** Points to the first column header. Value: Pixels, Unit: 100, Description: Fixed width of 100px.
- Callout 2:** Points to the second column header. Value: 2fr, Unit: Frames, Description: 2/3 of the remaining space.
- Callout 3:** Points to the third column header. Value: 1fr, Unit: Frames, Description: 1/3 of the remaining space.

- Table 1. Grid layout columns width explained

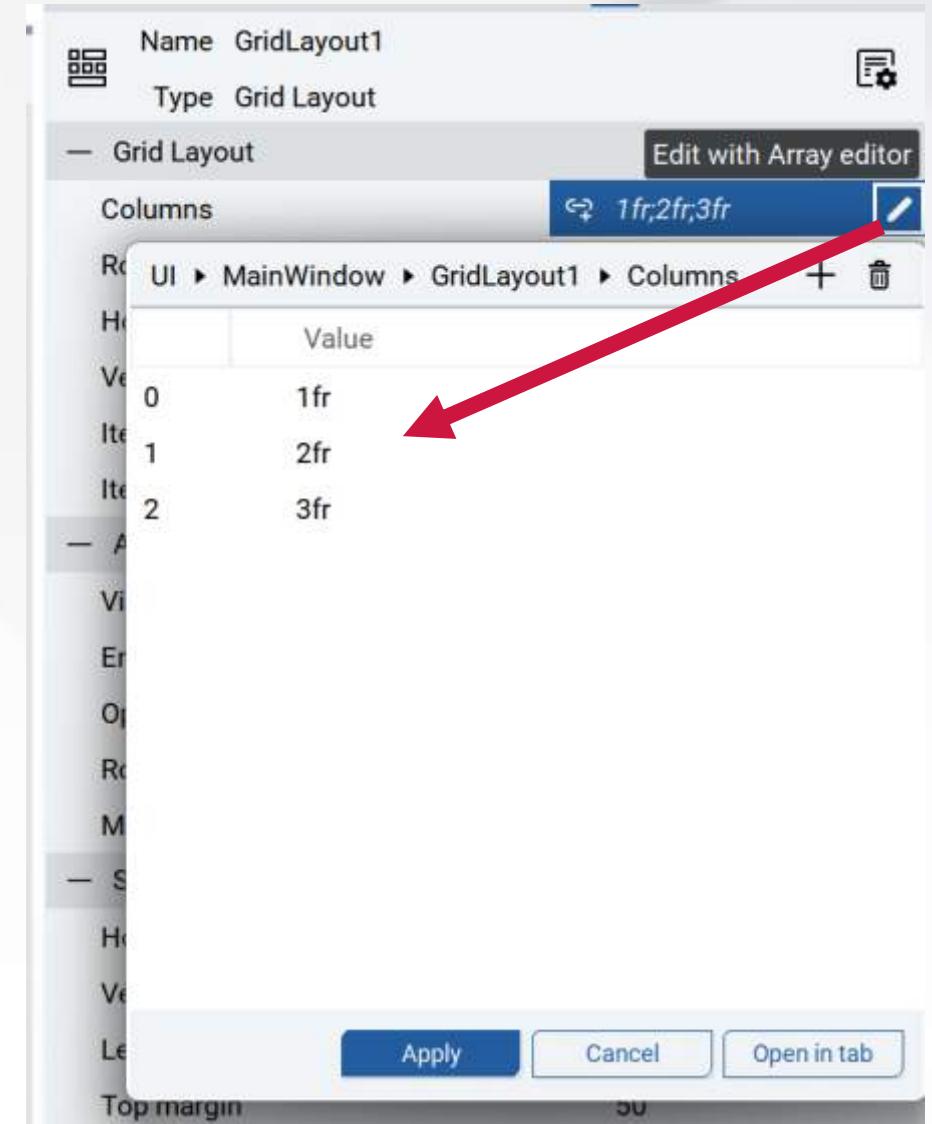
Callout	Unit	Value	Description
1	Pixels	100	Fixed width of 100px
2	Frames	2fr	2/3 of the remaining space
3	Frames	1fr	1/3 of the remaining space

- Columns and Rows are arrays of values dictating content behaviour
- Column and Row array values can be static pixels or dynamic at runtime using xfr notation
- Column and Row array values are separated by semicolon
- Example:
  - 3x3 grid
  - Column 1 = 1/6 of Grid Layout Width
  - Column 2 = 1/3 of Grid Layout Width
  - Column 3 = 1/2 of Grid Layout Width
  - Row 1 = 1/6 of Grid Layout Height
  - Row 2 = 1/3 of Grid Layout Height
  - Row 3 = 1/2 of Grid Layout Height

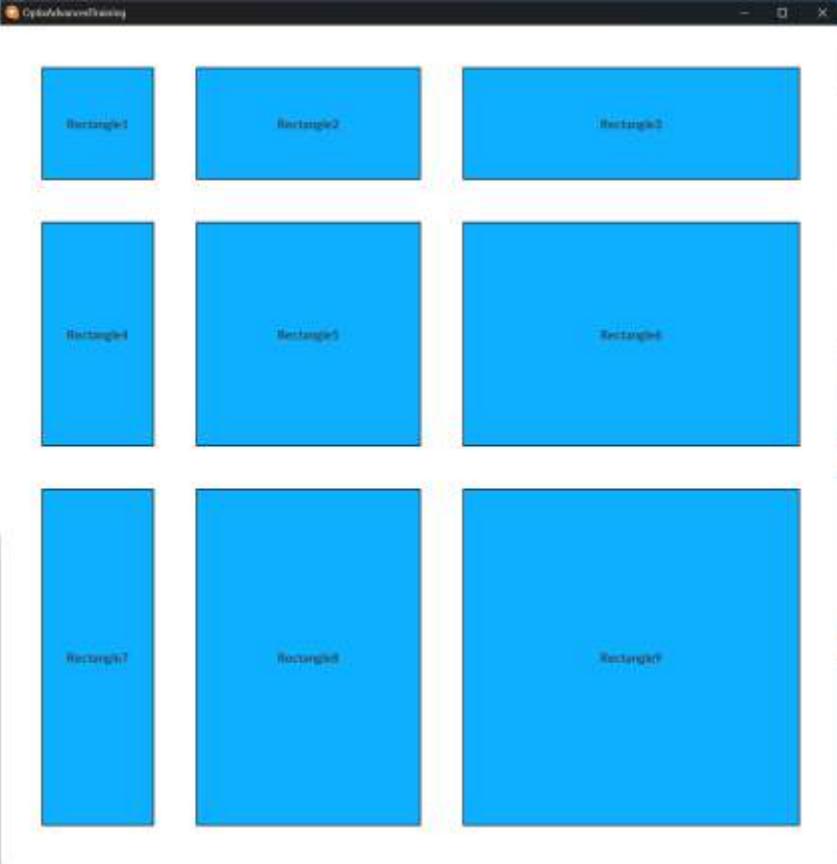
— Grid Layout	
Columns	1fr,2fr,3fr
Rows	1fr,2fr,3fr

# Grid Layout

- Grid Pattern Configuration
  - Add / Delete Columns and Rows using the Edit (Pencil) button



# Grid Layout

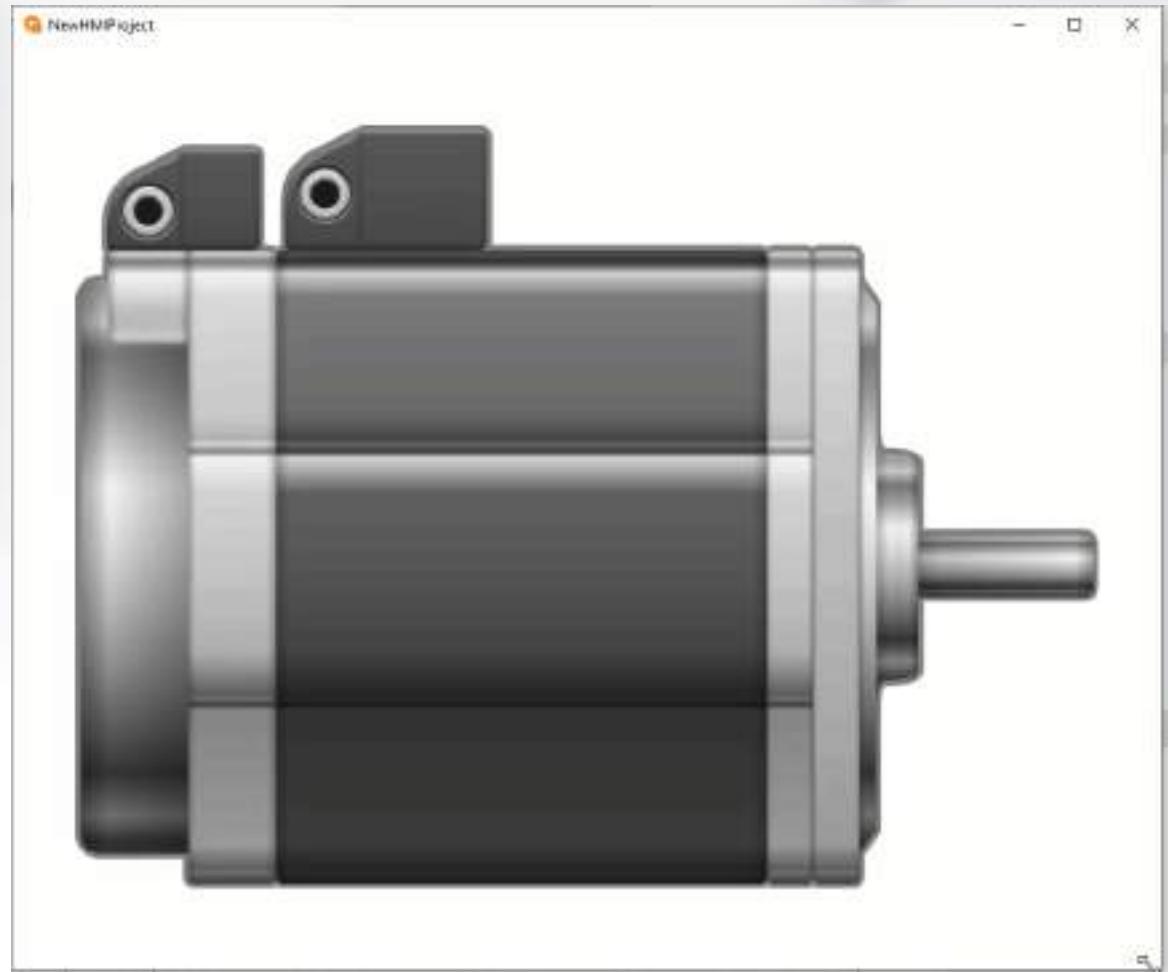


The screenshot shows a Windows application window titled "OptimizationTraining". Inside, there is a "MainWindow" containing a "GridLayout1" control. The "GridLayout1" contains nine blue rectangular elements labeled "Rectangle1" through "Rectangle9". The "MainWindow" has a standard Windows-style title bar with minimize, maximize, and close buttons. On the left, a tree view displays the window structure: "MainWindow (type)" contains "GridLayout1", which in turn contains "Rectangle1" through "Rectangle9". To the right of the window is a properties panel for "GridLayout1". The properties panel includes:

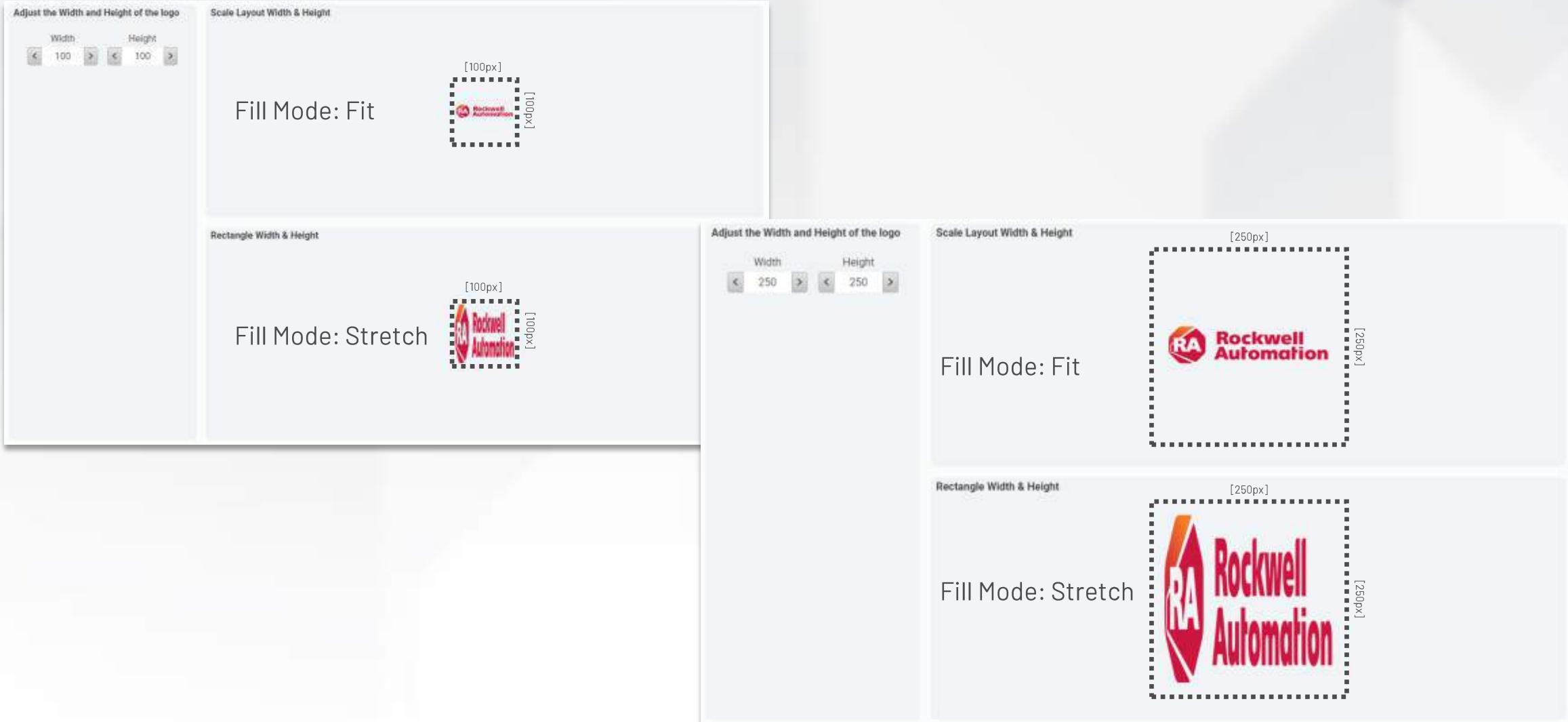
Name	GridLayout1
Type	Grid Layout
Columns	1fr;2fr;3fr
Rows	1fr;2fr;3fr
Horizontal gap	50
Vertical gap	50
Items Horizontal Alignment	Stretch
Items Vertical Alignment	Stretch
Appearance	
Visible	True
Enabled	True
Opacity	100
Rotation	0
Move target	None
Size and layout	
Horizontal alignment	Stretch
Vertical alignment	Stretch
Left margin	50
Top margin	50
Right margin	50
Bottom margin	50

# ScaleLayout

- Objects can be adapted to the size of the container using a **ScaleLayout**
- Scaling ratio can be set to:
  - Fit: original ratio is maintained, empty bars can appear around the element if the container's ratio is different from the object's ratio
  - Stretch: object is stretch to the container size, ratio is ignored, graphics may get distorted

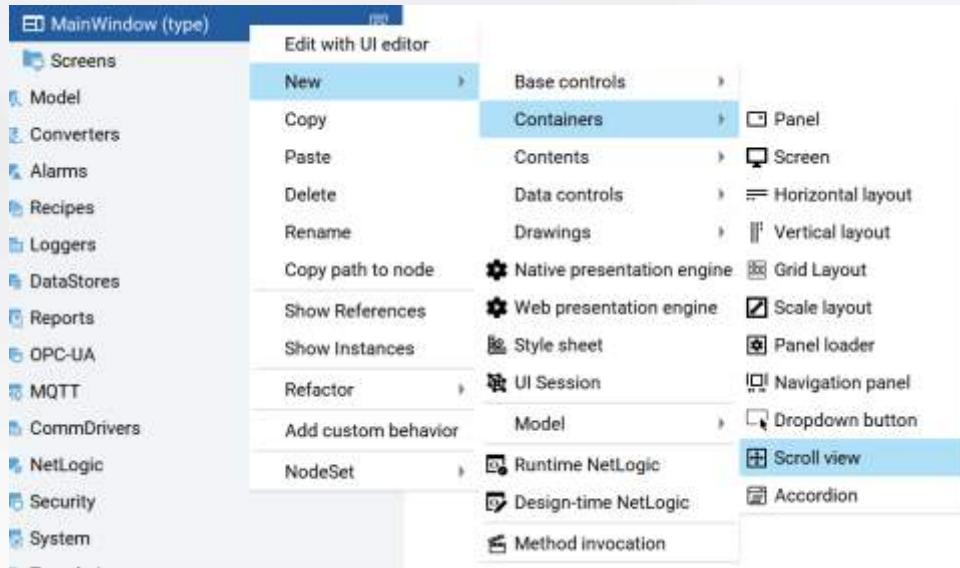


# Scale Layout



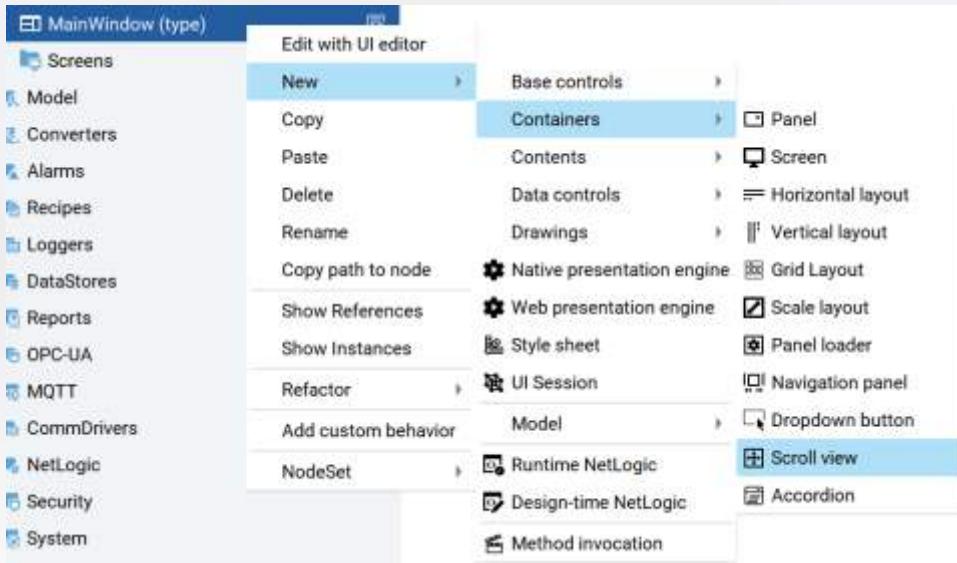
# ScrollView

- Pan and Zoom on content
  - Zoom supported with Ctrl+Mouse Scroll or Multi-touch pinch



# ScrollView

- Pan and Zoom on content



— ScrollView	
Enables zoom	False
Maximum zoom in	100
Minimum zoom out	100
Scroll axes enabled	Both

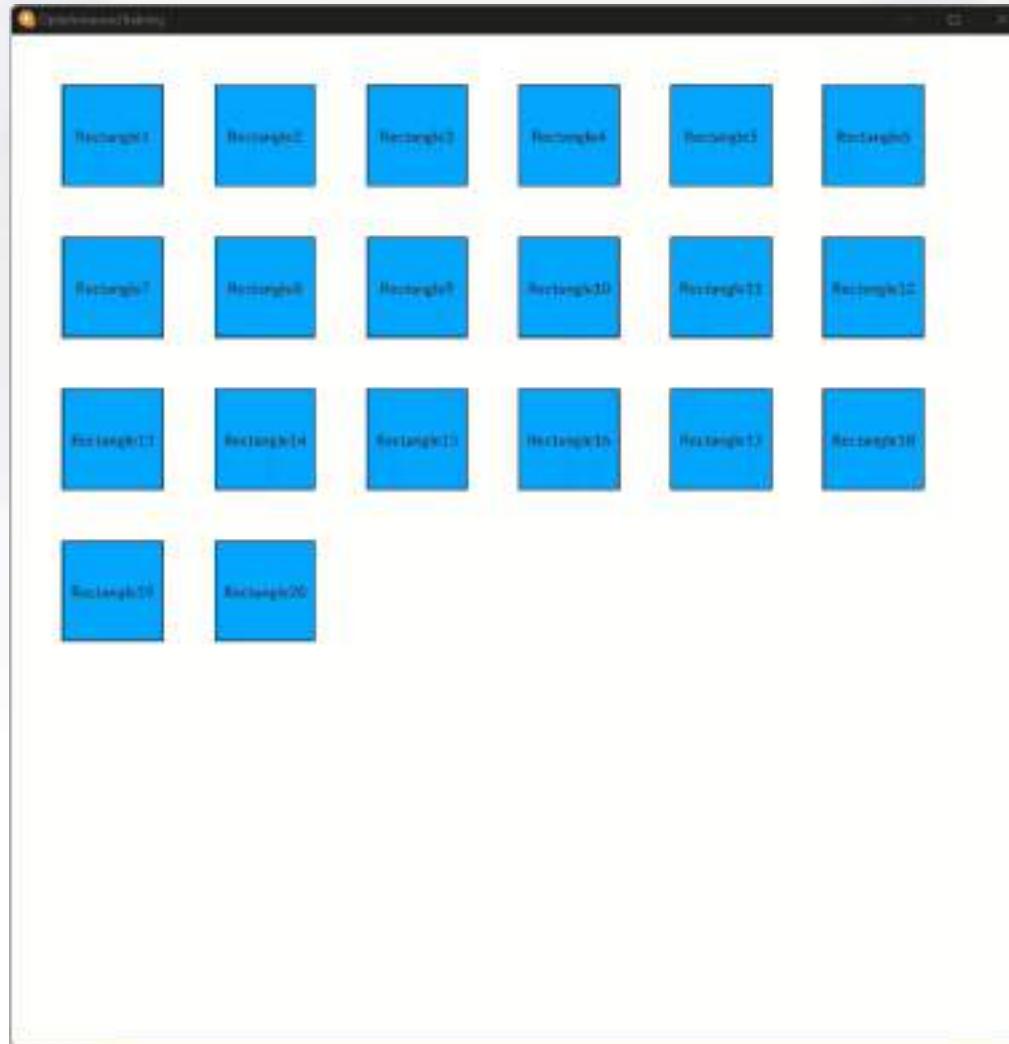
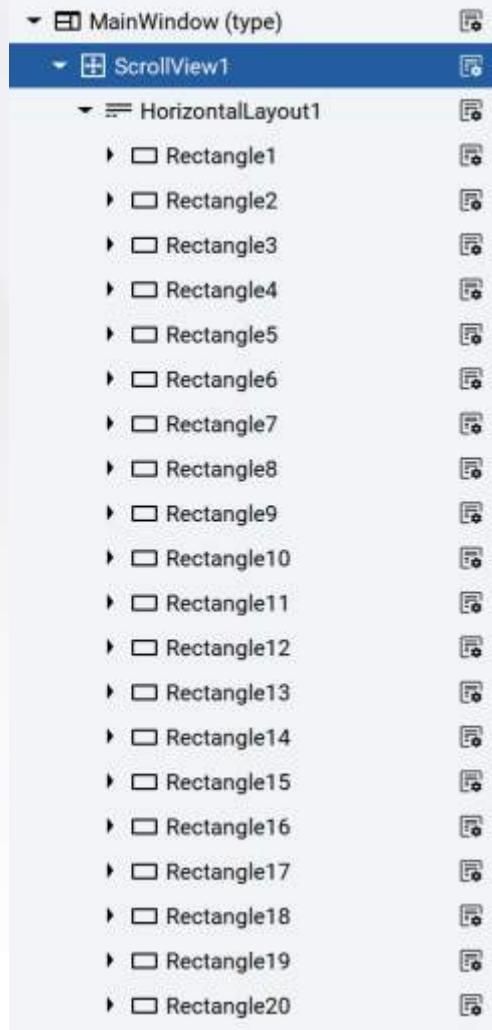
- Enables zoom: Turn on / off zoom capabilities
- Maximum / Minimum zoom in / out: Set zoom boundaries as a percentage
- Scroll axes enabled: Control scroll directions available (X Axis, Y Axis, Both)

# ScrollView + Horizontal Layout (Wrapped)

The screenshot shows a configuration interface with three main sections:

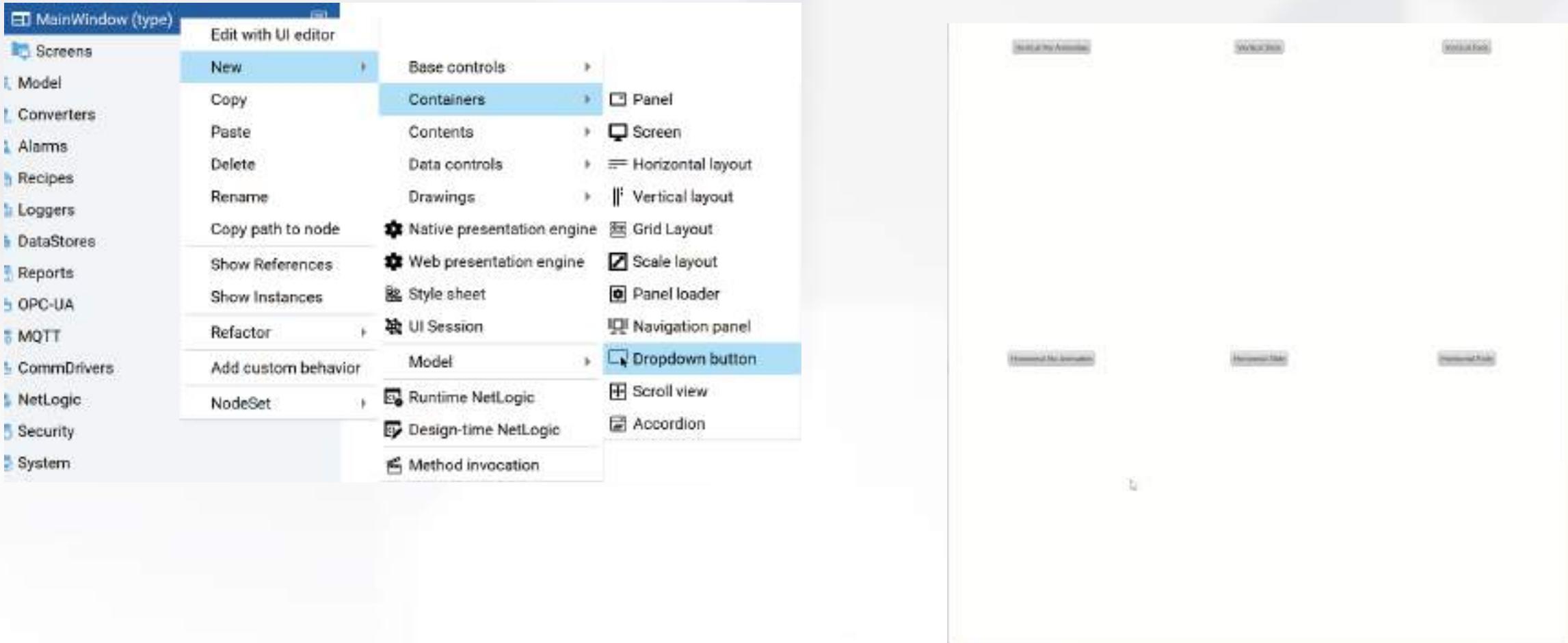
- MainWindow (type) Tree View:** On the left, it lists the structure of the application:
  - MainWindow (type)
    - ScrollView1
      - HorizontalLayout1
        - Rectangle1
        - Rectangle2
        - Rectangle3
        - Rectangle4
        - Rectangle5
        - Rectangle6
        - Rectangle7
        - Rectangle8
        - Rectangle9
        - Rectangle10
        - Rectangle11
        - Rectangle12
        - Rectangle13
        - Rectangle14
        - Rectangle15
        - Rectangle16
        - Rectangle17
        - Rectangle18
        - Rectangle19
        - Rectangle20
  - ScrollView1 Properties:** In the center, the properties for ScrollView1 are displayed:
    - Name: ScrollView1
    - Type: Scroll view
    - ScrollView
      - Enables zoom: False
      - Maximum zoom in: 100
      - Minimum zoom out: 100
      - Scroll axes enabled: Both
    - Appearance
      - Visible: True
      - Enabled: True
      - Opacity: 100
      - Rotation: 0
      - ScrollBar style: Default
    - Size and layout
      - Horizontal alignment: Stretch
      - Vertical alignment: Stretch
      - Left margin: 0
      - Top margin: 0
      - Right margin: 0
      - Bottom margin: 0
  - HorizontalLayout1 Properties:** On the right, the properties for HorizontalLayout1 are displayed:
    - Name: HorizontalLayout1
    - Type: Horizontal layout
    - Horizontal Layout
      - Content alignment: Left aligned
      - Wrap: True
      - Horizontal gap: 50
      - Vertical gap: 50
    - Appearance
      - Visible: True
      - Enabled: True
      - Opacity: 100
      - Rotation: 0
      - Move target: None
    - Size and layout
      - Horizontal alignment: Stretch
      - Vertical alignment: Top
      - Height: Auto
      - Left margin: 50
      - Top margin: 50
      - Right margin: 50

# ScrollView + Horizontal Layout (Wrapped)



# Dropdown button

- Present another container where clicked (i.e. Drop down menus or Faceplate-like functionality)



# Dropdown button

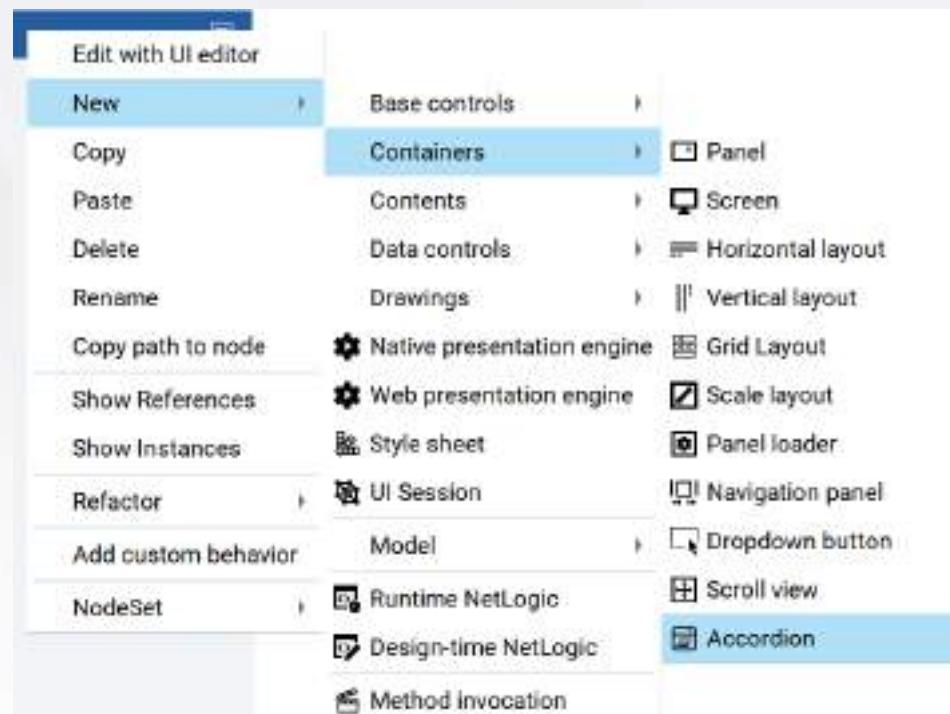
- Present another container where clicked (i.e. Drop down menus or Faceplate-like functionality)

— DropDown Button	
Panel	Faceplate (type)
Panel alias node	
Direction	Vertical
Animation	None
Close on click inside	False
Close on click outside	True

- Panel: Panel to load on click
- Panel alias node: Aliases to pass to Panel
- Direction: Vertical or Horizontal
- Animation: None, Slide, Fade
- Close on Click inside / outside: Close panel behavior

# Accordion

- Configurable Dropdown button with nesting capabilities



A screenshot of the FactoryTalk Design interface titled 'DESIGN / Accordion'. It displays three examples of nested Accordions:

- Example 1:** Shows a single 'Accordion 1' component.
- Example 2:** Shows two nested Accordions: 'Accordion 1' which contains 'Accordion 2'.
- Example 3:** Shows three nested Accordions: 'Accordion 1' which contains 'Accordion 2', which in turn contains 'Accordion 3'.

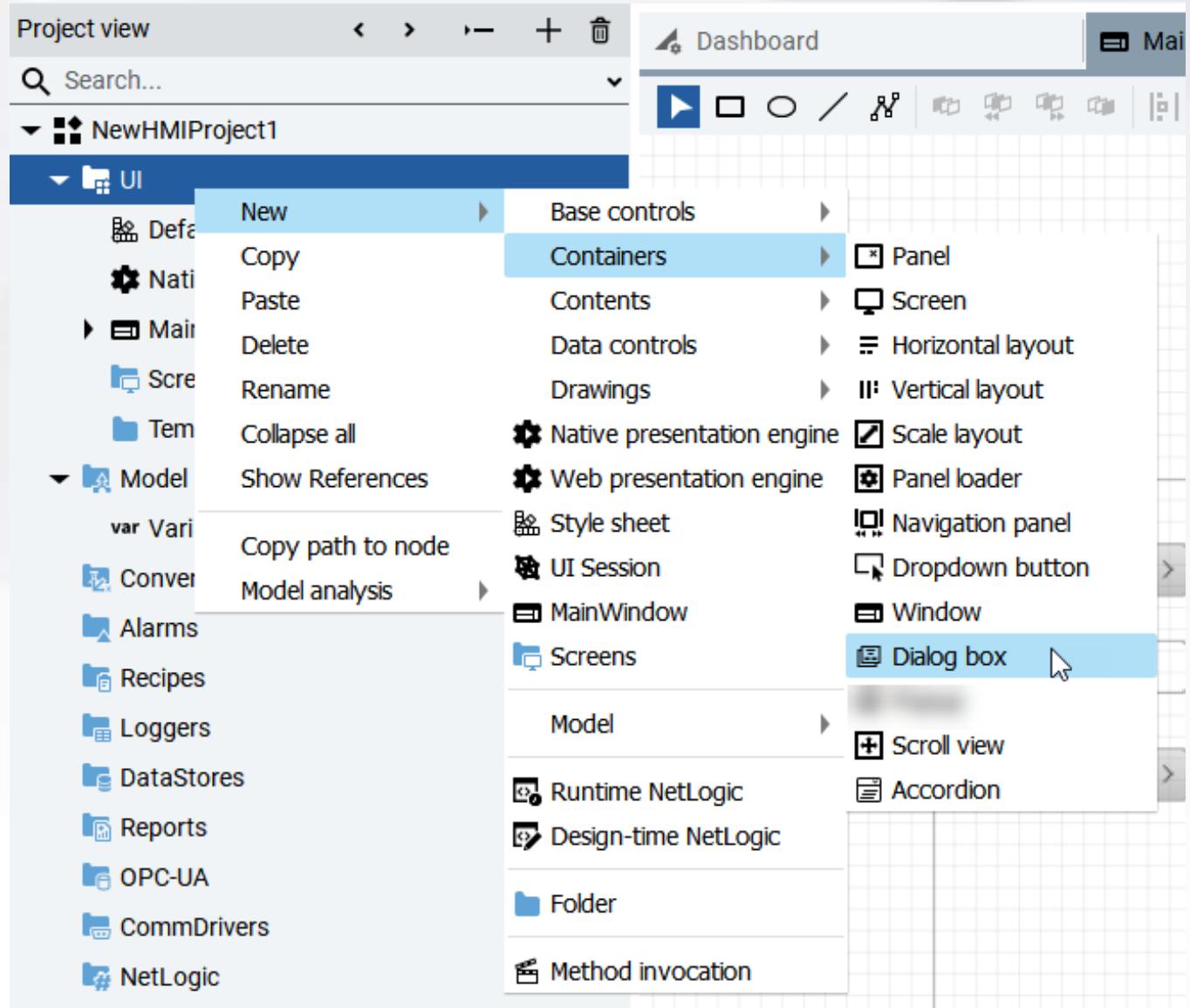
# Accordion

- Configurable Dropdown button with nesting capabilities



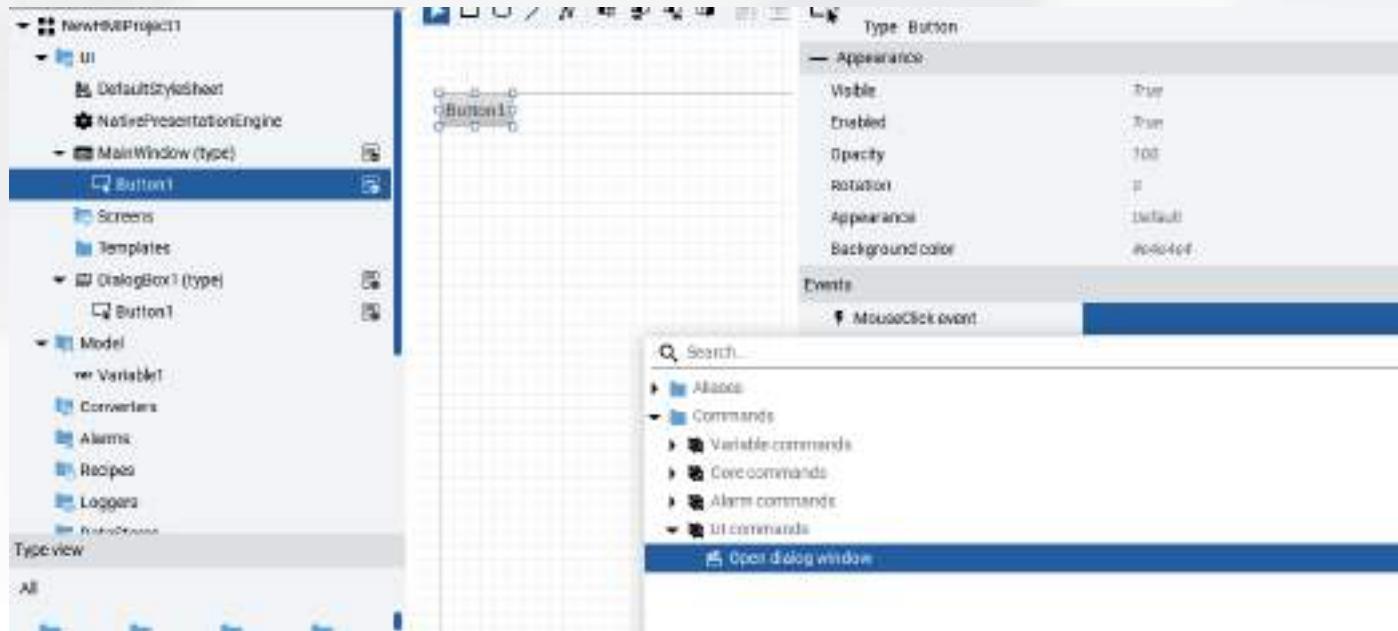
# DialogBox

- Popup displays in Optix are called «DialogBox»
- DialogBox must be a type
- DialogBoxes are transparent by default
- **Note:** DialogBox does not open a new window on the device, it is a container on the highest Z-level of the MainWindow
- The «Close» method is exposed by the DialogBox itself



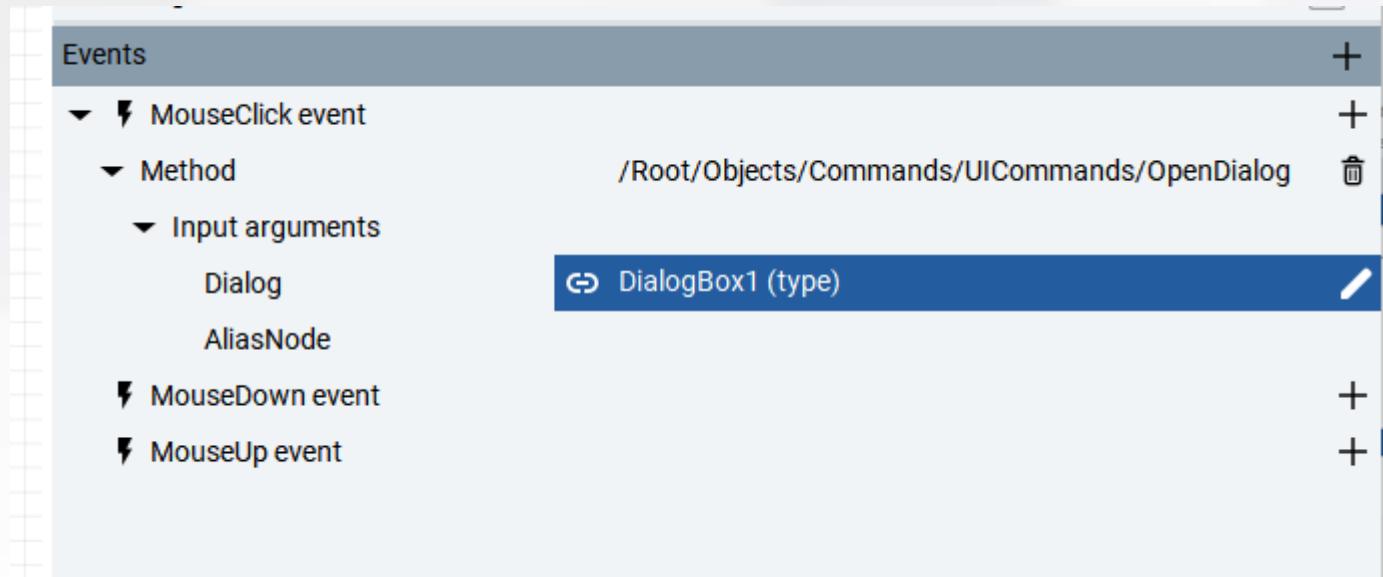
# DialogBox

- Popup displays in Optix are called «DialogBox»
- DialogBox must be a type
- DialogBoxes are transparent by default
- **Note:** DialogBox does not open a new window on the device, it is a container on the highest Z-level of the MainWindow
- The «Close» method is exposed by the DialogBox itself



# DialogBox

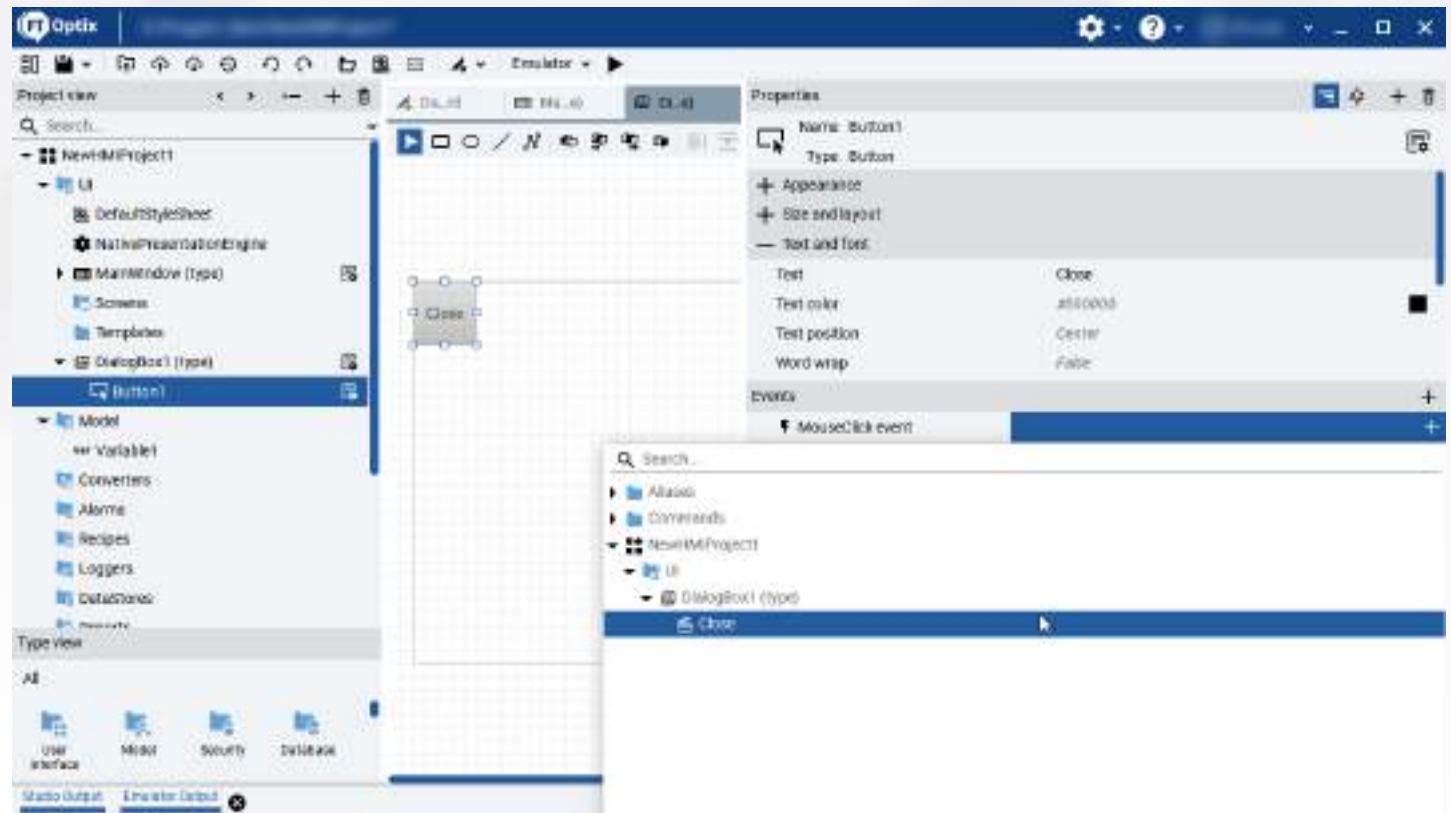
- Popup displays in Optix are called «DialogBox»
- DialogBox must be a type
- DialogBoxes are transparent by default
- **Note:** DialogBox does not open a new window on the device, it is a container on the highest Z-level of the MainWindow



- The «Close» method is exposed by the DialogBox itself

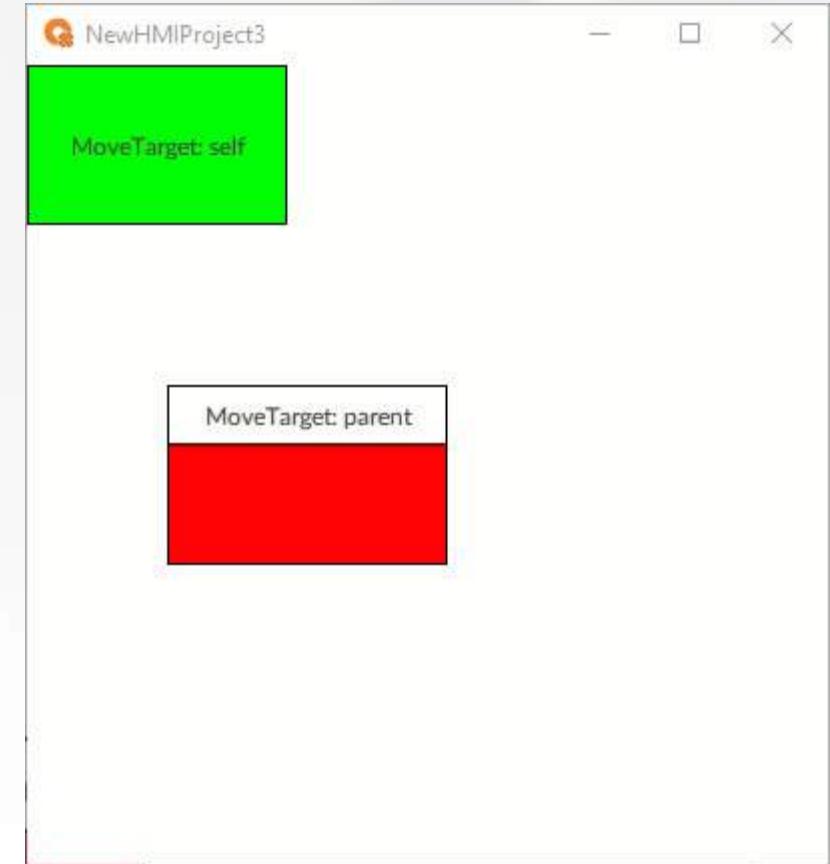
# DialogBox

- Popup displays in Optix are called «DialogBox»
- DialogBox must be a type
- DialogBoxes are transparent by default
- **Note:** DialogBox does not open a new window on the device, it is a container on the highest Z-level of the MainWindow
- The «Close» method is exposed by the DialogBox itself



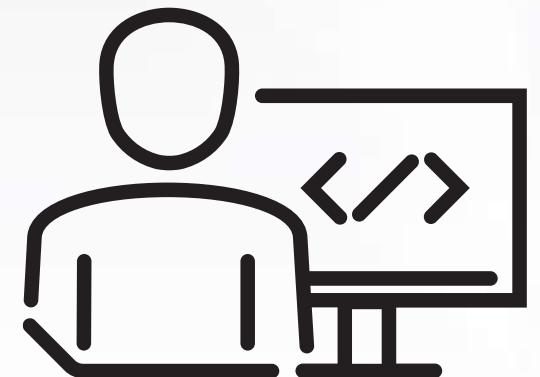
# Move Target

- Elements can be moved on the screen with the MoveTarget property
  - Self -> Dragging the object will move the object itself (and its children)
  - Parent -> When selecting any of the parent objects, dragging the control will move the whole parent and all children (e.g: dragging the title bar moves the whole DialogBox)



# Hands-on session

- Create a simple HMI with two or three screens and a Navigation Panel
- Add a web presentation engine and connect to the emulator using a browser
- Try to make the graphics responsive

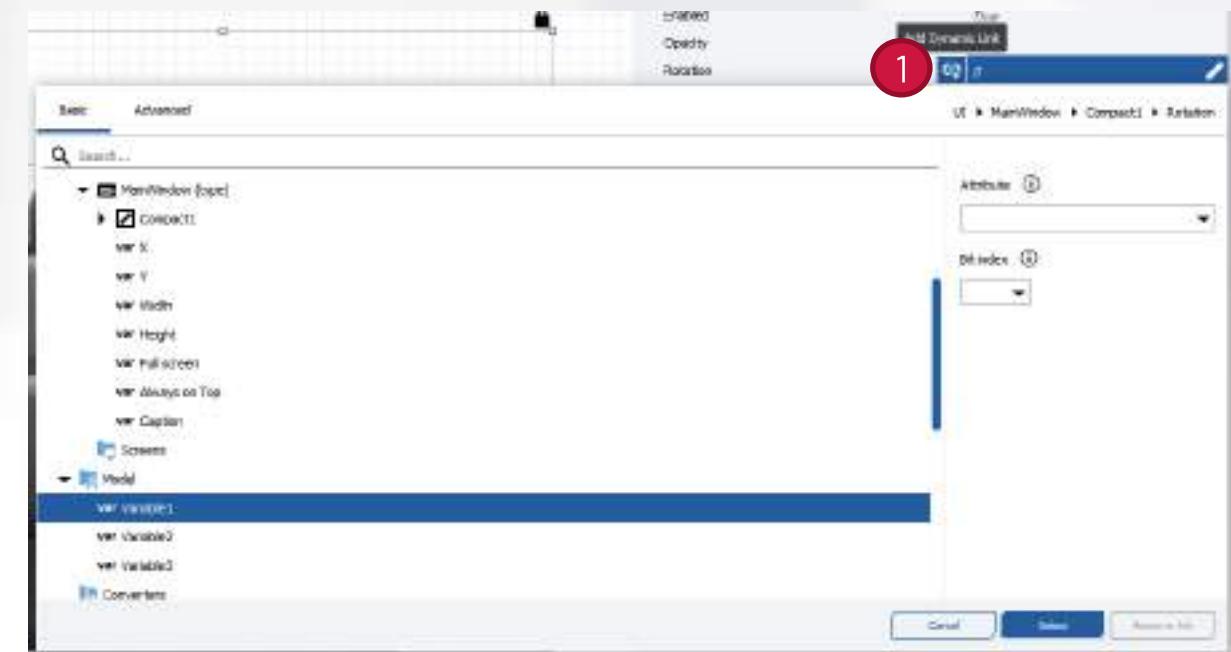


# Dynamic links



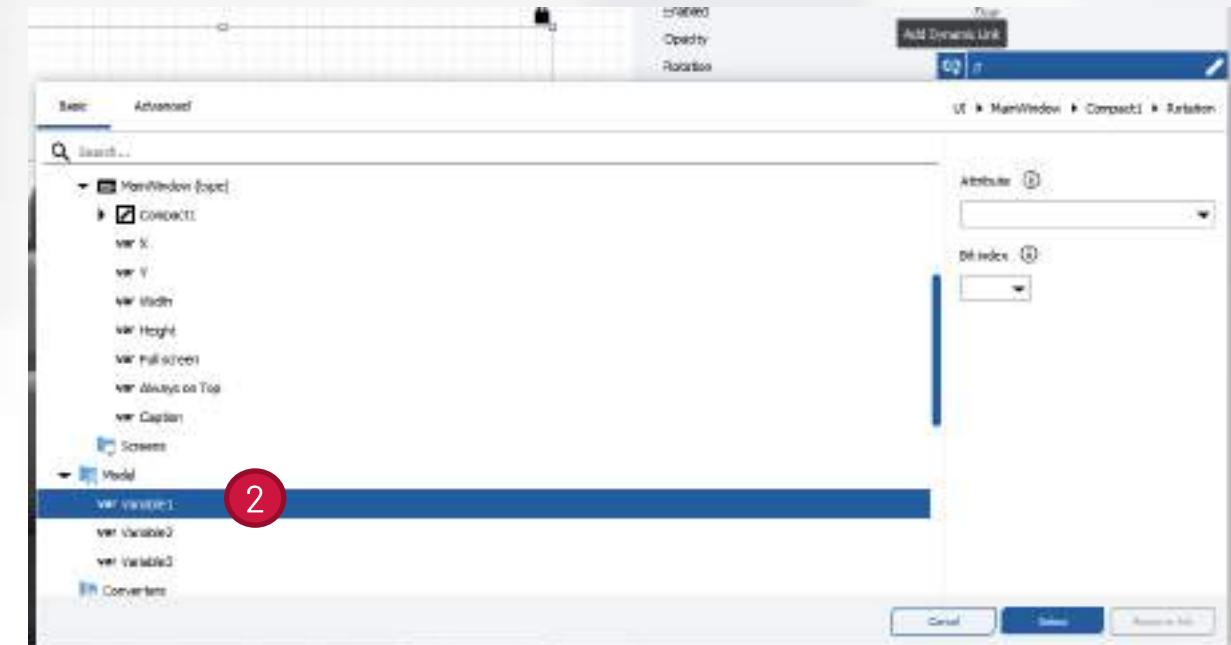
# Dynamic links

- Dynamic Link is a connection between two nodes
  - Property connected to a Variable  
absolute link = "Variable1"
  - Property connected to a Property of the same object  
relative link = "../Enabled"
  - Property connected to a Property of another object  
relative link = "../../Switch1/Checked"
- Can be created in different ways
  - Manual selection using the dynamic link window
  - Dragging the variable on a property
  - Dragging the variable on the graphical object  
in this case, Studio sets the dynamic link  
on the most suitable property, based on heuristics.



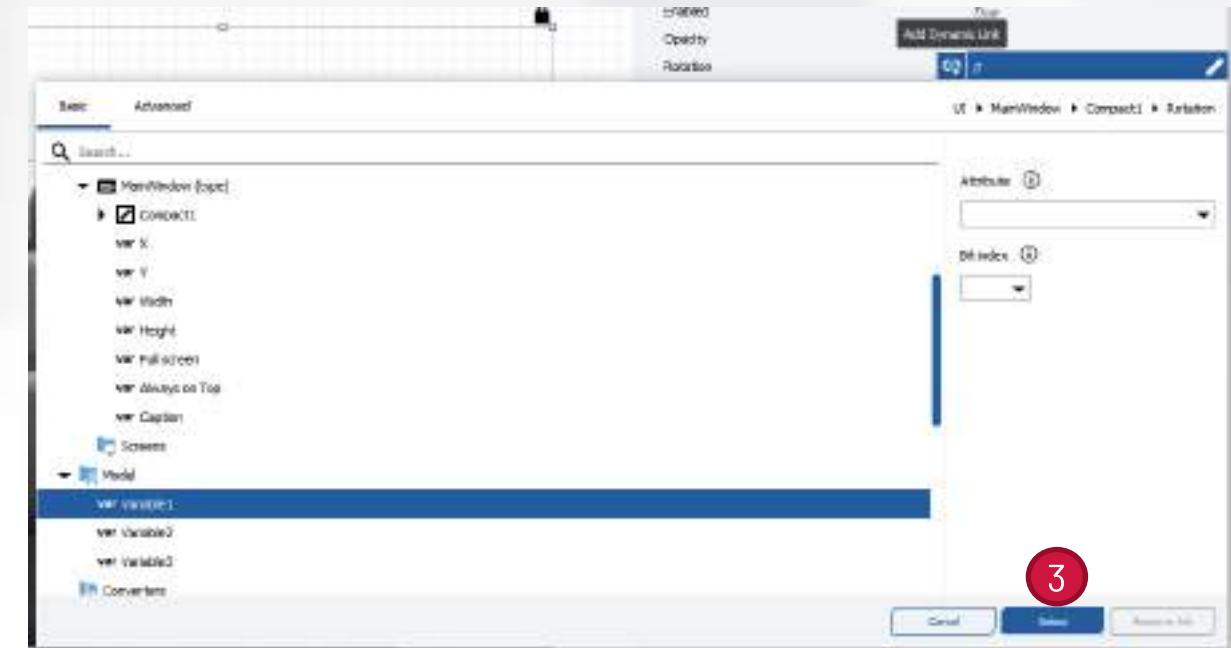
# Dynamic links

- Dynamic Link is a connection between two nodes
  - Property connected to a Variable  
absolute link = "Variable1"
  - Property connected to a Property of the same object  
relative link = "../Enabled"
  - Property connected to a Property of another object  
relative link = ".../Switch1/Checked"
- Can be created in different ways
  - Manual selection using the dynamic link window
  - Dragging the variable on a property
  - Dragging the variable on the graphical object  
in this case, Studio sets the dynamic link  
on the most suitable property, based on heuristics.



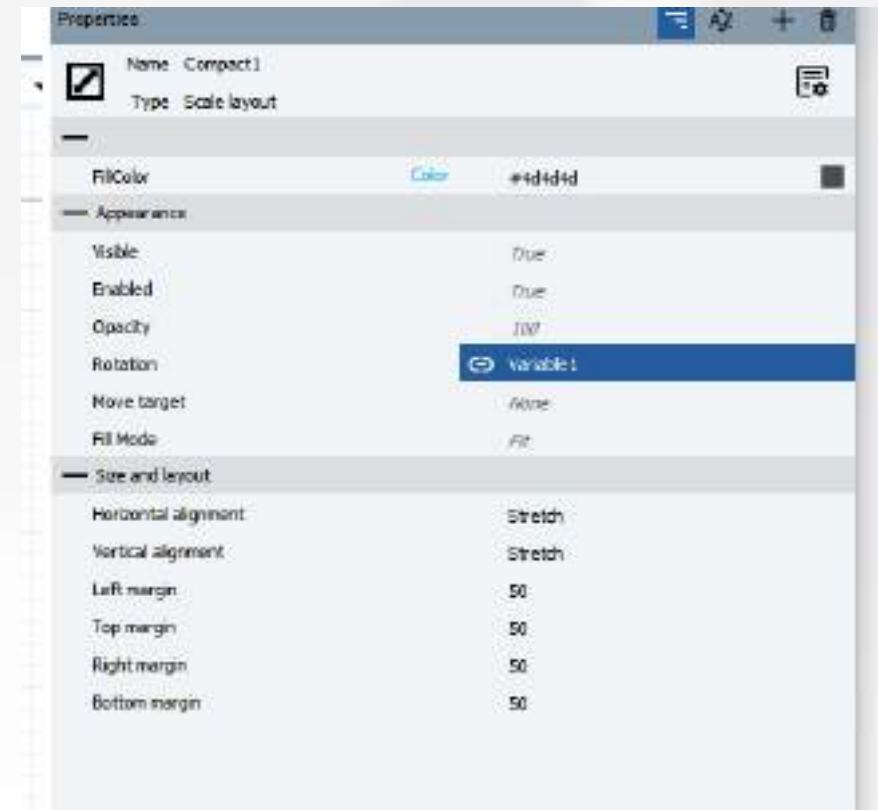
# Dynamic links

- Dynamic Link is a connection between two nodes
  - Property connected to a Variable  
absolute link = "Variable1"
  - Property connected to a Property of the same object  
relative link = "../Enabled"
  - Property connected to a Property of another object  
relative link = "../../Switch1/Checked"
- Can be created in different ways
  - Manual selection using the dynamic link window
  - Dragging the variable on a property
  - Dragging the variable on the graphical object  
in this case, Studio sets the dynamic link  
on the most suitable property, based on heuristics.



# Dynamic links

- Dynamic Link is a connection between two nodes
  - Property connected to a Variable  
absolute link = "Variable1"
  - Property connected to a Property of the same object  
relative link = "../Enabled"
  - Property connected to a Property of another object  
relative link = ".../Switch1/Checked"
- Can be created in different ways
  - Manual selection using the dynamic link window
  - Dragging the variable on a property
  - Dragging the variable on the graphical object  
in this case, Studio sets the dynamic link  
on the most suitable property, based on heuristics.



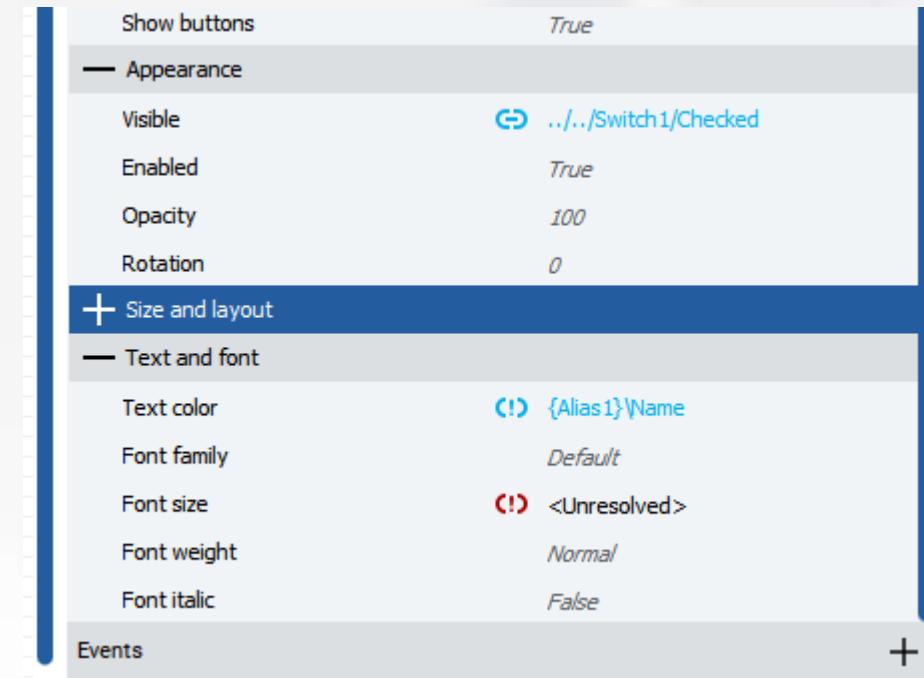
# Dynamic links

- Dynamic Link is a connection between two nodes
  - Property connected to a Variable  
absolute link = "Variable1"
  - Property connected to a Property of the same object  
relative link = "../Enabled"
  - Property connected to a Property of another object  
relative link = ".../Switch1/Checked"
- Can be created in different ways
  - Manual selection using the dynamic link window
  - Dragging the variable on a property
  - Dragging the variable on the graphical object  
in this case, Studio sets the dynamic link  
on the most suitable property, based on heuristics.

FillColor	Color	#4d4d4d
<strong>Appearance</strong>		
Visible		True
Enabled		True
Opacity		../../Width
Rotation		Variable1
Move target		None
Fill Mode		Fit
<strong>Size and layout</strong>		
Horizontal alignment		Stretch

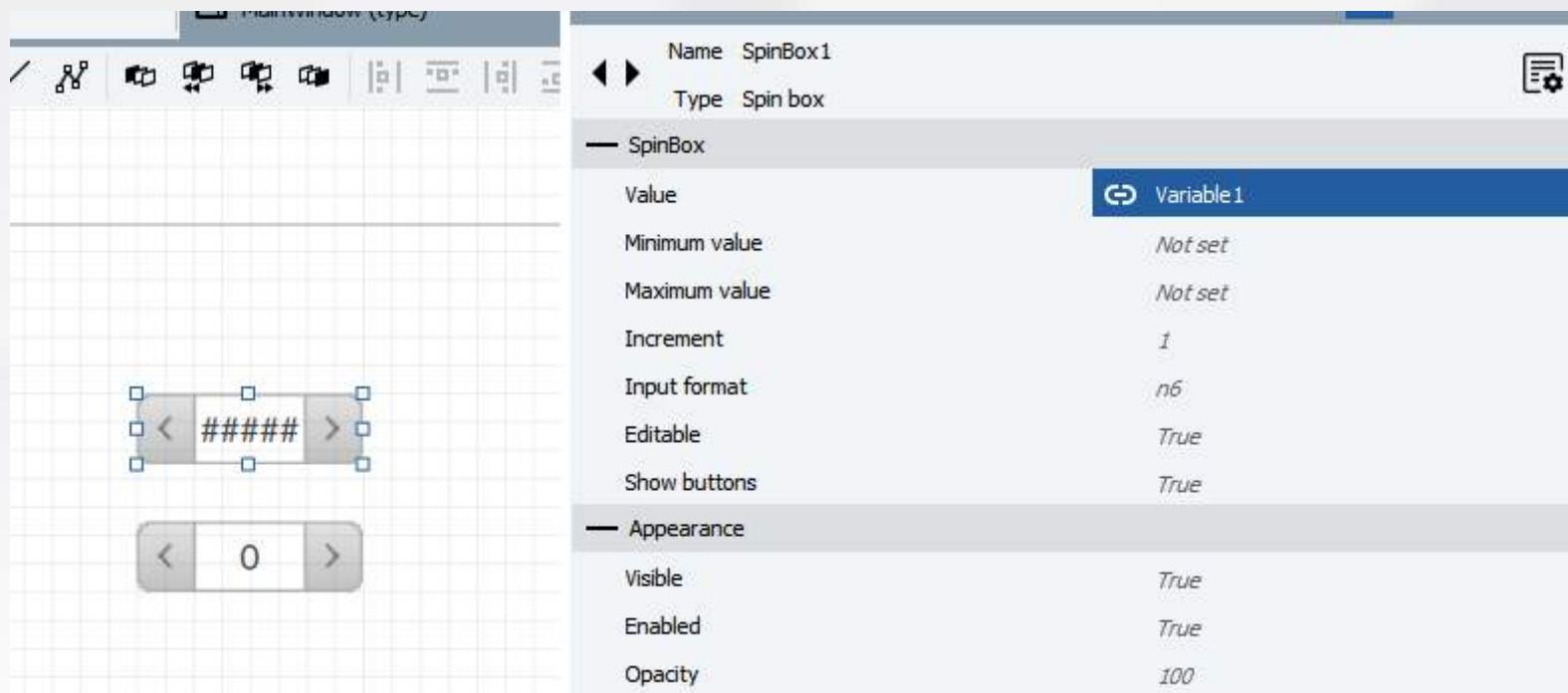
# Dynamic links indicators

- Dynamic links can be
  - **Valid:** solid blue indicator, target variable is reachable
  - **Unknown:** blue exclamation mark, the target variable is not reachable at DesignTime (an Alias for example) but can be good at Runtime
  - **Unresolved:** the target variable does not exist, DynamicLink is most likely broken



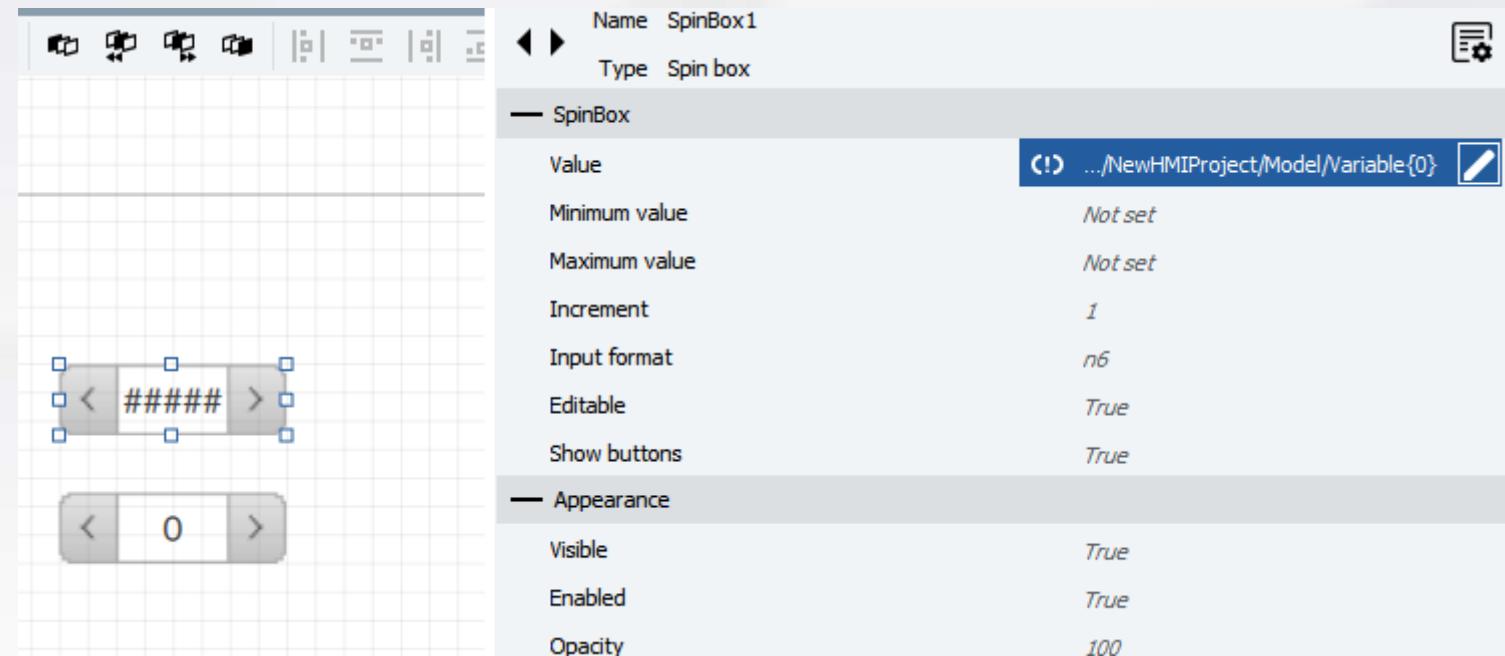
# Advanced dynamic links: indexing

- Variables can be indexed by using {0} placeholder



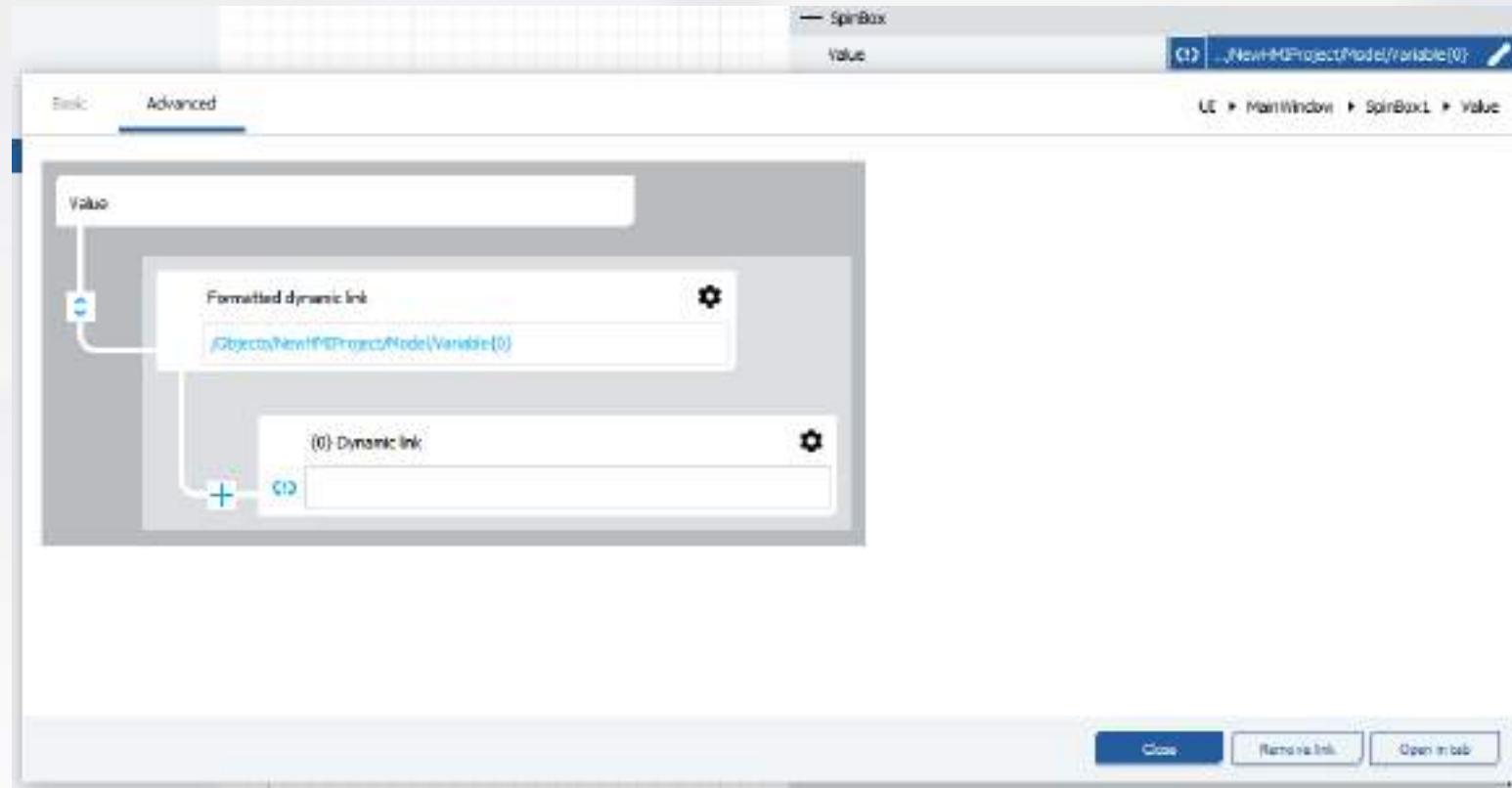
# Advanced dynamic links: indexing

- Variables can be indexed by using {0} placeholder



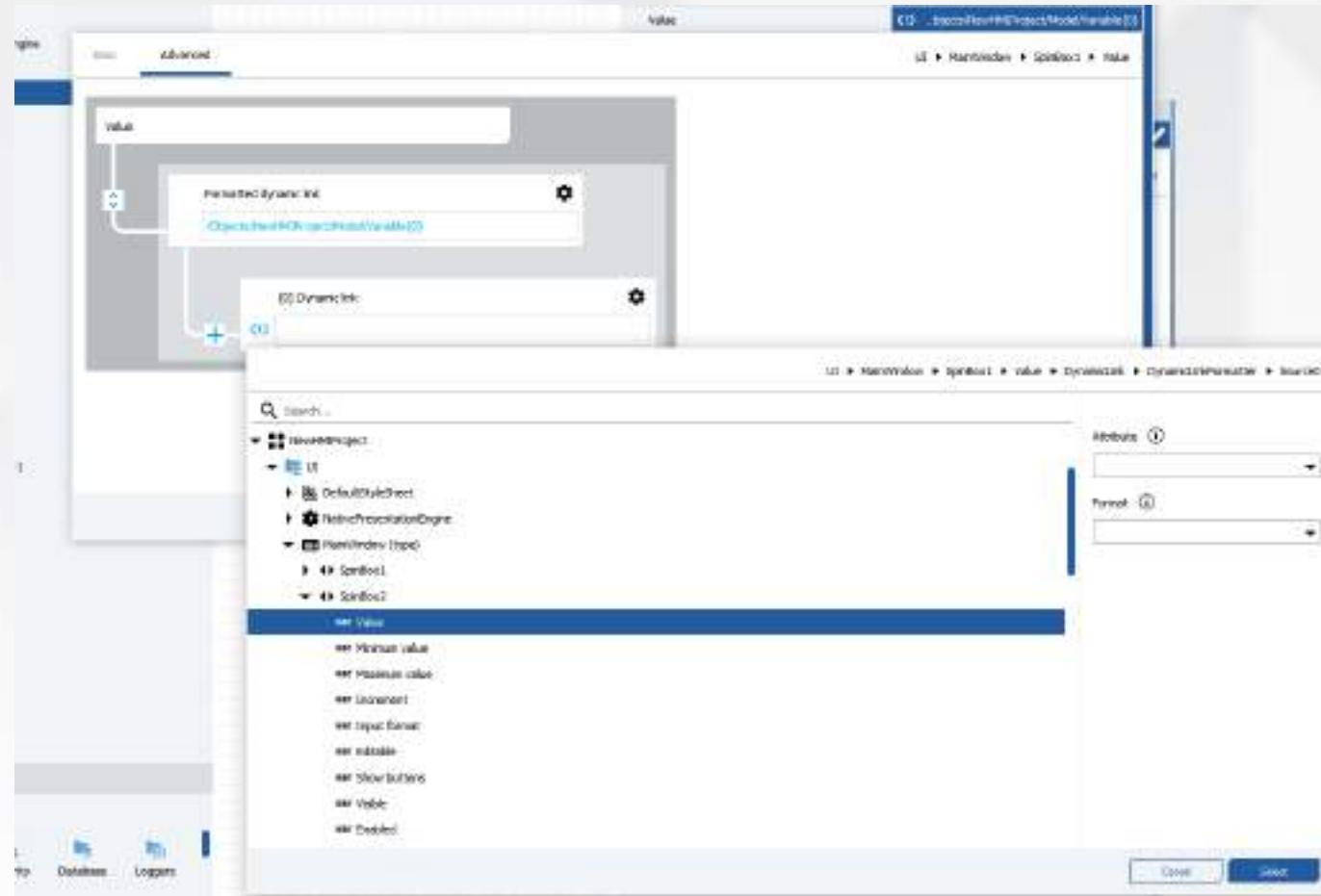
# Advanced dynamic links: indexing

- Variables can be indexed by using {0} placeholder



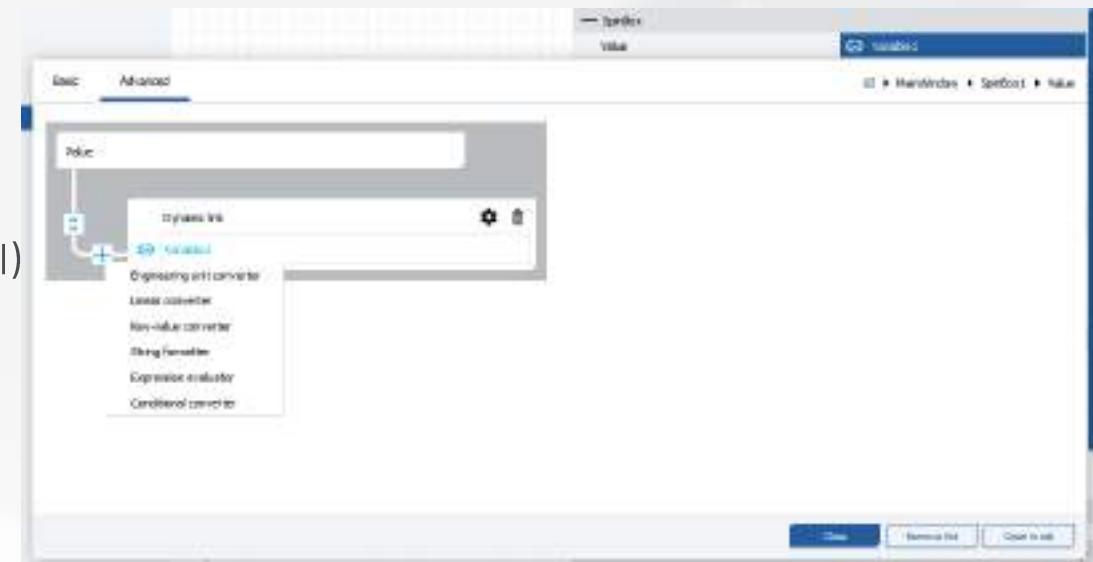
# Advanced dynamic links: indexing

- Variables can be indexed by using {0} placeholder



# Advanced dynamic links: converters

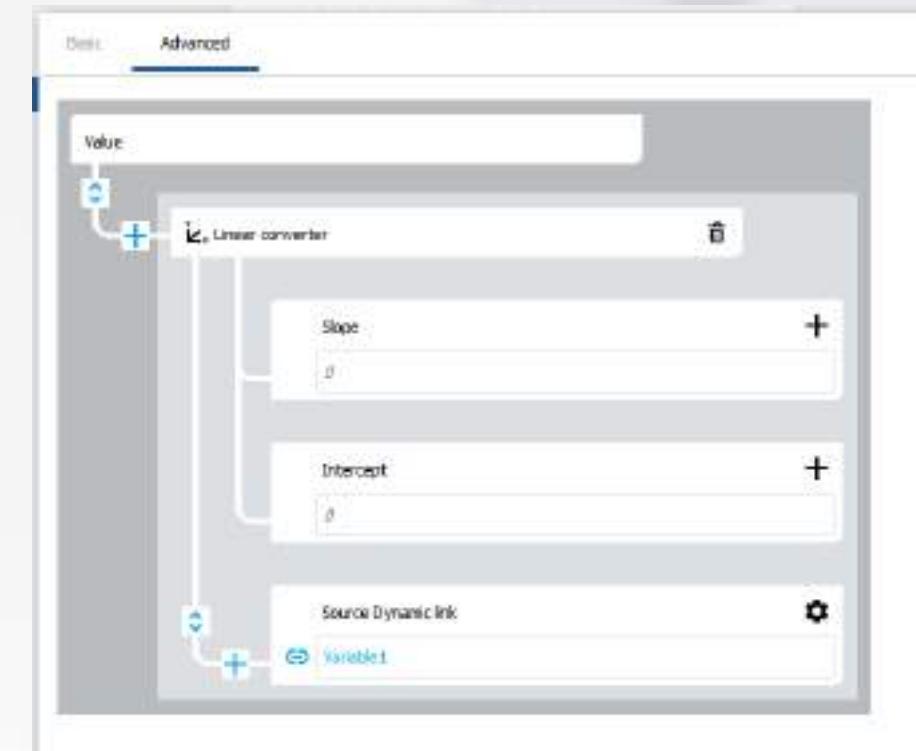
- By [Advanced...] button on Dynamic Link window, the value of a Property can be set through a Converter
  - **Engineering Unit converter**: it's a linear conversion defined by Raw range (PLC) and Scaled range (HMI)
  - **Linear converter**: it's a linear conversion defined by equation
$$y = mx + q$$
  - **Key-Value converter** \* : converts the value of the source variable according to a table of key-value pairs.
  - **String formatter**: modify the format of a Numeric, Date/Time, String, SQL Query
  - **Expression evaluator**: performs arithmetic calculations with one or more input values
  - **Conditional converter**: writes two different values according to a boolean Condition DynamicLink



\* Key-Value converter table can be exported to csv file and imported back, using a Template Library script

# Advanced dynamic links: converters

- By [Advanced...] button on Dynamic Link window, the value of a Property can be set through a Converter
  - **Engineering Unit converter**: it's a linear conversion defined by Raw range (PLC) and Scaled range (HMI)
  - **Linear converter**: it's a linear conversion defined by equation
$$y = mx + q$$
  - **Key-Value converter** \* : converts the value of the source variable according to a table of key-value pairs.
  - **String formatter**: modify the format of a Numeric, Date/Time, String, SQL Query
  - **Expression evaluator**: performs arithmetic calculations with one or more input values
  - **Conditional converter**: writes two different values according to a boolean Condition DynamicLink



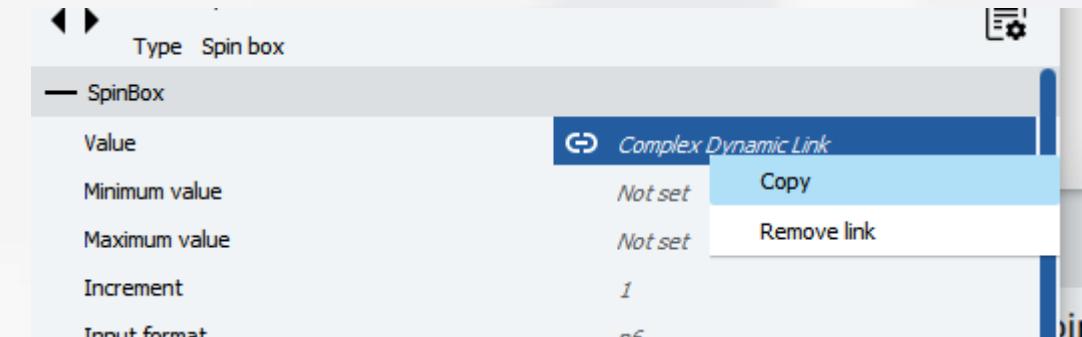
\* Key-Value converter table can be exported to csv file and imported back, using a Template Library script

# Advanced dynamic links: reusable converters

- Converters can be reused

- Using **copy/paste**

- Copy the converter from the property with "*Complex Dynamic Link*"
- Paste to another property



- Creating a **Converter as a Type**

- Create a Converter into Converters node
- The Converter will be automatically listed on the Advanced Dynamic Link configuration

# Advanced dynamic links: reusable converters

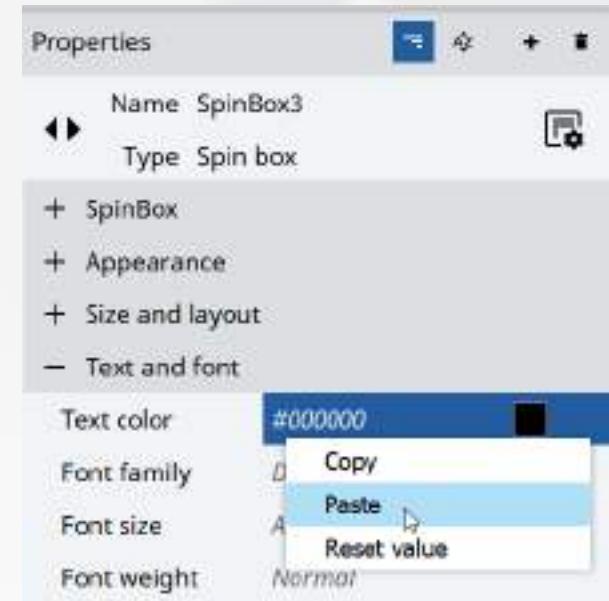
- Converters can be reused

- Using **copy/paste**

- Copy the converter from the property with "*Complex Dynamic Link*"
- Paste to another property

- Creating a **Converter as a Type**

- Create a Converter into Converters node
- The Converter will be automatically listed on the Advanced Dynamic Link configuration



# Advanced dynamic links: reusable converters

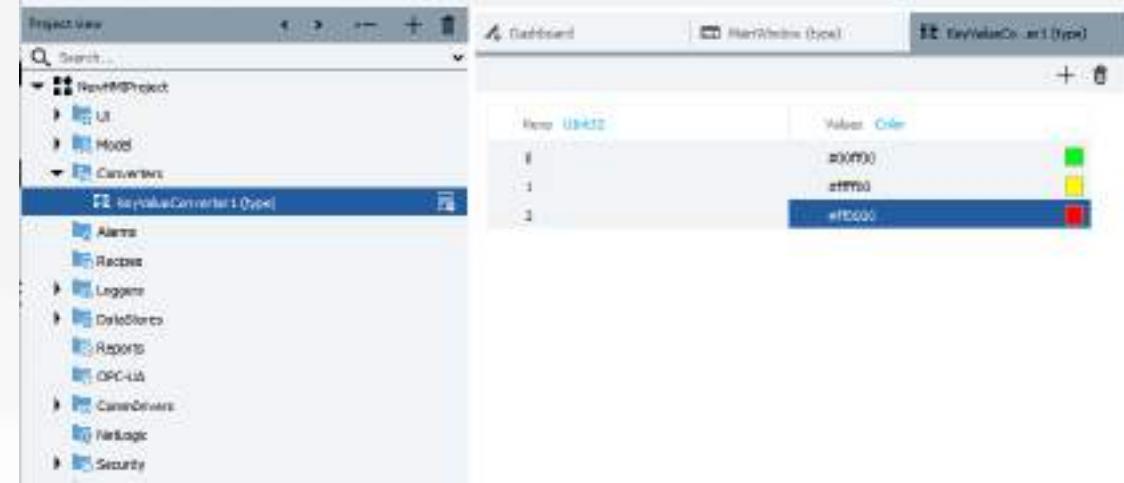
- Converters can be reused

- Using **copy/paste**

- Copy the converter from the property with "*Complex Dynamic Link*"
- Paste to another property

- Creating a **Converter as a Type**

- Create a Converter into Converters node
- The Converter will be automatically listed on the Advanced Dynamic Link configuration



# Advanced dynamic links: reusable converters

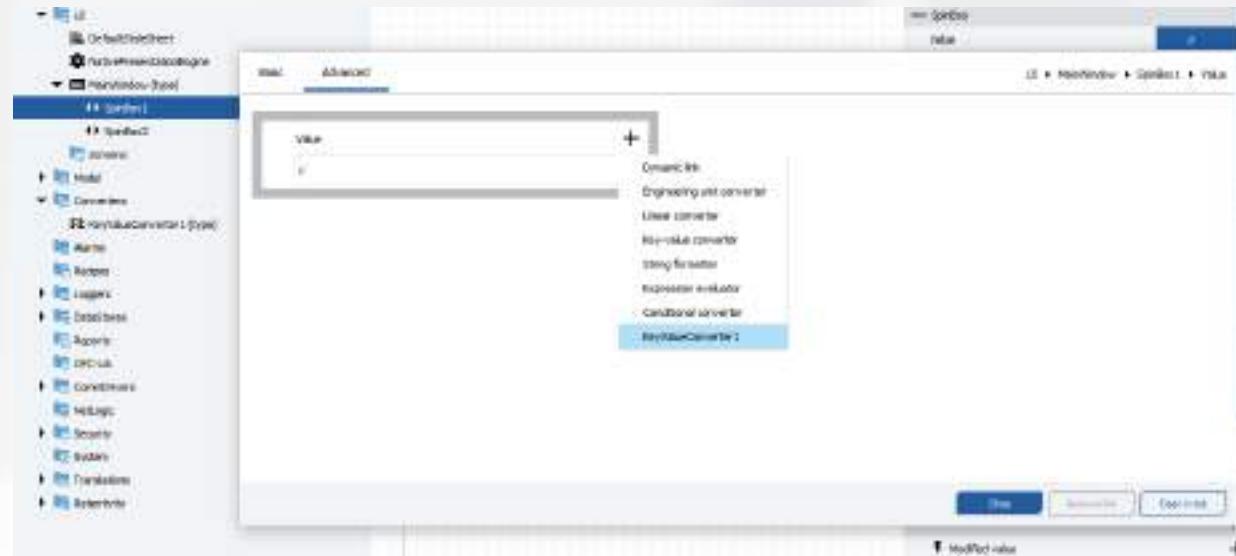
- Converters can be reused

- Using **copy/paste**

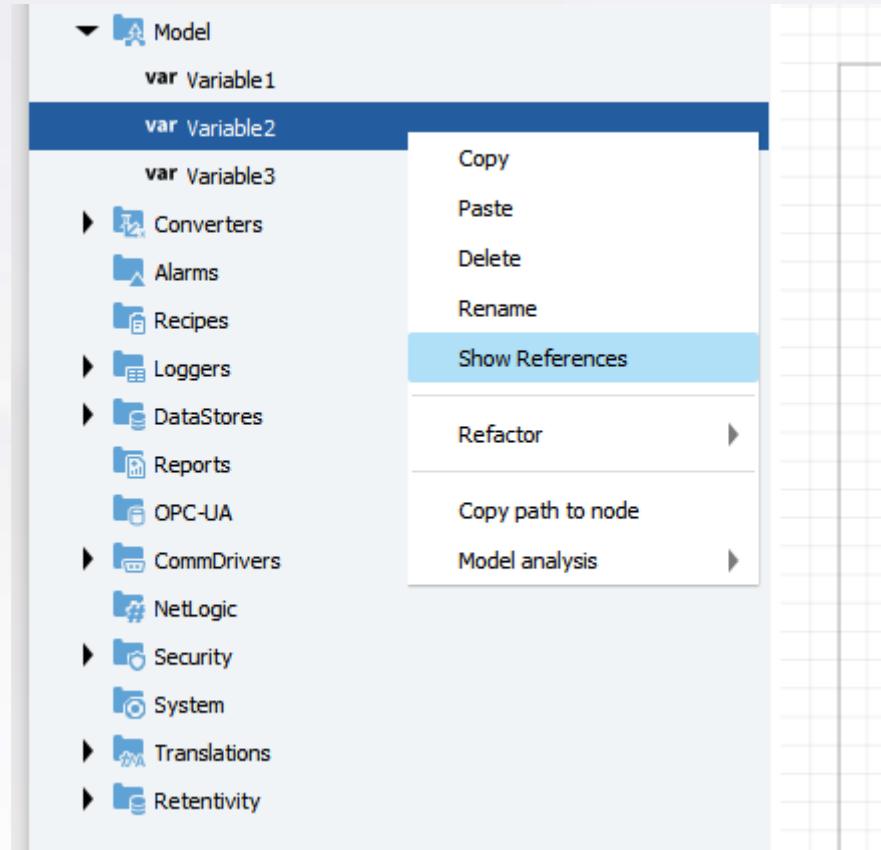
- Copy the converter from the property with "*Complex Dynamic Link*"
- Paste to another property

- Creating a **Converter as a Type**

- Create a Converter into Converters node
- The Converter will be automatically listed on the Advanced Dynamic Link configuration



# Show reference

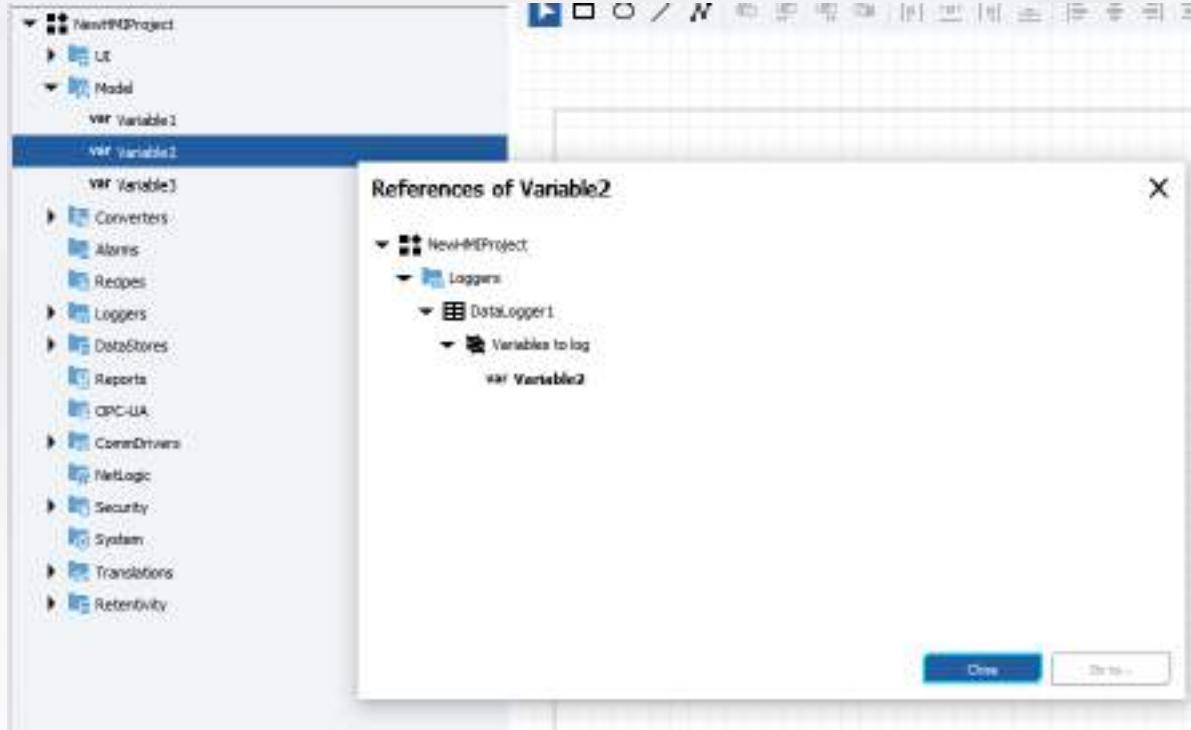


- Answer the question:  
**"who refers to this object ?"**

- Can be used mainly for Tags, but also, for other objects like
  - Datastore,
  - Scripts,
  - ...

# Show reference

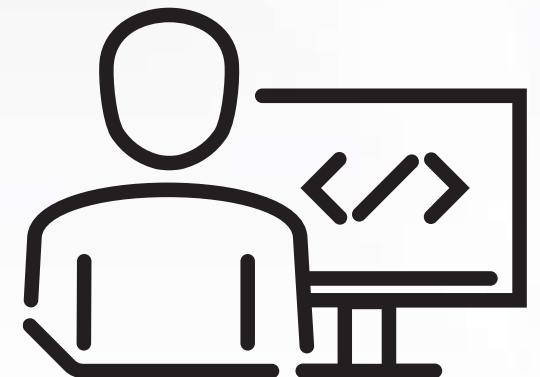
- Answer the question:  
**"who refers to this object ?"**



- Can be used mainly for Tags, but also, for other objects like
  - Datastore,
  - Scripts,
  - ...

# Hands-on session

- Define an "indexed dynamic link" to show the value of Variable1, Variable2, Variable3 using a SpinBox
- Use a Key-Value converter to change the background color of a rectangle based on the value of a variable
- Use a Key-Value converter to change the string of a label based on the value of a variable
- Use a Linear convert to show a scaled value



# Work with reusable objects

*Global objects, UDTs, Widgets*



# Object-oriented programming (OOP) paradigm

- Elements of OOP:

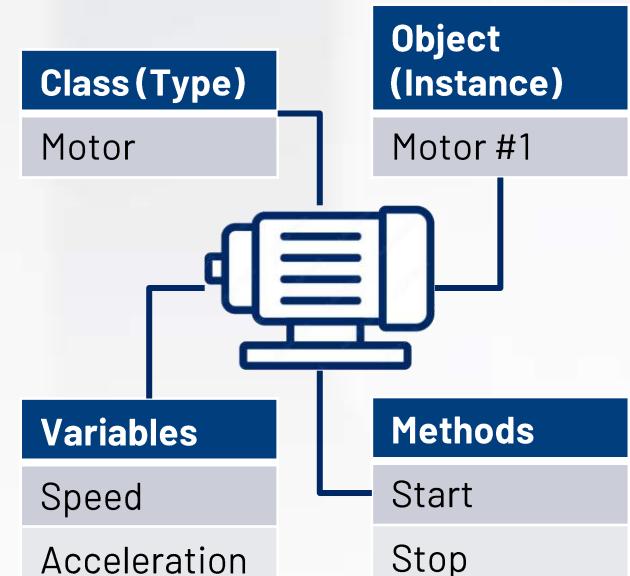
- **Class:** as a core element, it's a collection of methods and variables and generally, it is regarded as a **type for instance**.
- **Object:** as a basic element, it is regarded as an **instance** of a class

- Capabilities of OOP:

- Encapsulation, Abstraction, Inheritance, Polymorphism

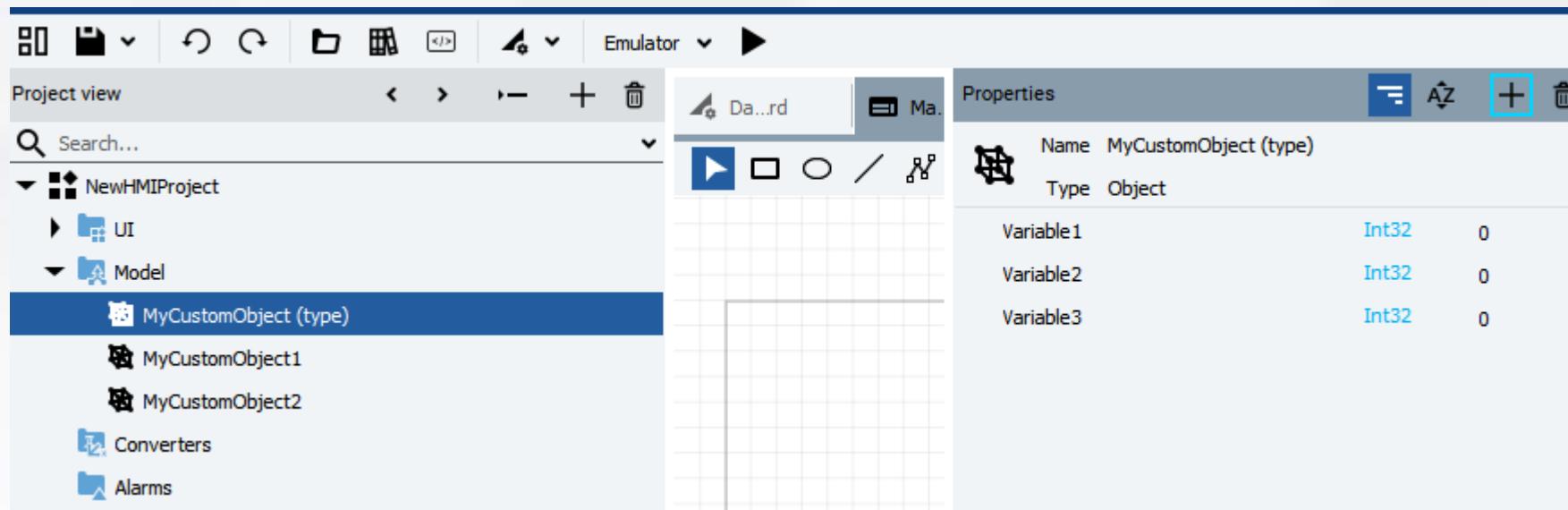
- Object-Oriented Programming in Optix means

- **Native Types** already seen →
- **Custom Types**



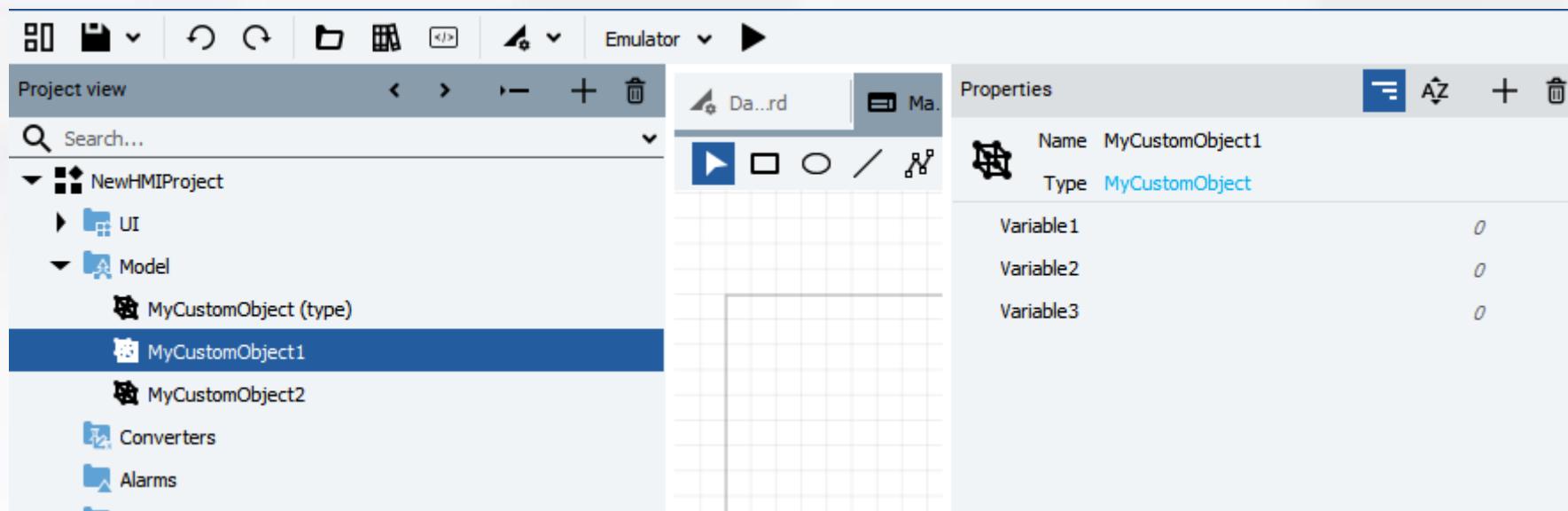
# Custom types: reusable tags

- Object-Oriented Programming paradigm can be applied to Model Tags
  - "MyMotor(type)" = Class = Type
  - "MyMotor1", "MyMotor2" = Instances



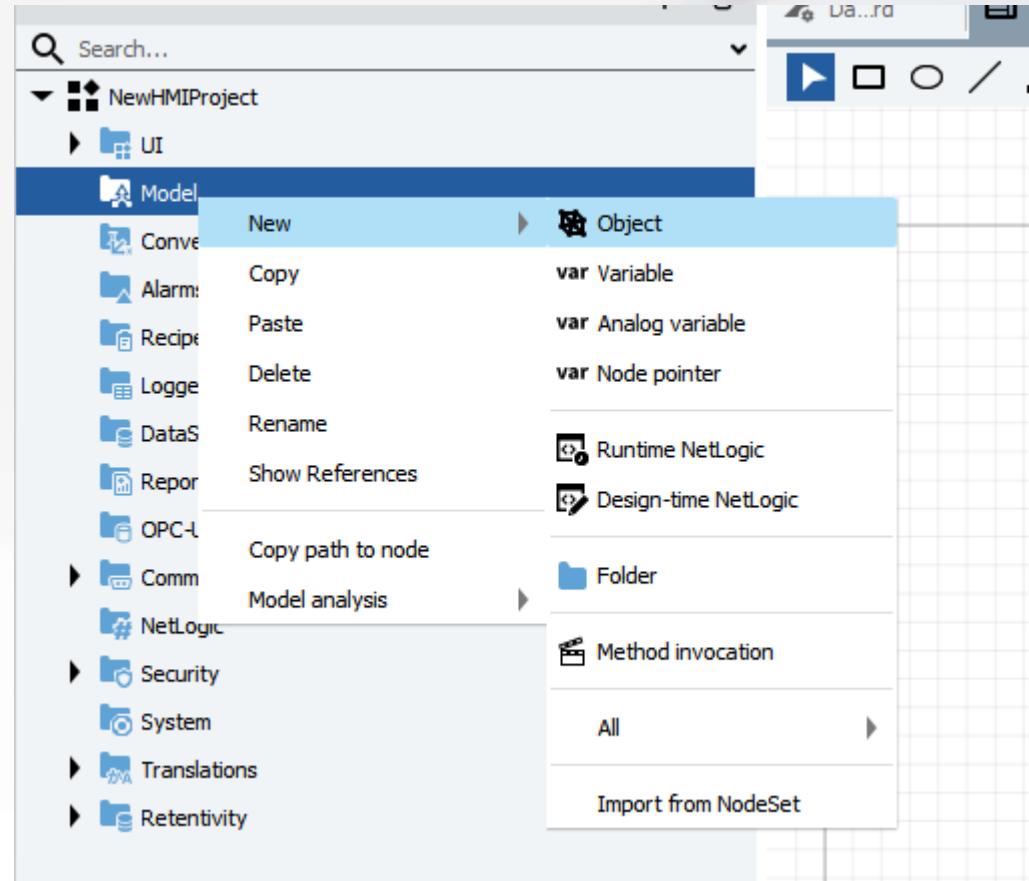
# Custom types: reusable tags

- Object-Oriented Programming paradigm can be applied to Model Tags
  - "MyMotor(type)" = Class = Type
  - "MyMotor1", "MyMotor2" = Instances



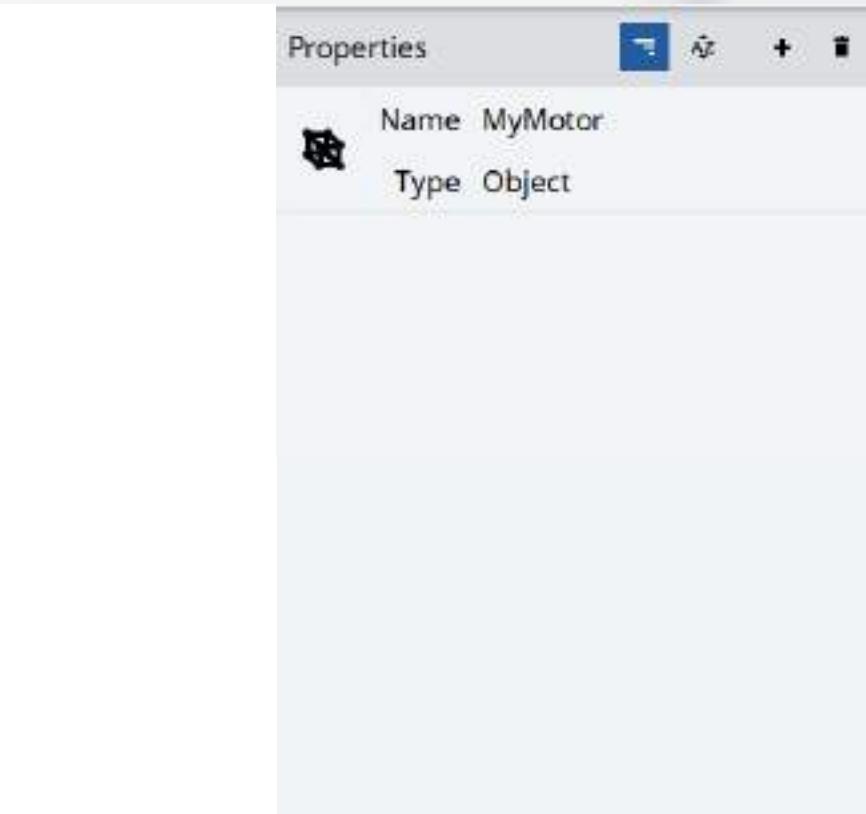
# Custom types: reusable tags example

- Example: a Structured Tag into Model folder
- Created as a brand **new object** that becomes a **new type**
- How to make your object a Type:
  - Option 1: Drag&Drop into Types View panel
  - Option 2: in "Advanced Mode",  
Right click > Refactor > Transform instance to type



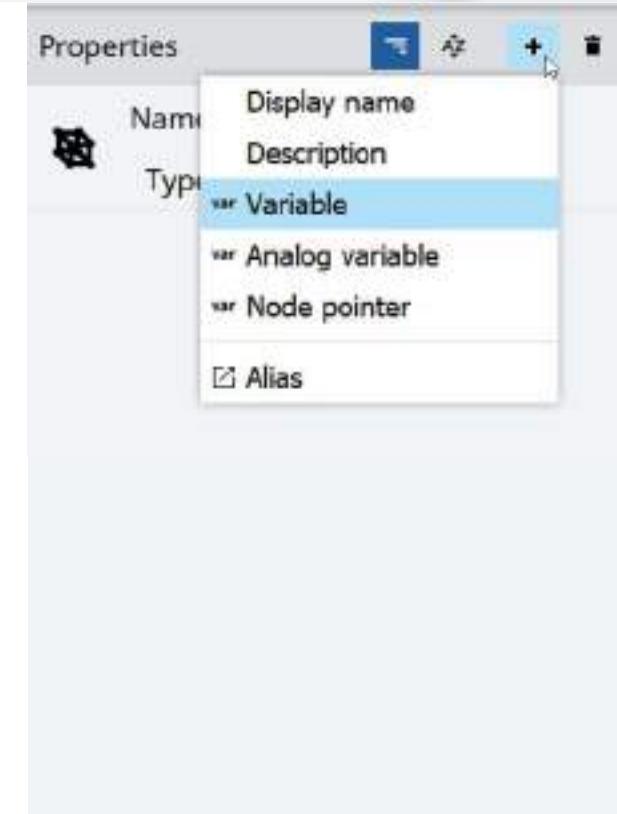
# Custom types: reusable tags example

- Example: a Structured Tag into Model folder
- Created as a brand **new object** that becomes a **new type**
- How to make your object a Type:
  - Option 1: Drag&Drop into Types View panel
  - Option 2: in "Advanced Mode", Right click > Refactor > Transform instance to type



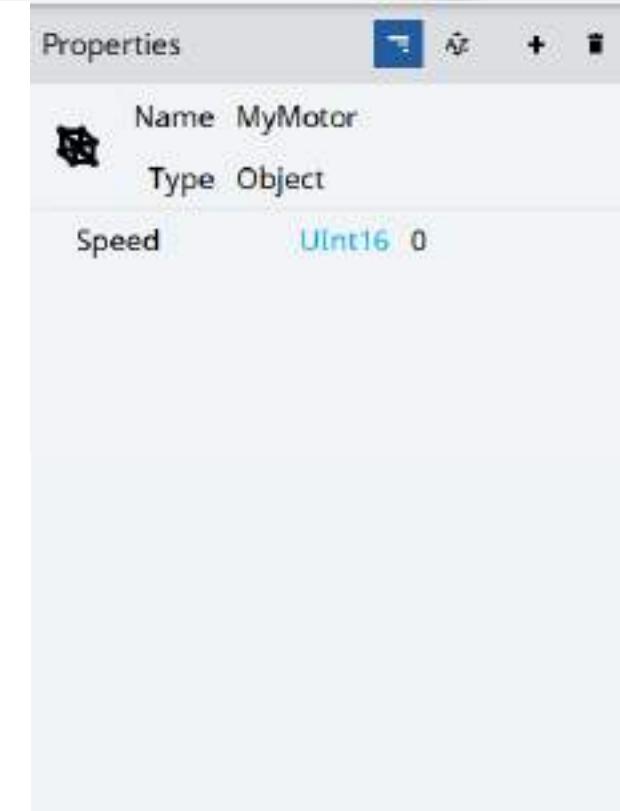
# Custom types: reusable tags example

- Example: a Structured Tag into Model folder
- Created as a brand **new object** that becomes a **new type**
- How to make your object a Type:
  - Option 1: Drag&Drop into Types View panel
  - Option 2: in "Advanced Mode", Right click > Refactor > Transform instance to type



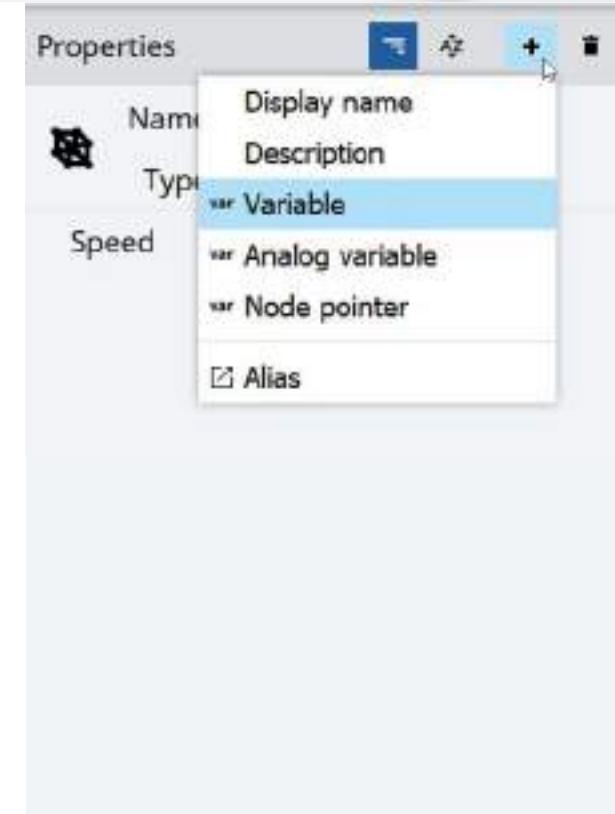
# Custom types: reusable tags example

- Example: a Structured Tag into Model folder
- Created as a brand **new object** that becomes a **new type**
- How to make your object a Type:
  - Option 1: Drag&Drop into Types View panel
  - Option 2: in "Advanced Mode", Right click > Refactor > Transform instance to type



# Custom types: reusable tags example

- Example: a Structured Tag into Model folder
- Created as a brand **new object** that becomes a **new type**
- How to make your object a Type:
  - Option 1: Drag&Drop into Types View panel
  - Option 2: in "Advanced Mode", Right click > Refactor > Transform instance to type



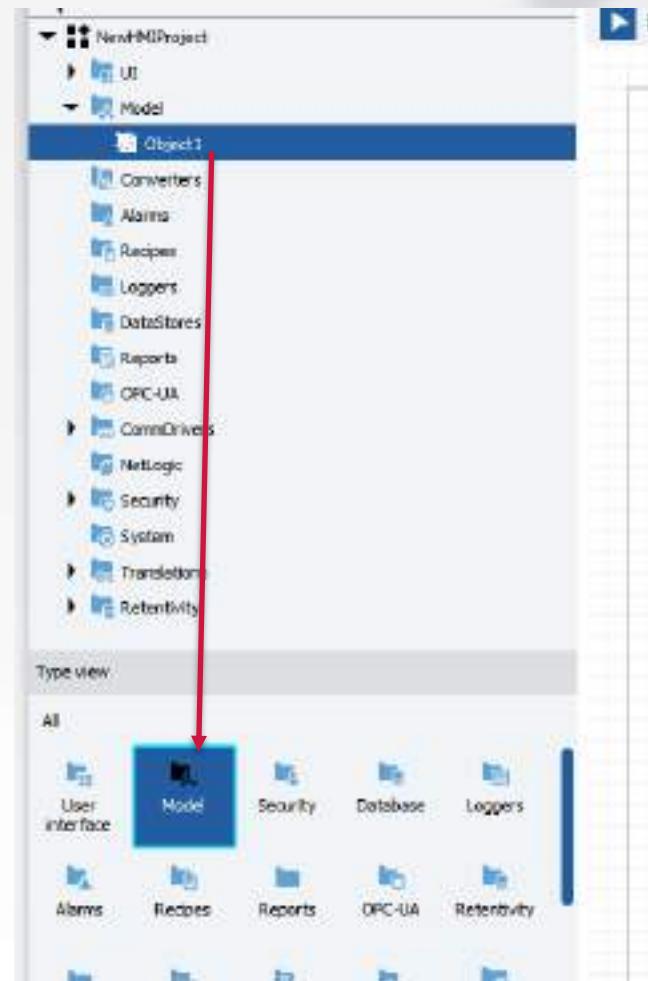
# Custom types: reusable tags example

- Example: a Structured Tag into Model folder
- Created as a brand **new object** that becomes a **new type**
- How to make your object a Type:
  - Option 1: Drag&Drop into Types View panel
  - Option 2: in "Advanced Mode", Right click > Refactor > Transform instance to type



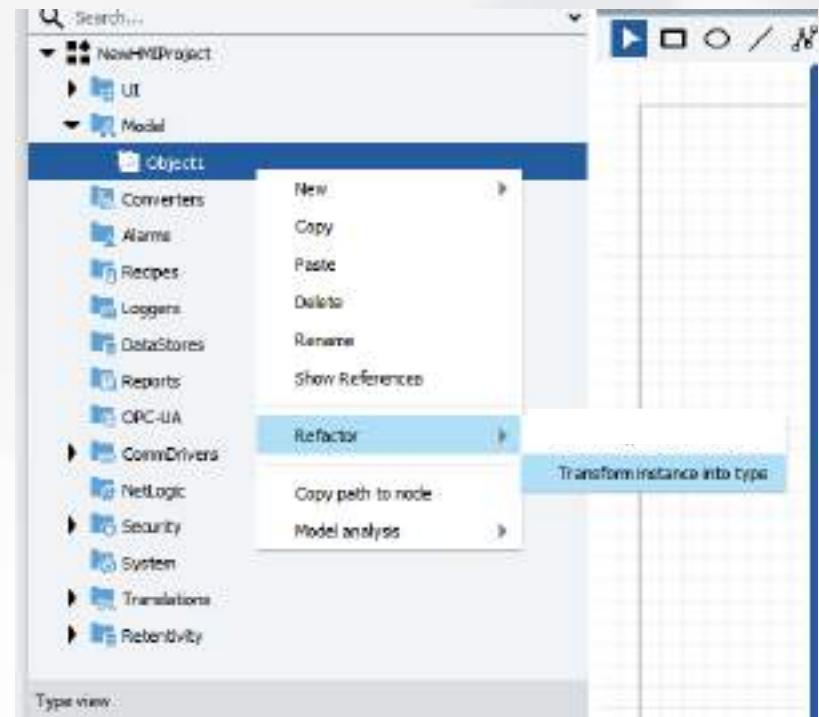
# Custom types: reusable tags example

- Example: a Structured Tag into Model folder
- Created as a brand **new object** that becomes a **new type**
- How to make your object a Type:
  - Option 1: Drag&Drop into Types View panel
  - Option 2: in "Advanced Mode", Right click > Refactor > Transform instance to type



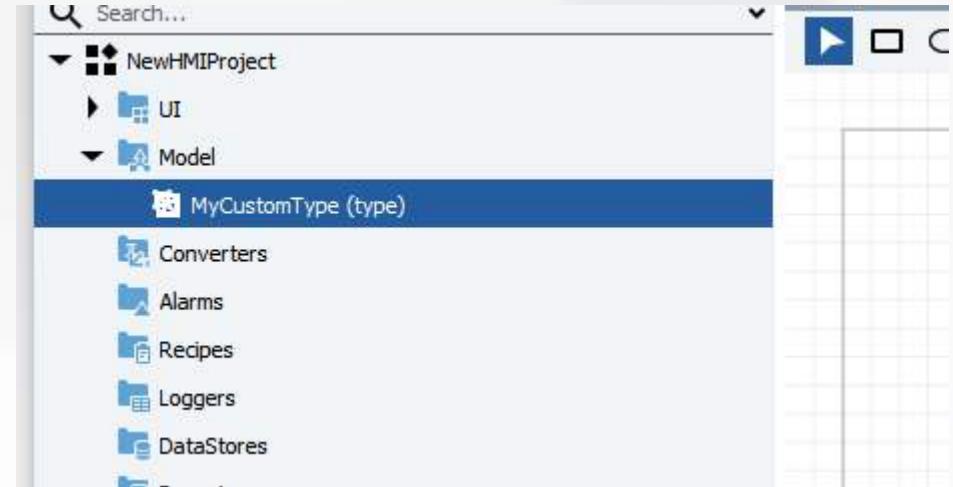
# Custom types: reusable tags example

- Example: a Structured Tag into Model folder
- Created as a brand **new object** that becomes a **new type**
- How to make your object a Type:
  - Option 1: Drag&Drop into Types View panel
  - Option 2: in "Advanced Mode", Right click > Refactor > Transform instance to type



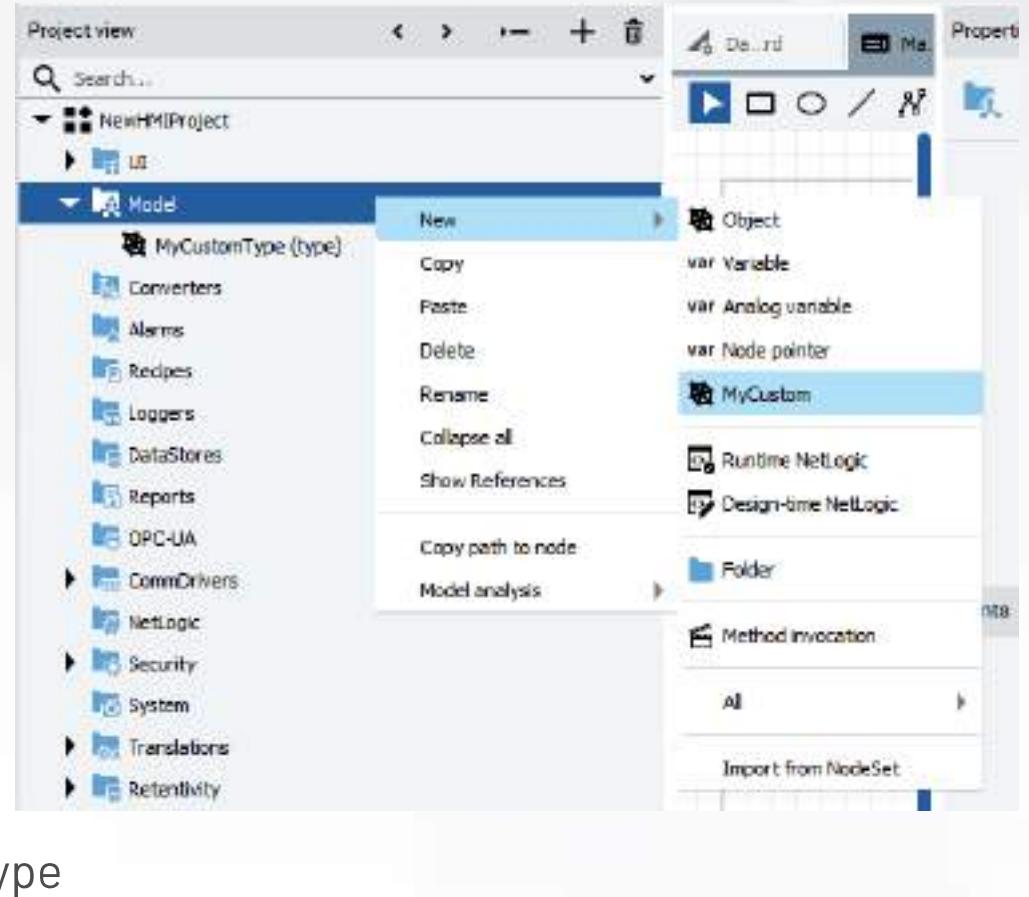
# Custom types: reusable tags example

- Example: a Structured Tag into Model folder
- Created as a brand **new object** that becomes a **new type**
- How to make your object a Type:
  - Option 1: Drag&Drop into Types View panel
  - Option 2: in "Advanced Mode",  
Right click > Refactor > Transform instance to type



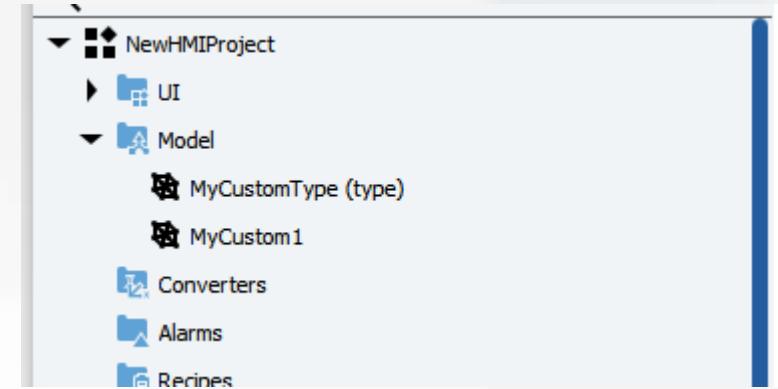
# Custom types: reusable tags example

- Example: a Structured Tag into Model folder
- Created as a brand **new object** that becomes a **new type**
- How to make your object a Type:
  - Option 1: Drag&Drop into Types View panel
  - Option 2: in "Advanced Mode",  
Right click > Refactor > Transform instance to type



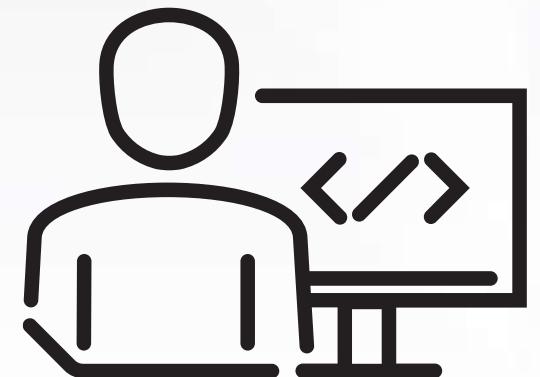
# Custom types: reusable tags example

- Example: a Structured Tag into Model folder
- Created as a brand **new object** that becomes a **new type**
- How to make your object a Type:
  - Option 1: Drag&Drop into Types View panel
  - Option 2: in "Advanced Mode",  
Right click > Refactor > Transform instance to type



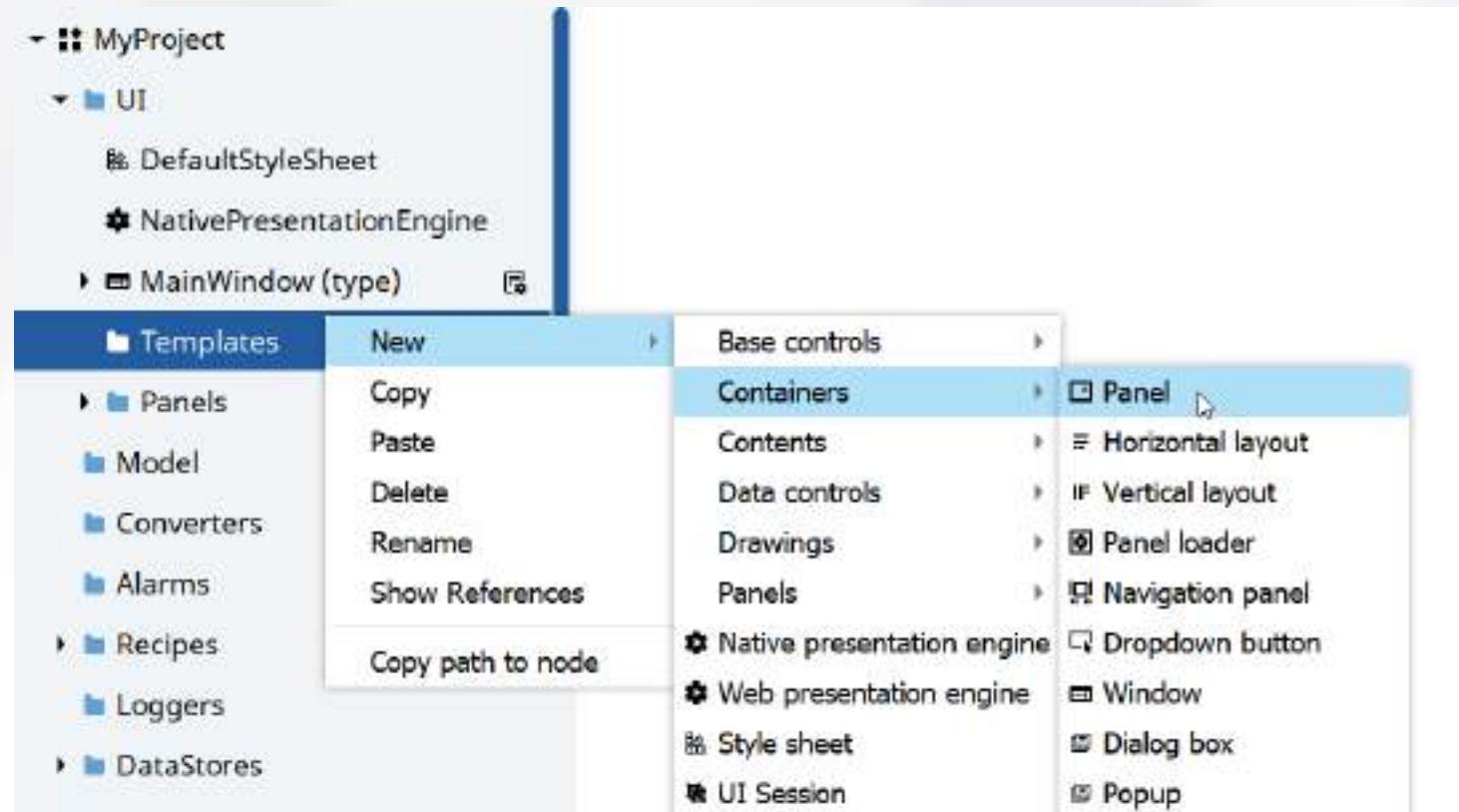
# Hands-on session

- Create a Motor Type variable into the Model folder with these elements
  - Power (Float)
  - Speed (Int16)
- Create 3 instances of the Motor Type
  - Motor1, Motor2, Motor3...



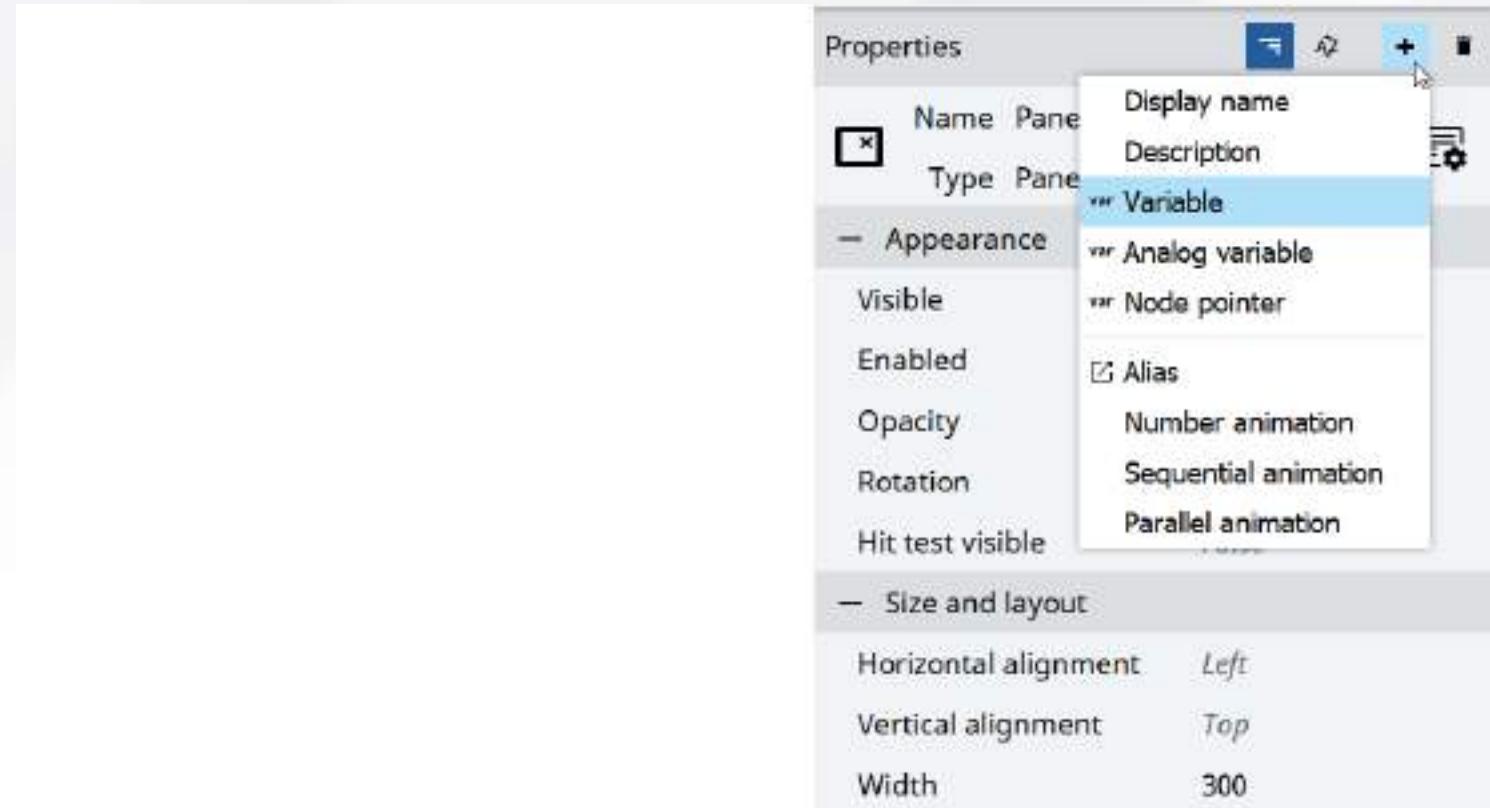
# Custom types: reusable graphics with a "local variable"

- Object-Oriented Programming paradigm can be also applied to Graphics
  - "Panel\_with\_BG(type)" = Class = Type
  - "Panel1" = Instance
- Reusable Graphics can be
  - Widgets
  - Dialogs
  - Panels
- Example: a customized Panel with a rectangle used to set the background color



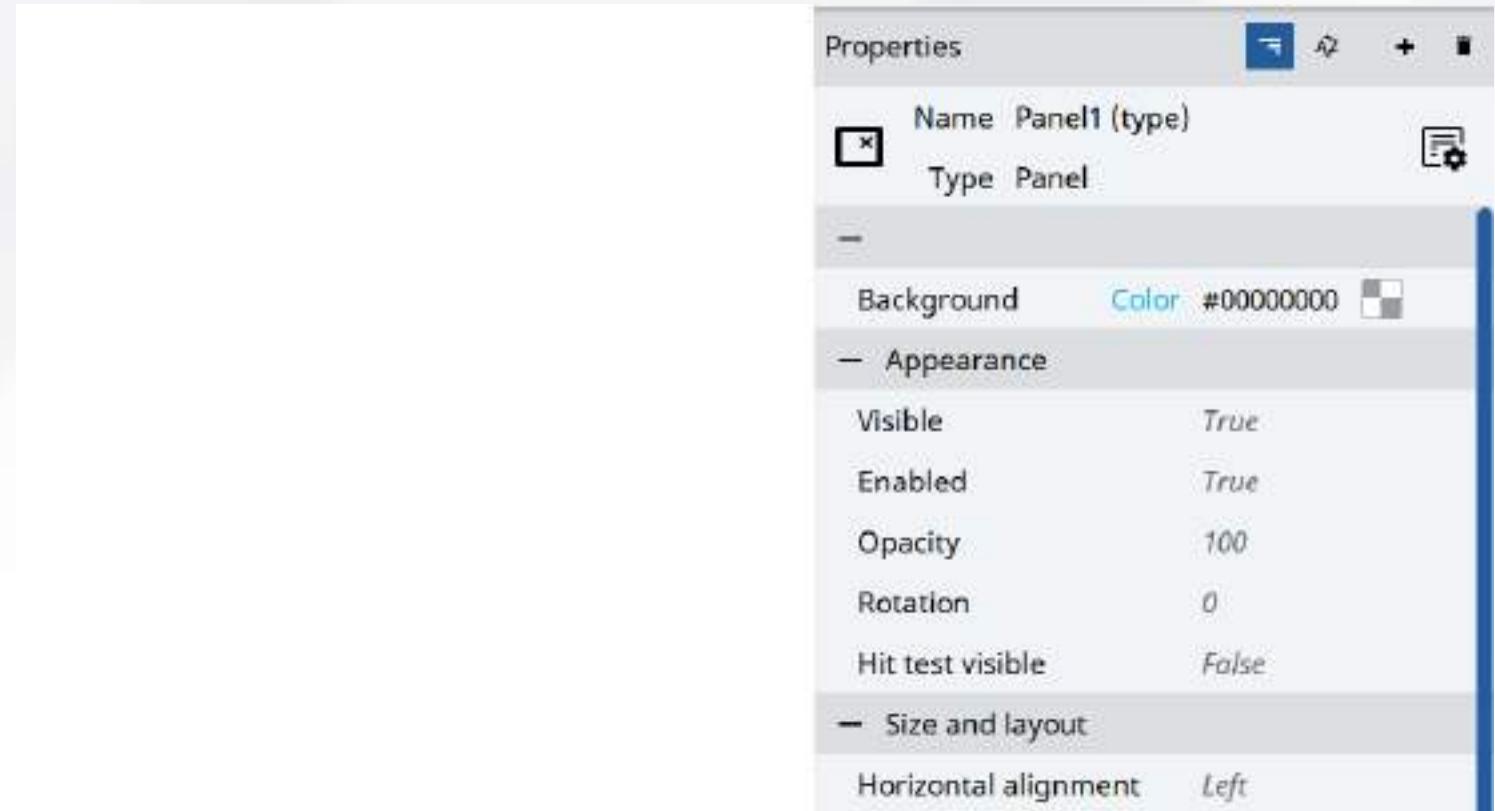
# Custom types: reusable graphics with a "local variable"

- Object-Oriented Programming paradigm can be also applied to Graphics
  - "Panel\_with\_BG(type)" = Class = Type
  - "Panel1" = Instance
- Reusable Graphics can be
  - Widgets
  - Dialogs
  - Panels
- Example: a customized Panel with a rectangle used to set the background color



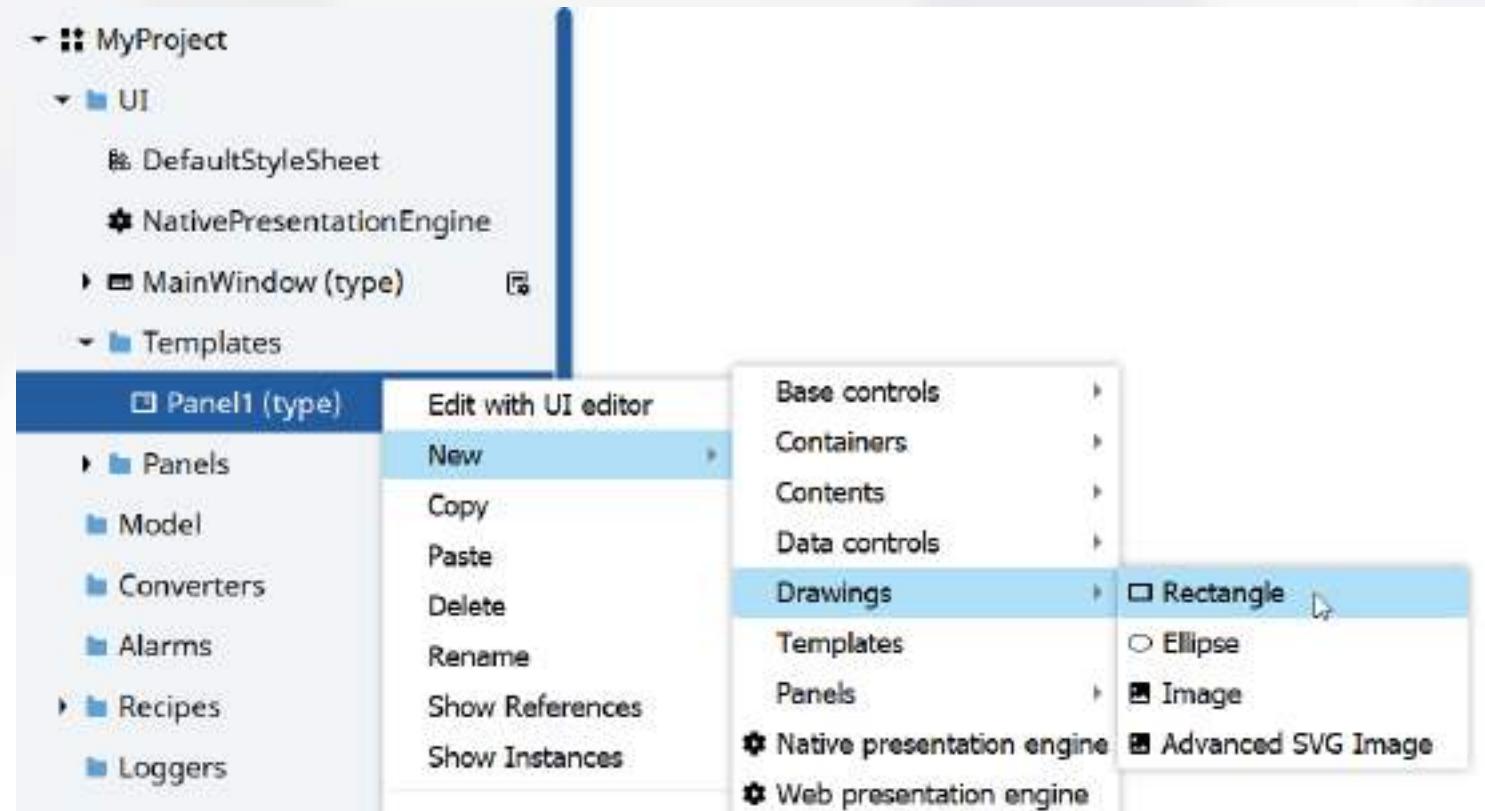
# Custom types: reusable graphics with a "local variable"

- Object-Oriented Programming paradigm can be also applied to Graphics
  - "Panel\_with\_BG(type)" = Class = Type
  - "Panel1" = Instance
- Reusable Graphics can be
  - Widgets
  - Dialogs
  - Panels
- Example: a customized Panel with a rectangle used to set the background color



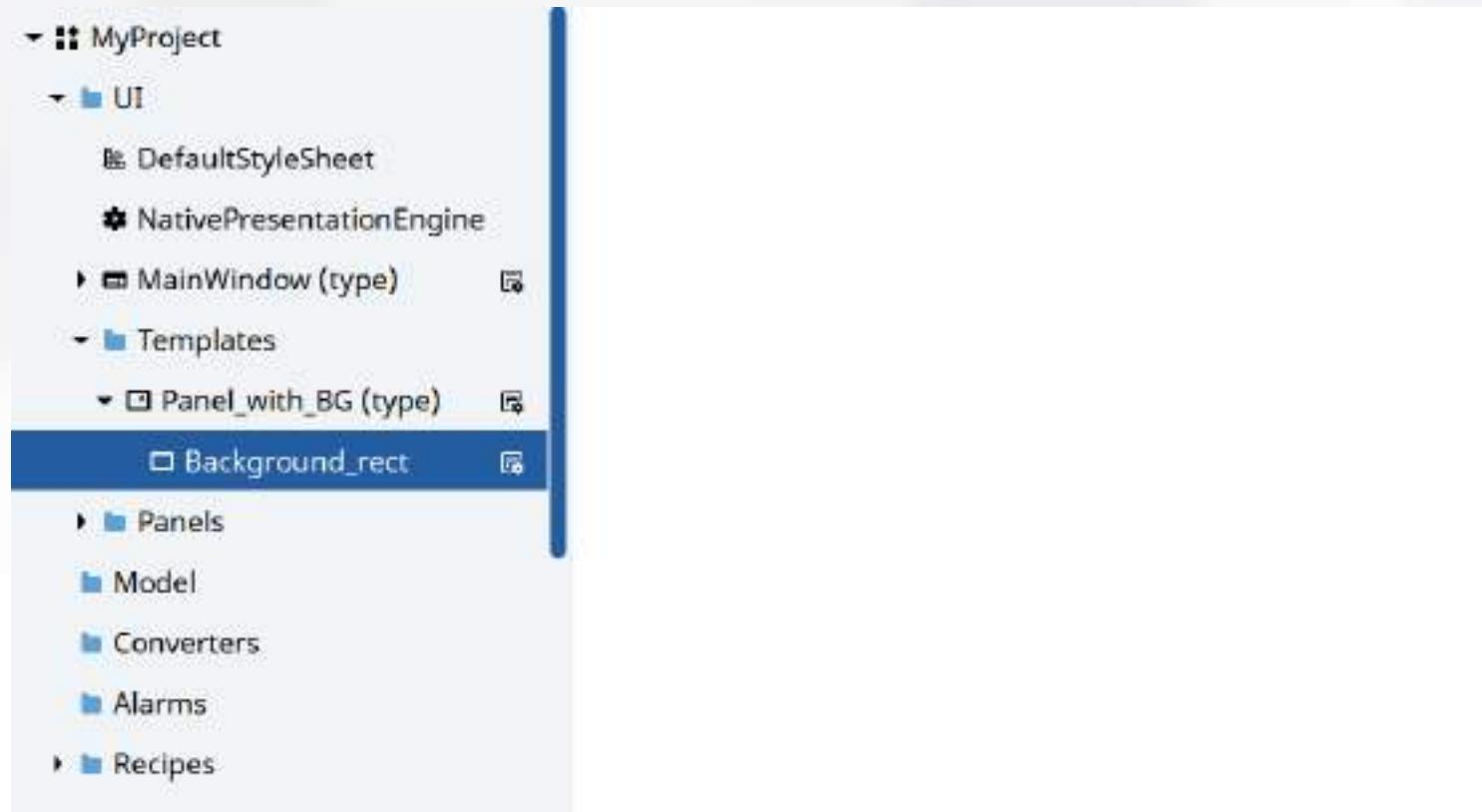
# Custom types: reusable graphics with a "local variable"

- Object-Oriented Programming paradigm can be also applied to Graphics
  - "Panel\_with\_BG(type)" = Class = Type
  - "Panel1" = Instance
- Reusable Graphics can be
  - Widgets
  - Dialogs
  - Panels
- Example: a customized Panel with a rectangle used to set the background color



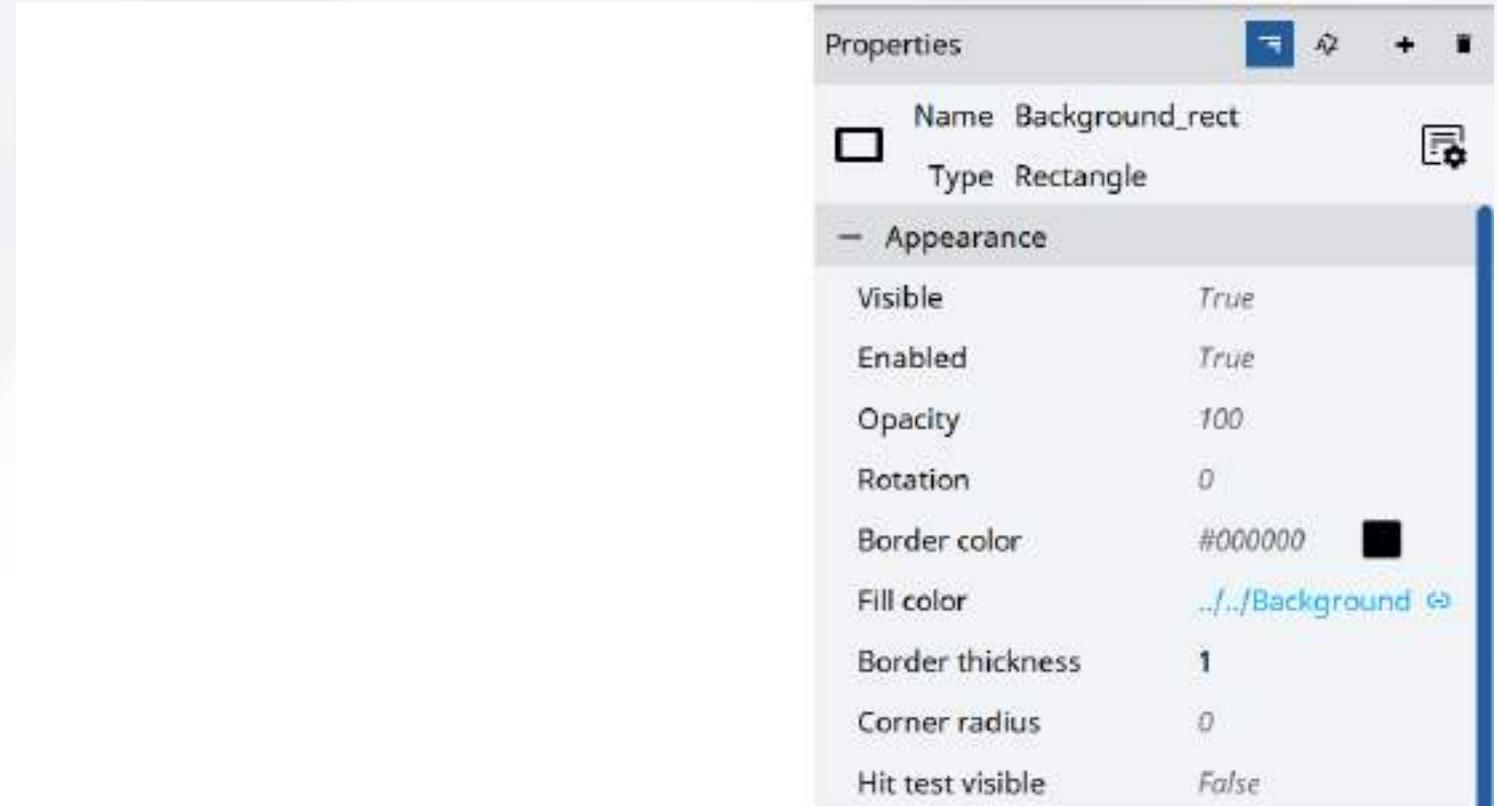
# Custom types: reusable graphics with a "local variable"

- Object-Oriented Programming paradigm can be also applied to Graphics
  - "Panel\_with\_BG(type)" = Class = Type
  - "Panel1" = Instance
- Reusable Graphics can be
  - Widgets
  - Dialogs
  - Panels
- Example: a customized Panel with a rectangle used to set the background color



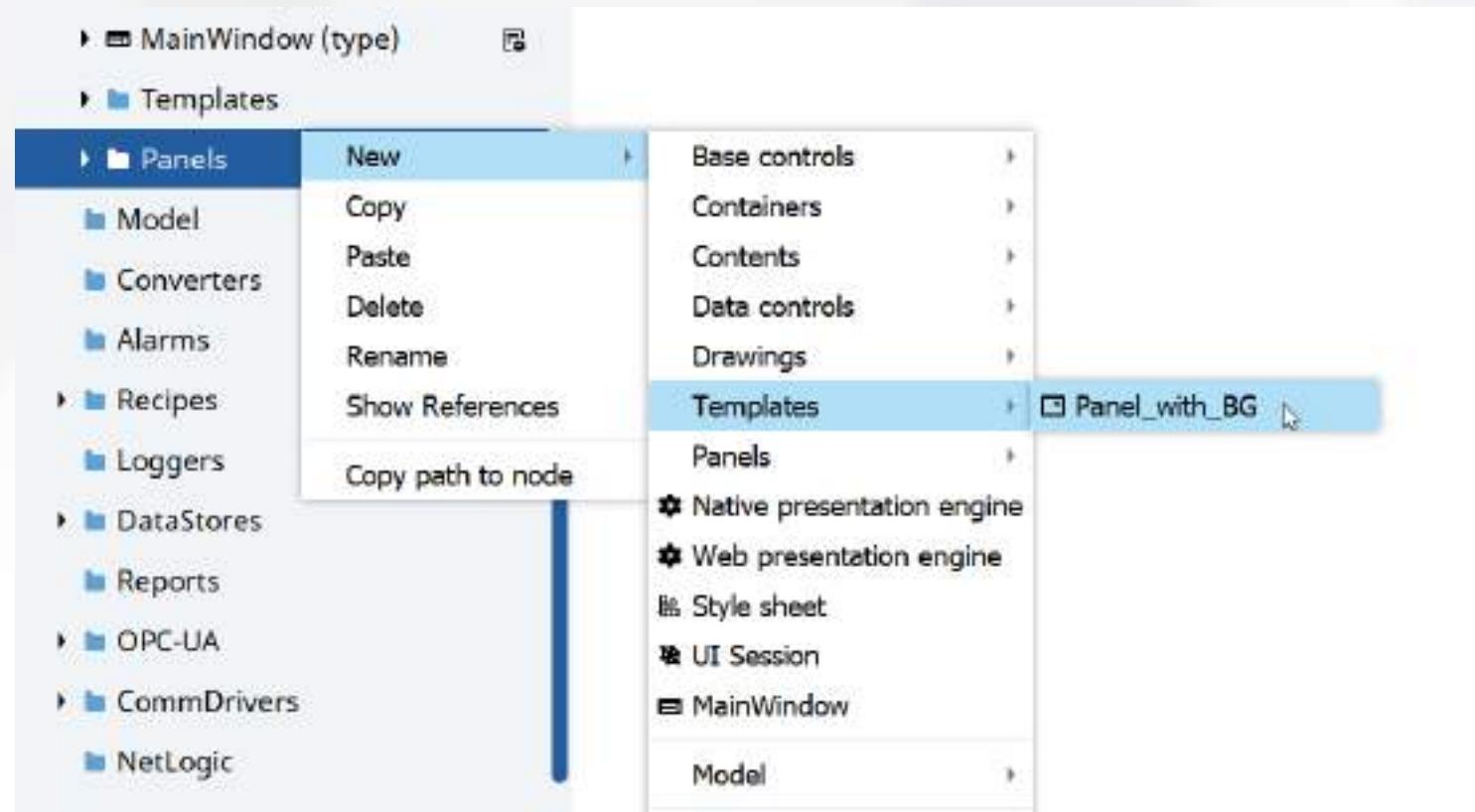
# Custom types: reusable graphics with a "local variable"

- Object-Oriented Programming paradigm can be also applied to Graphics
  - "Panel\_with\_BG(type)" = Class = Type
  - "Panel1" = Instance
- Reusable Graphics can be
  - Widgets
  - Dialogs
  - Panels
- Example: a customized Panel with a rectangle used to set the background color



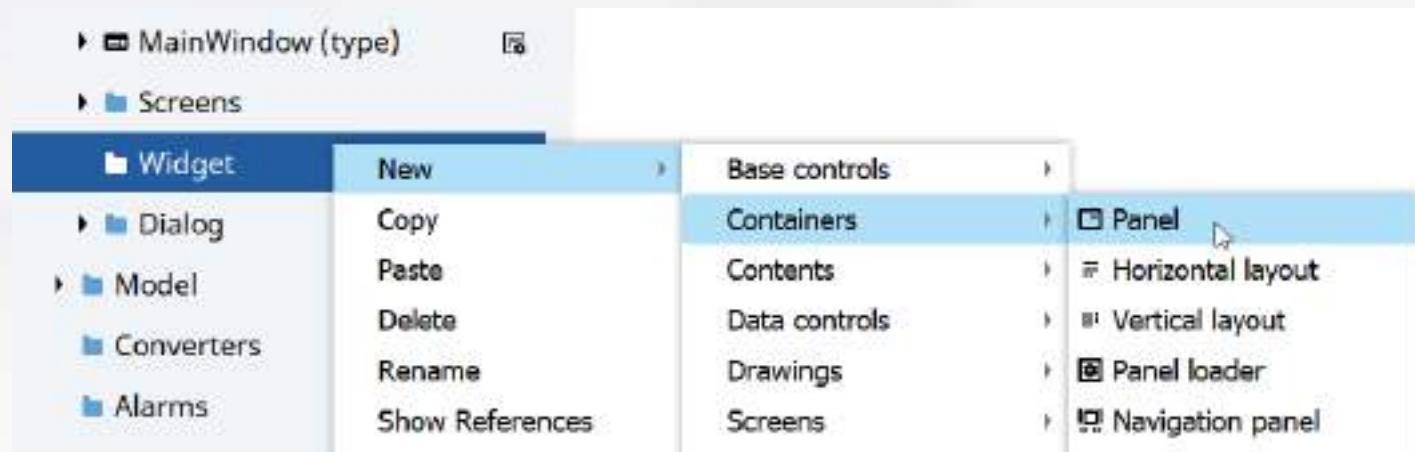
# Custom types: reusable graphics with a "local variable"

- Object-Oriented Programming paradigm can be also applied to Graphics
  - "Panel\_with\_BG(type)" = Class = Type
  - "Panel1" = Instance
- Reusable Graphics can be
  - Widgets
  - Dialogs
  - Panels
- Example: a customized Panel with a rectangle used to set the background color



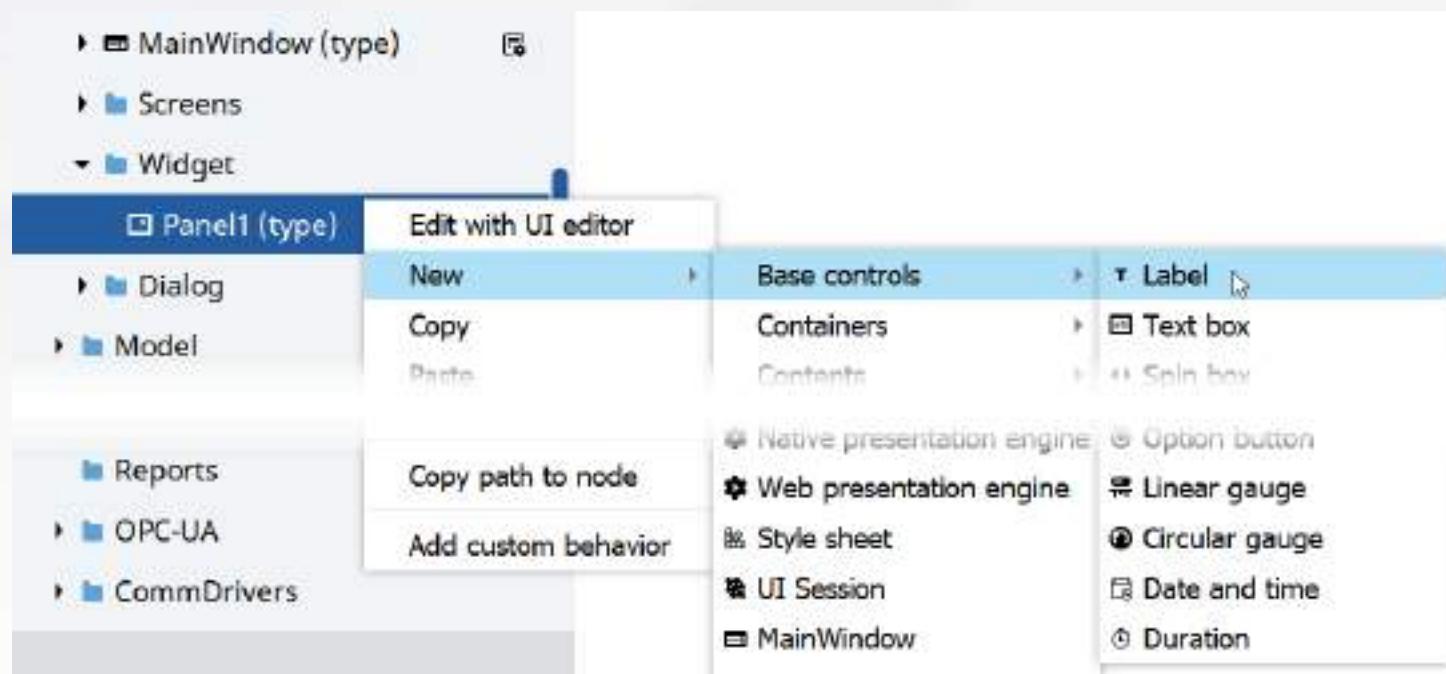
# Custom types: reusable graphics with an alias

- Example: a motor widget that shows the values of different instances of the motor



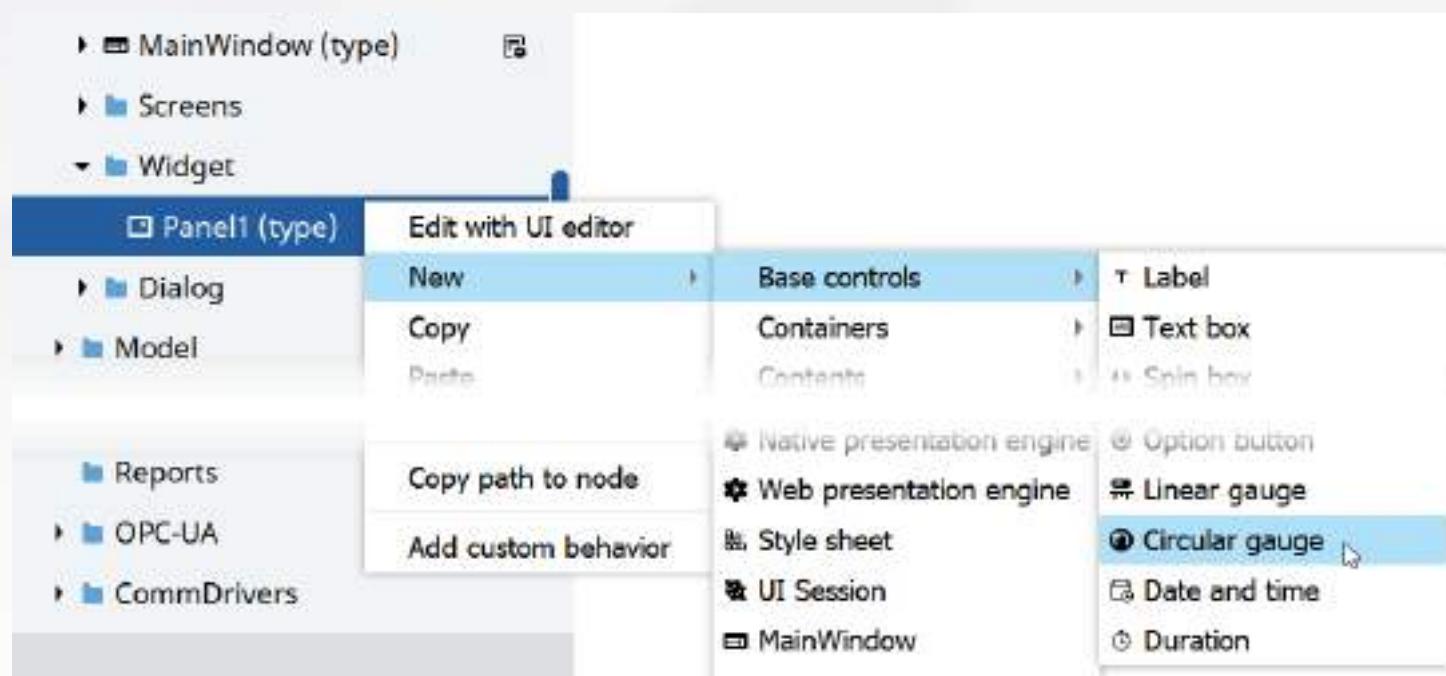
# Custom types: reusable graphics with an alias

- Example: a motor widget that shows the values of different instances of the motor



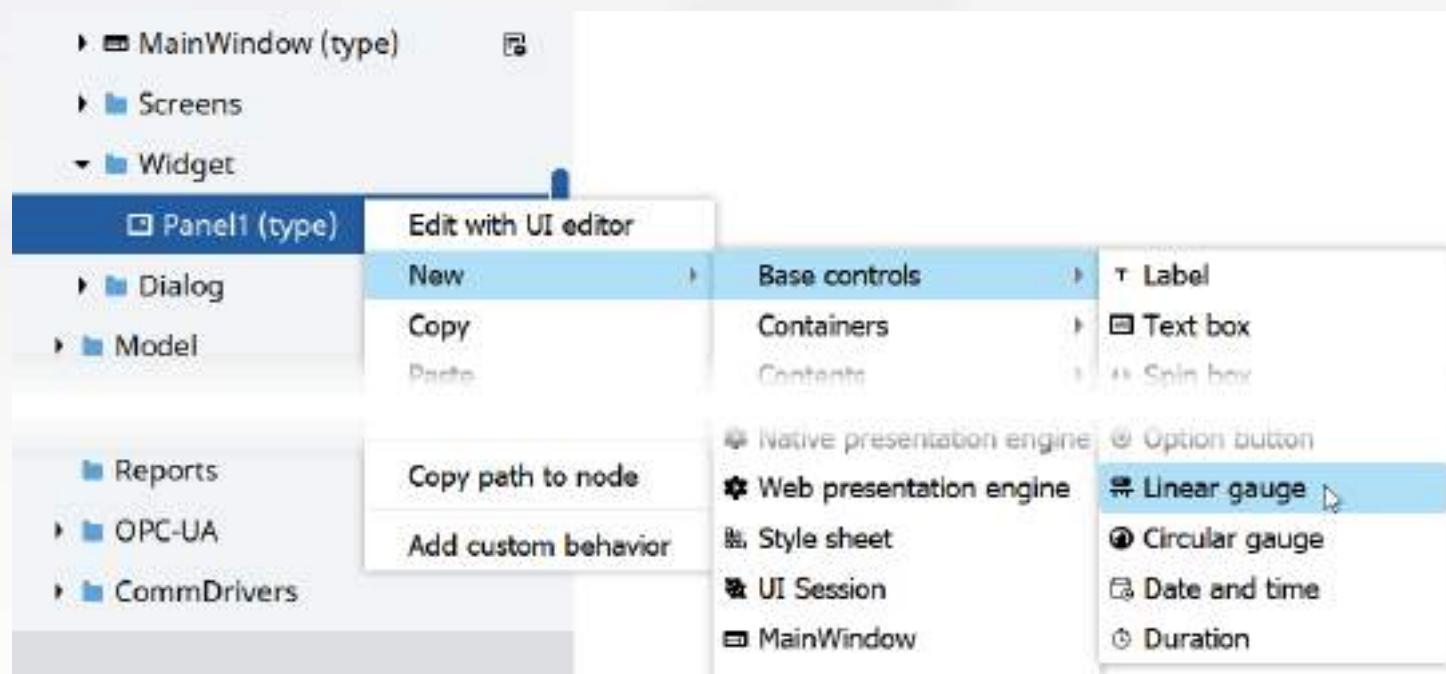
# Custom types: reusable graphics with an alias

- Example: a motor widget that shows the values of different instances of the motor



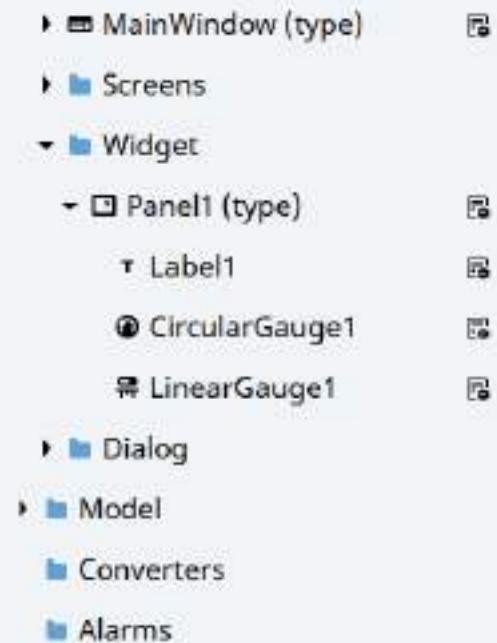
# Custom types: reusable graphics with an alias

- Example: a motor widget that shows the values of different instances of the motor



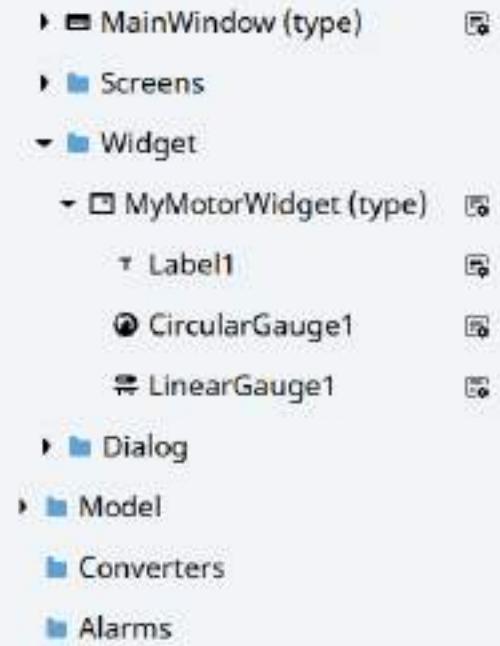
# Custom types: reusable graphics with an alias

- Example: a motor widget that shows the values of different instances of the motor



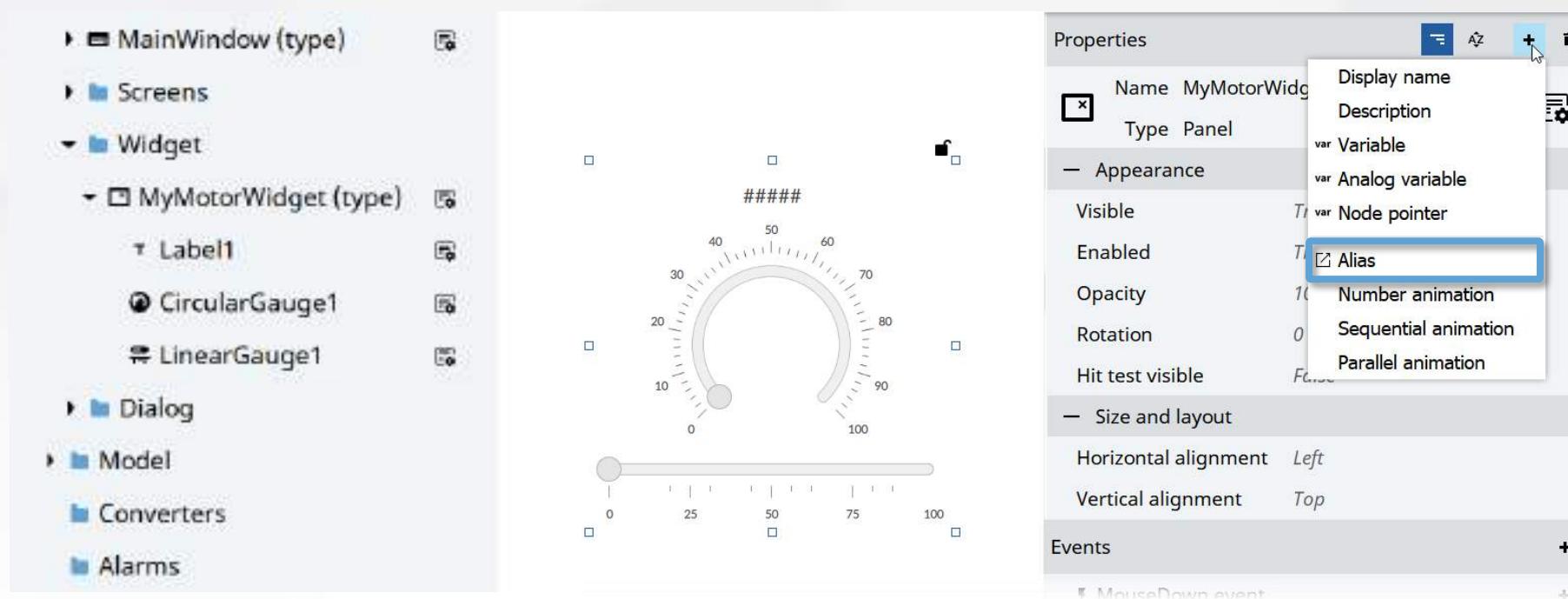
# Custom types: reusable graphics with an alias

- Example: a motor widget that shows the values of different instances of the motor



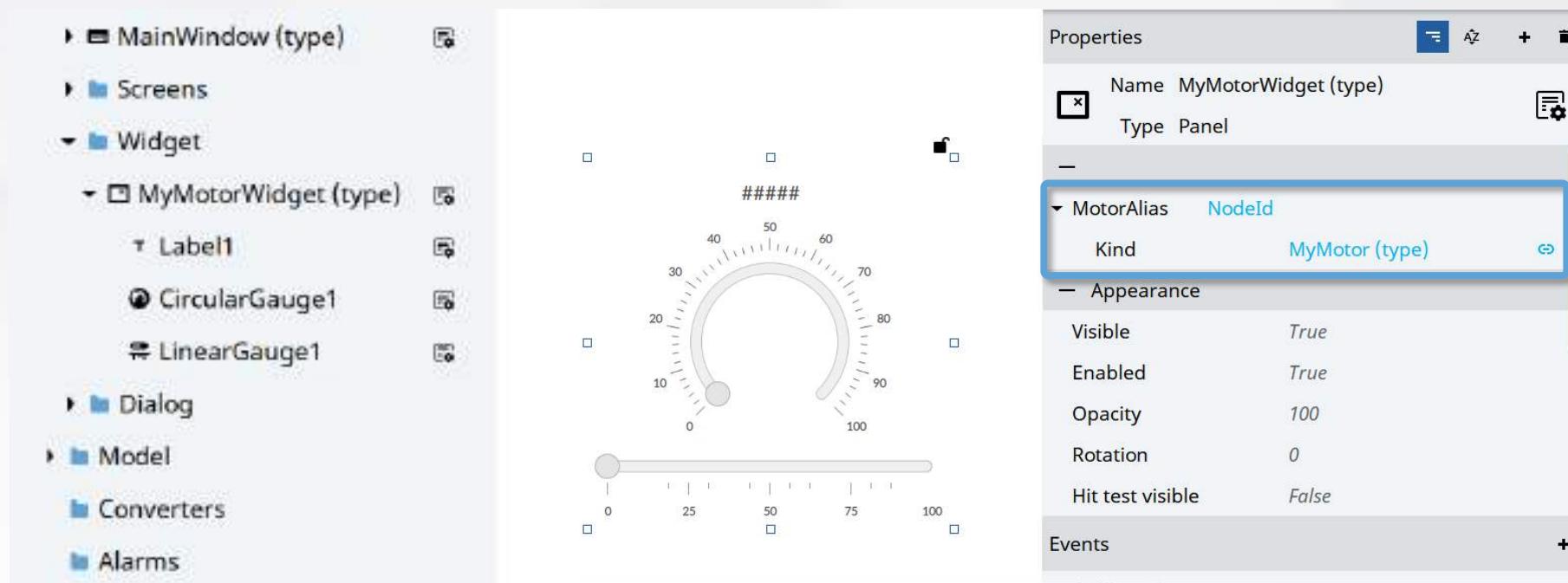
# Custom types: reusable graphics with an alias

- Example: a motor widget that shows the values of different instances of the motor



# Custom types: reusable graphics with an alias

- Example: a motor widget that shows the values of different instances of the motor



# Custom types: reusable graphics with an alias

- Example: a motor widget that shows the values of different instances of the motor

The screenshot displays a software interface for creating graphical user interfaces (GUIs). On the left, there is a tree view of project components:

- MainWindow (type)
- Screens
- Widget
  - MyMotorWidget (type)
    - Label1
    - CircularGauge1
    - LinearGauge1
- Dialog
- Model
- Converters
- Alarms

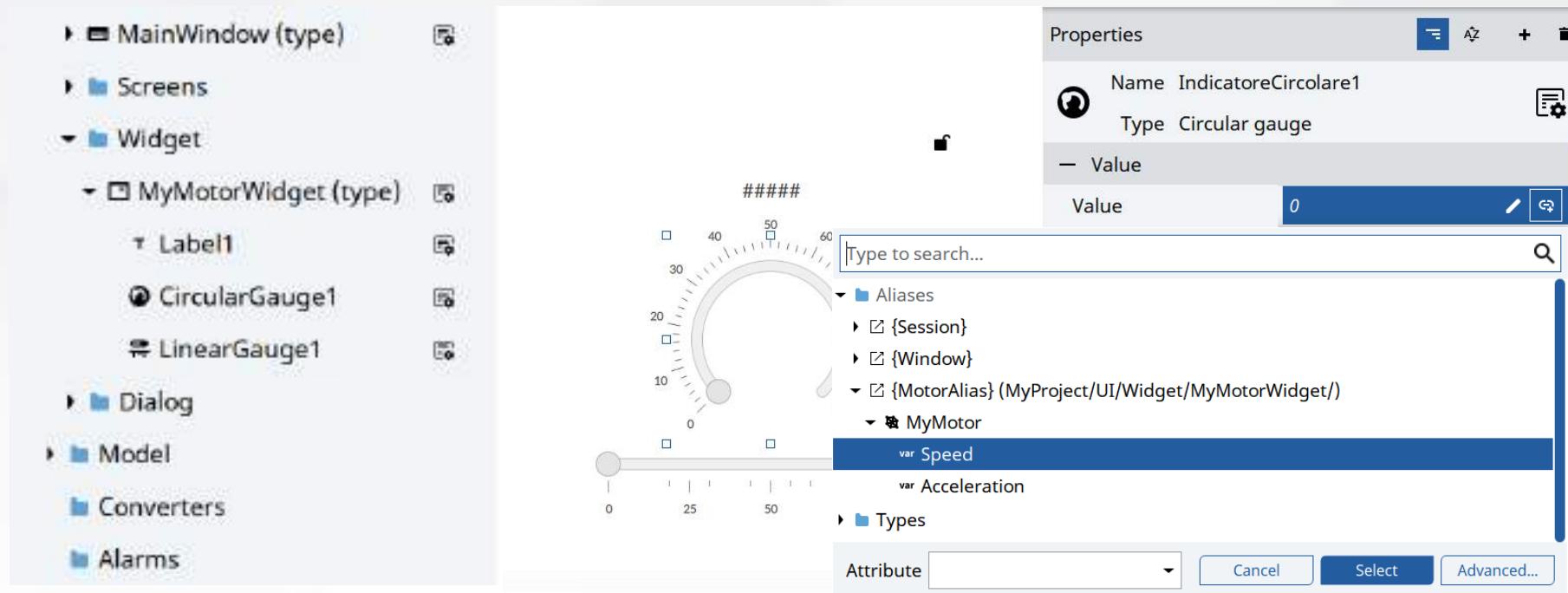
In the center, there is a preview window showing a circular gauge with a scale from 0 to 100. The gauge has major tick marks at 0, 25, 50, 75, and 100, and minor tick marks every 10 units. The value is currently set to 0. Below the preview is a code snippet placeholder: #####.

On the right, the Properties panel is open for the "CircularGauge1" component:

Properties	
Name	IndicatoreCircolare1
Type	Circular gauge
Value	
Value	0
Minimum value	0
Maximum value	100
Value change behavior ...ue change on edit finished	
Gauge	
Start angle	-135
End angle	135
Editable	True
Events	
ModifiedValueEvent	+

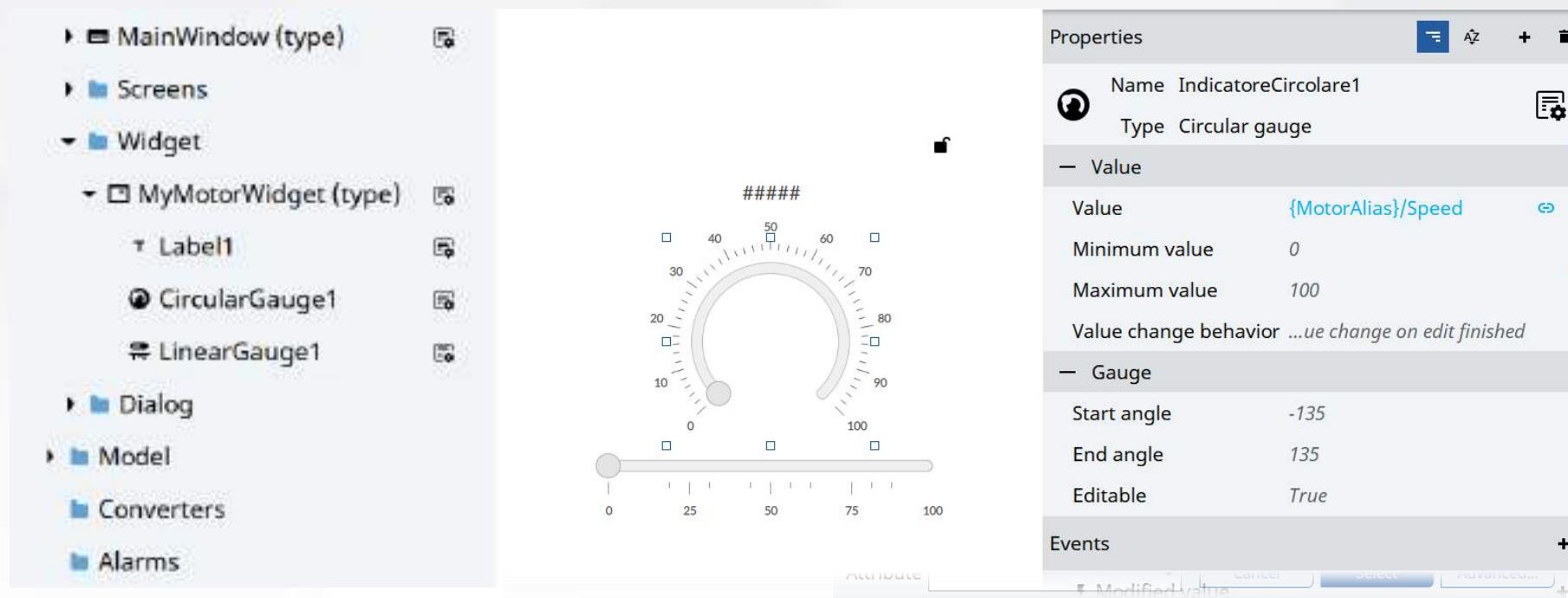
# Custom types: reusable graphics with an alias

- Example: a motor widget that shows the values of different instances of the motor



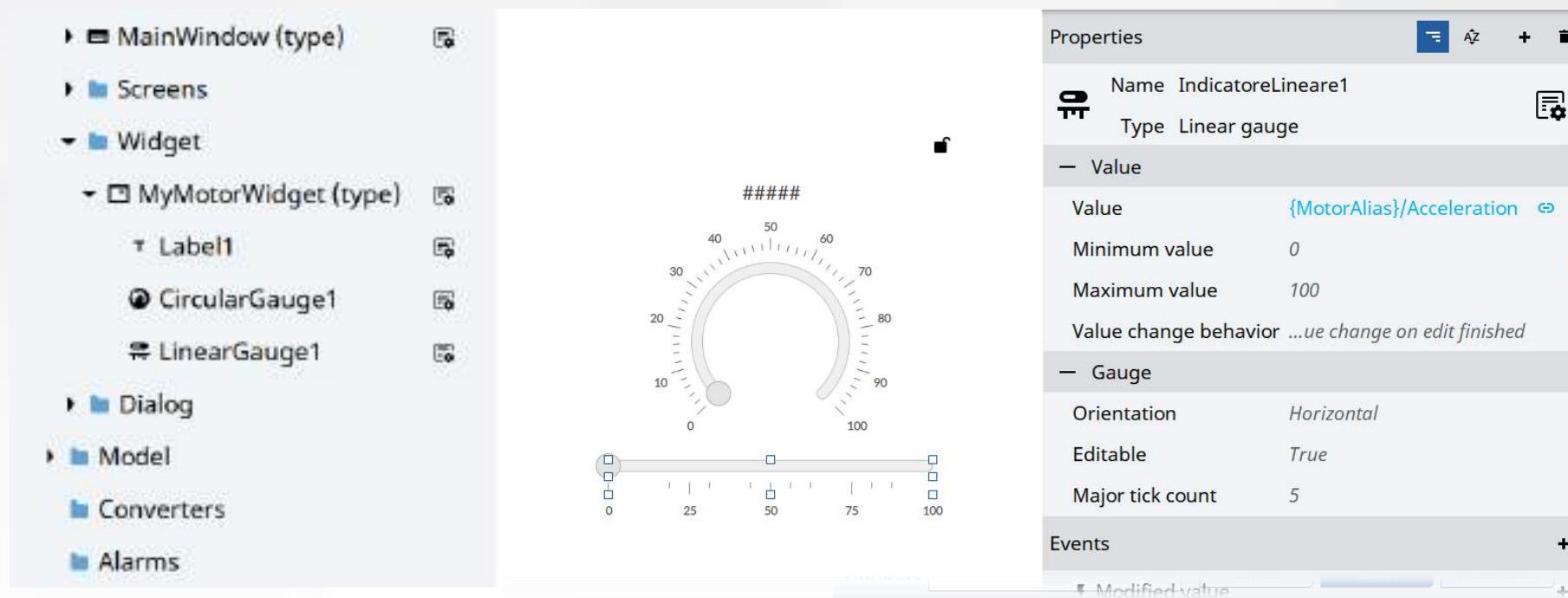
# Custom types: reusable graphics with an alias

- Example: a motor widget that shows the values of different instances of the motor



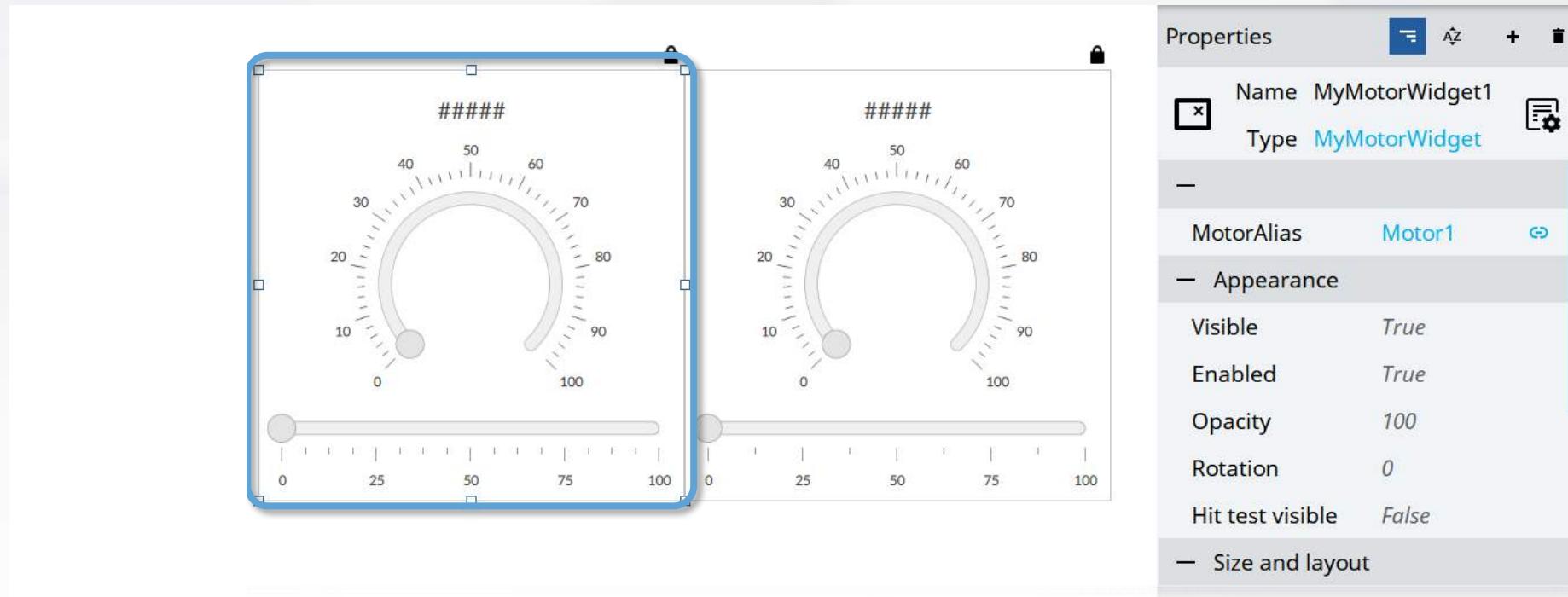
# Custom types: reusable graphics with an alias

- Example: a motor widget that shows the values of different instances of the motor



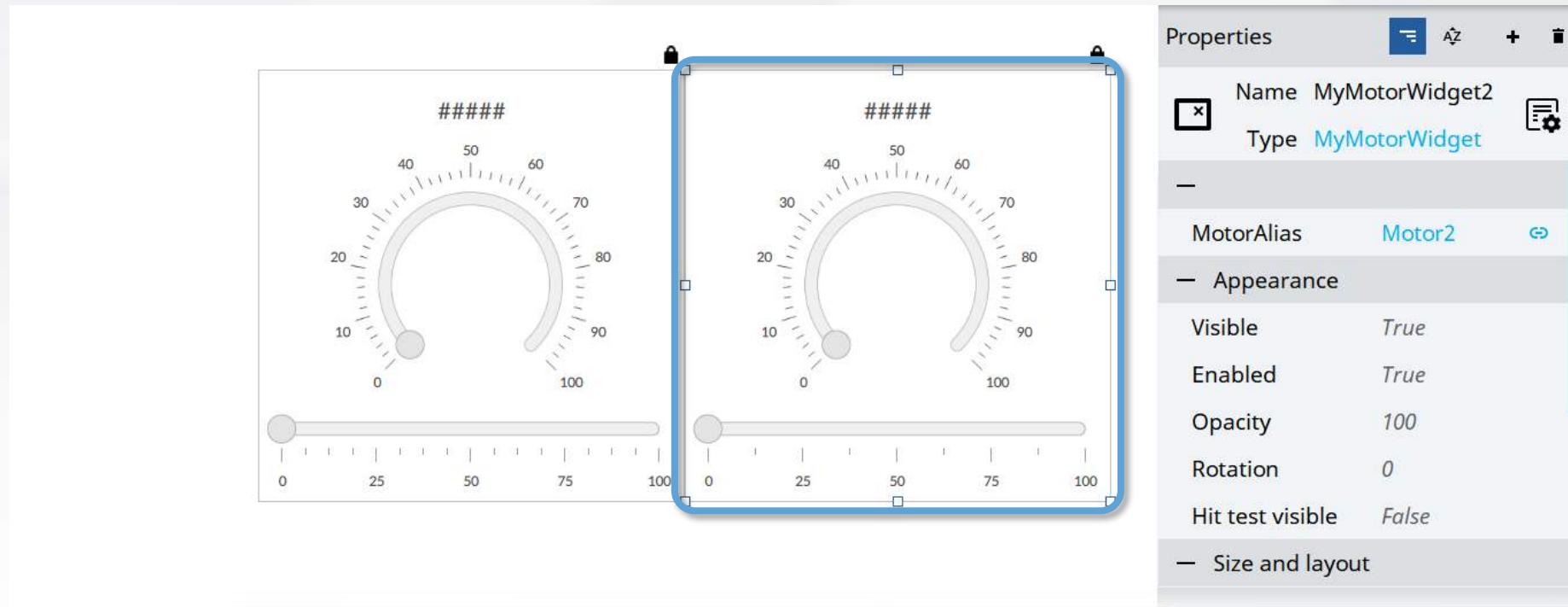
# Custom types: reusable graphics with an alias

- Example: a motor widget that shows the values of different instances of the motor



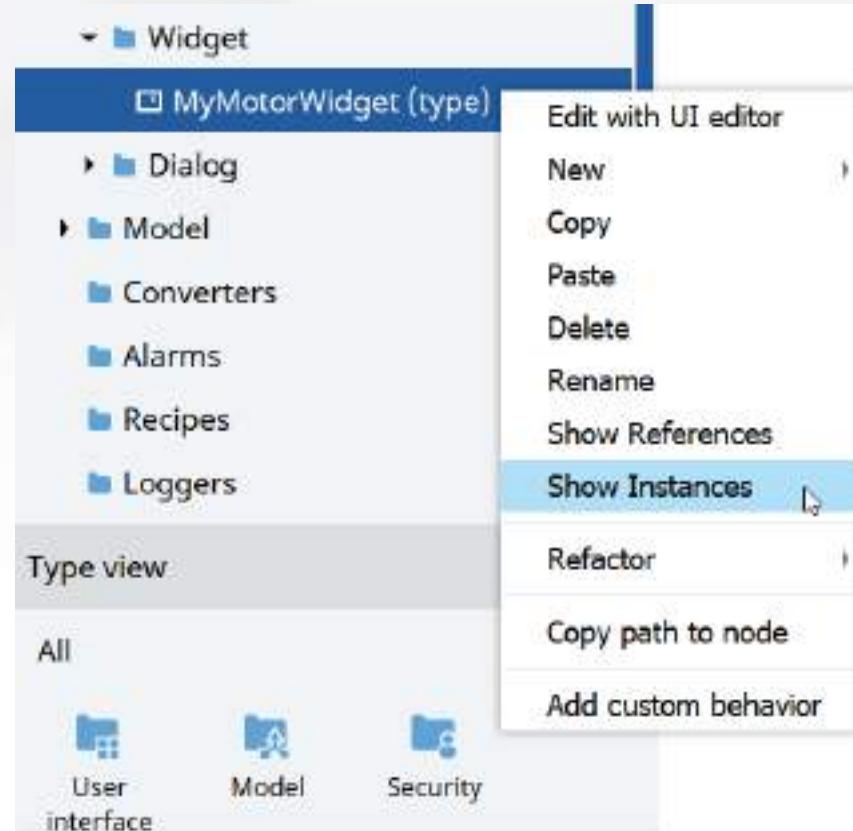
# Custom types: reusable graphics with an alias

- Example: a motor widget that shows the values of different instances of the motor



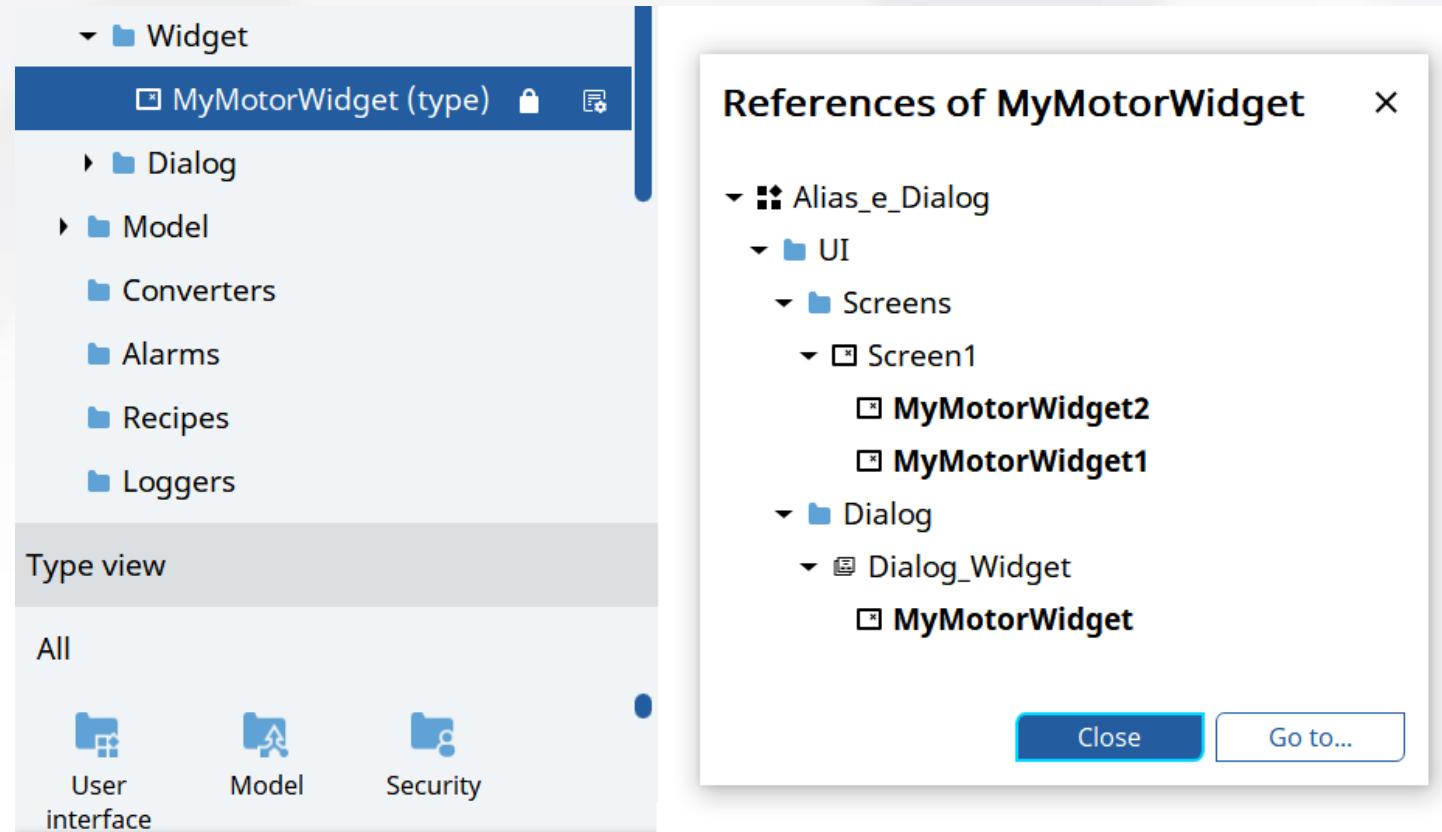
# Custom types: show instances

- Answer the question: "**where are the instances of this type?**"
- Can be used to find instances of
  - Widgets,
  - Panels,
  - Tags (Structure)
  - "typed" Alarm
  - ...



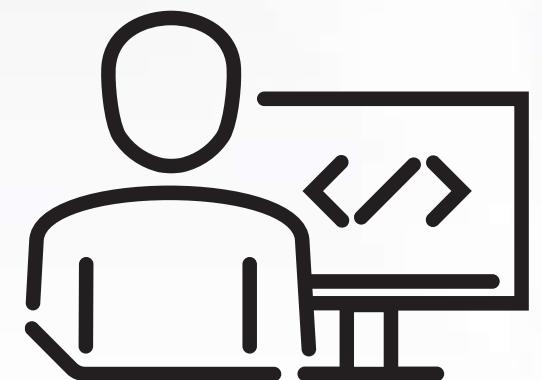
# Custom types: show instances

- Answer the question: "**where are the instances of this type?**"
- Can be used to find instances of
  - Widgets,
  - Panels,
  - Tags (Structure)
  - "typed" Alarm
  - ...



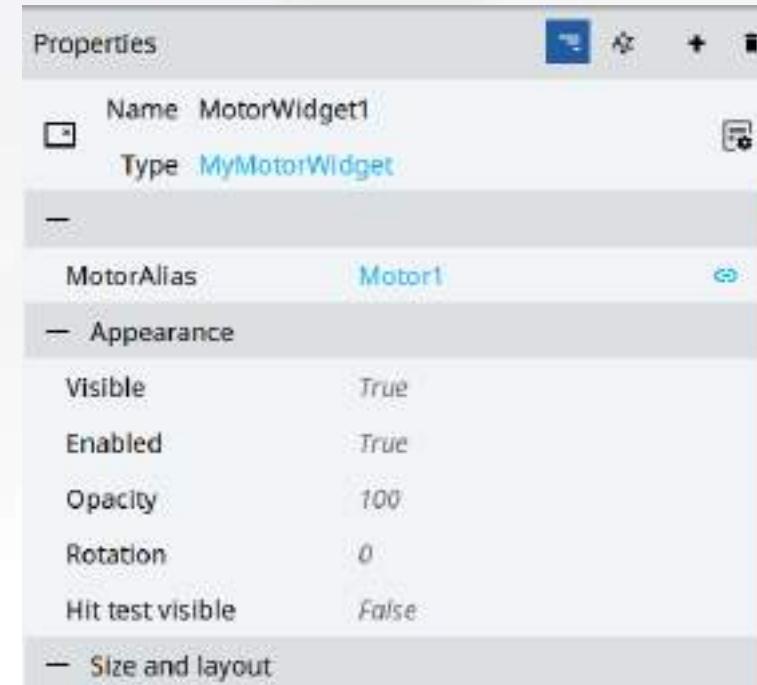
# Hands-on session

- Create a widget called MotorWidget as a Type with an Alias to be used with the Motor variables
- Add 3 instances of the MotorWidget into a new Panel
- Specify, at design time, the alias on each MotorWidget instance associating to each Motor variable



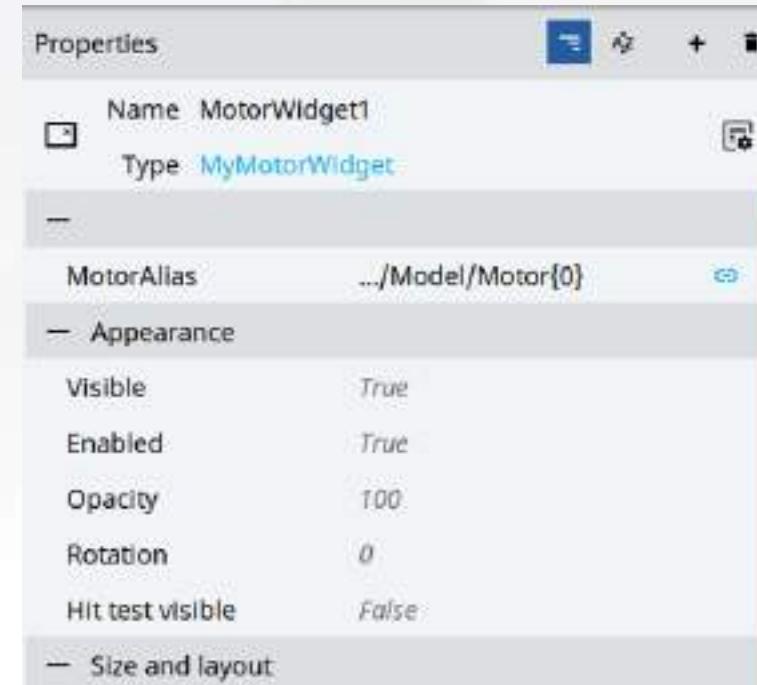
# Reusable graphics: set aliases using indexing

- Alias variables can be indexed like simple variables
- Allow to easily change the alias of a widget



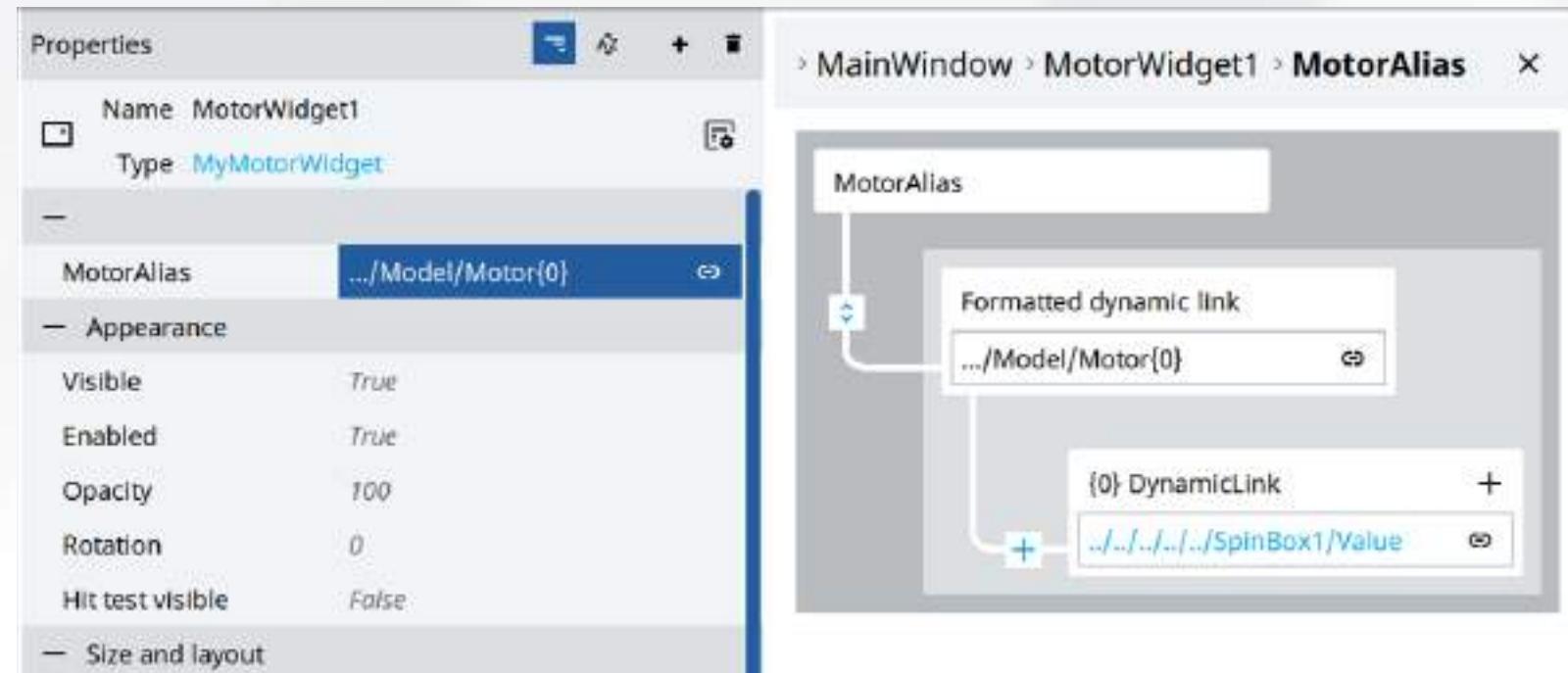
# Reusable graphics: set aliases using indexing

- Alias variables can be indexed like simple variables
- Allow to easily change the alias of a widget



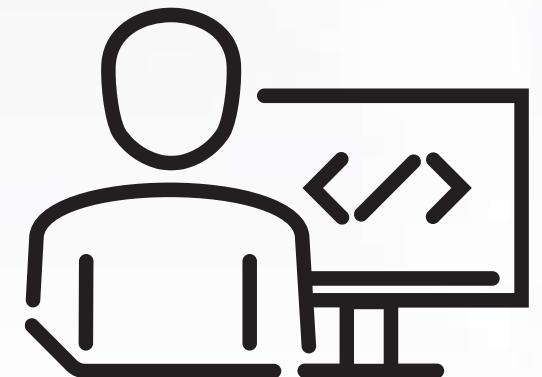
# Reusable graphics: set aliases using indexing

- Alias variables can be indexed like simple variables
- Allow to easily change the alias of a widget



# Hands-on session

- Add just only one instance of the MotorWidget into a new Panel
- add also a Switch object in the Panel
- use the "indexing" method to define the Alias on the MotorWidget instance, associating the index to the Switch value



# Reusable graphics: set aliases at runtime

- For Dialog and Panels,  
the Alias at Runtime can be set on:

- Open Dialog command
- Change Panel command
- Navigation Panel properties
- Panel Loader properties

Events		+
⚡	MouseClicked event	+
⚡	MouseDown event	+
▼	Method	.../UICommands/OpenDialog
▼	Input arguments	
Dialog	Dialog_with_Alias (type)	🔗
AliasNode	Motor1	🔗
⚡	MouseUp event	+

# Reusable graphics: set aliases at runtime

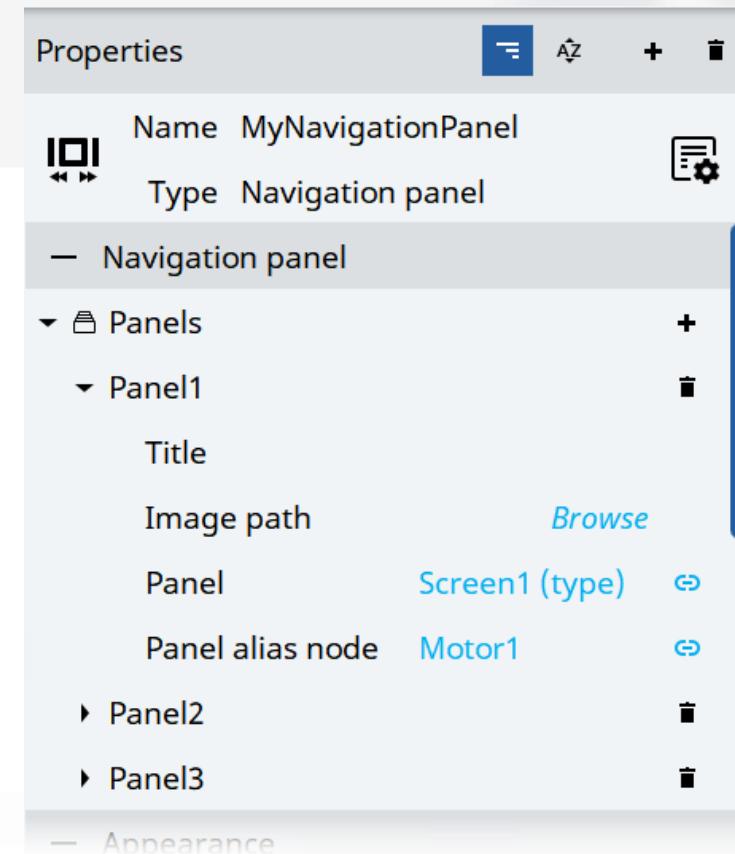
- For Dialog and Panels,  
the Alias at Runtime can be set on:

- Open Dialog command
- Change Panel command
- Navigation Panel properties
- Panel Loader properties

Events		
▼ ⚡ MouseClick event		+
▼ Method	.../PanelLoader/ChangePanel	■
▼ Input arguments		
NewPanel	Screen1 (type)	🔗
AliasNode	Motor1	🔗
▼ ⚡ MouseDown event		+
▼ ⚡ MouseUp event		+

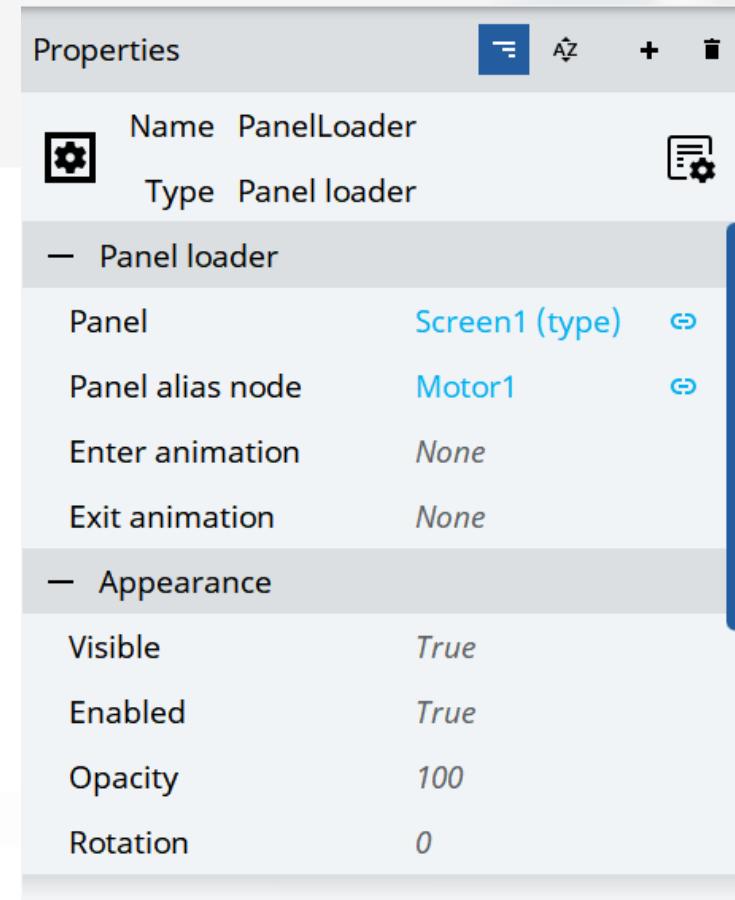
# Reusable graphics: set aliases at runtime

- For Dialog and Panels,  
the Alias at Runtime can be set on:
  - Open Dialog command
  - Change Panel command
  - Navigation Panel properties
  - Panel Loader properties



# Reusable graphics: set aliases at runtime

- For Dialog and Panels,  
the Alias at Runtime can be set on:
  - Open Dialog command
  - Change Panel command
  - Navigation Panel properties
  - Panel Loader properties



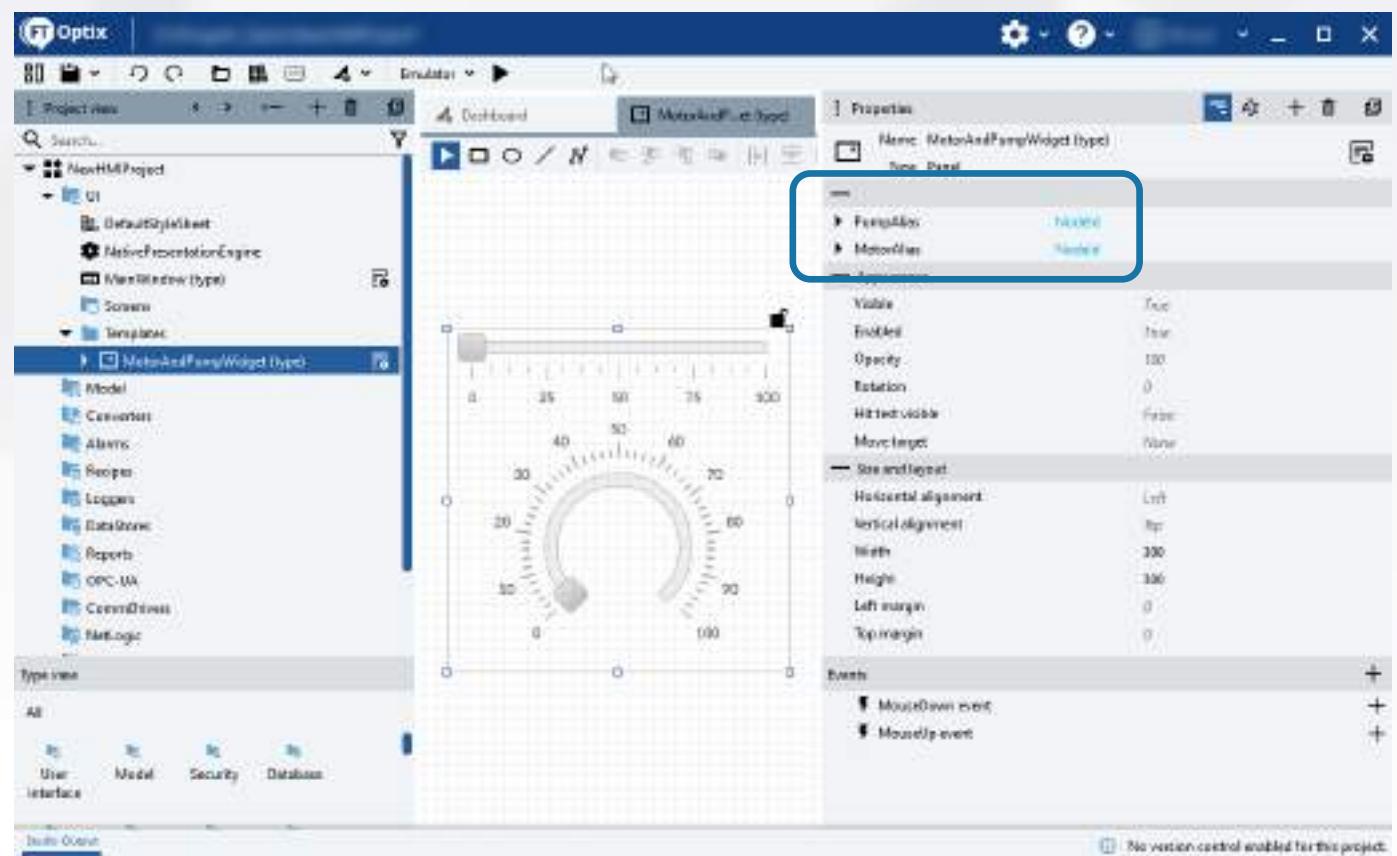
# Hands-on session

- Add 3 Dropdown buttons (it's under the "container" group) into a new Panel
- Configure each button, with the following properties:
  - Panel = MotorWidget type
  - Panel alias node = the Alias of a Motor variable
- NOTE: you "probably" need to add a background to the MotorWidget type



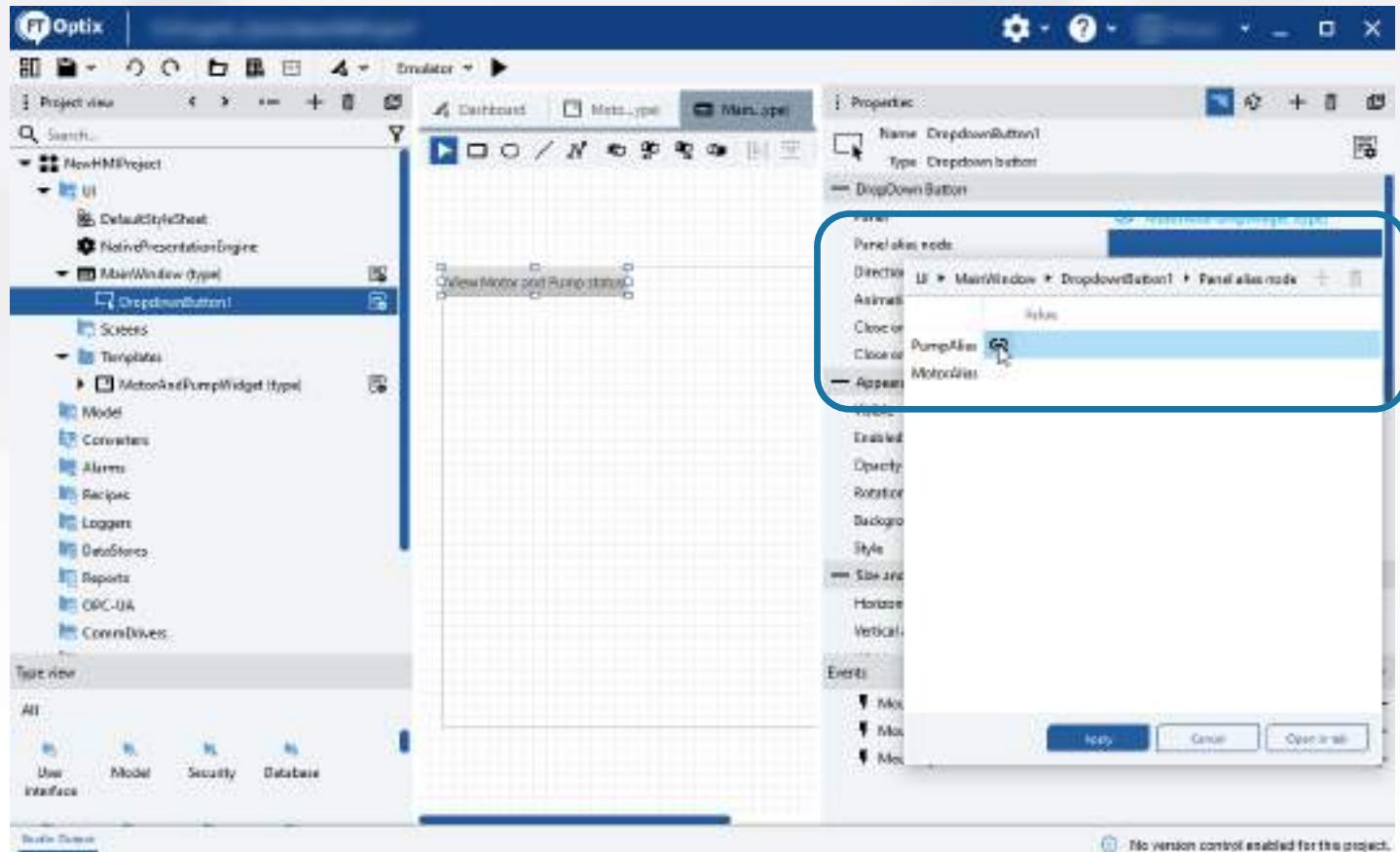
# Multiple aliases

- If multiple aliases are used in a type, the Aliases editor panel will come out when configuring the Alias Node property of a container loader



# Multiple aliases

- If multiple aliases are used in a type, the Aliases editor panel will come out when configuring the Alias Node property of a container loader

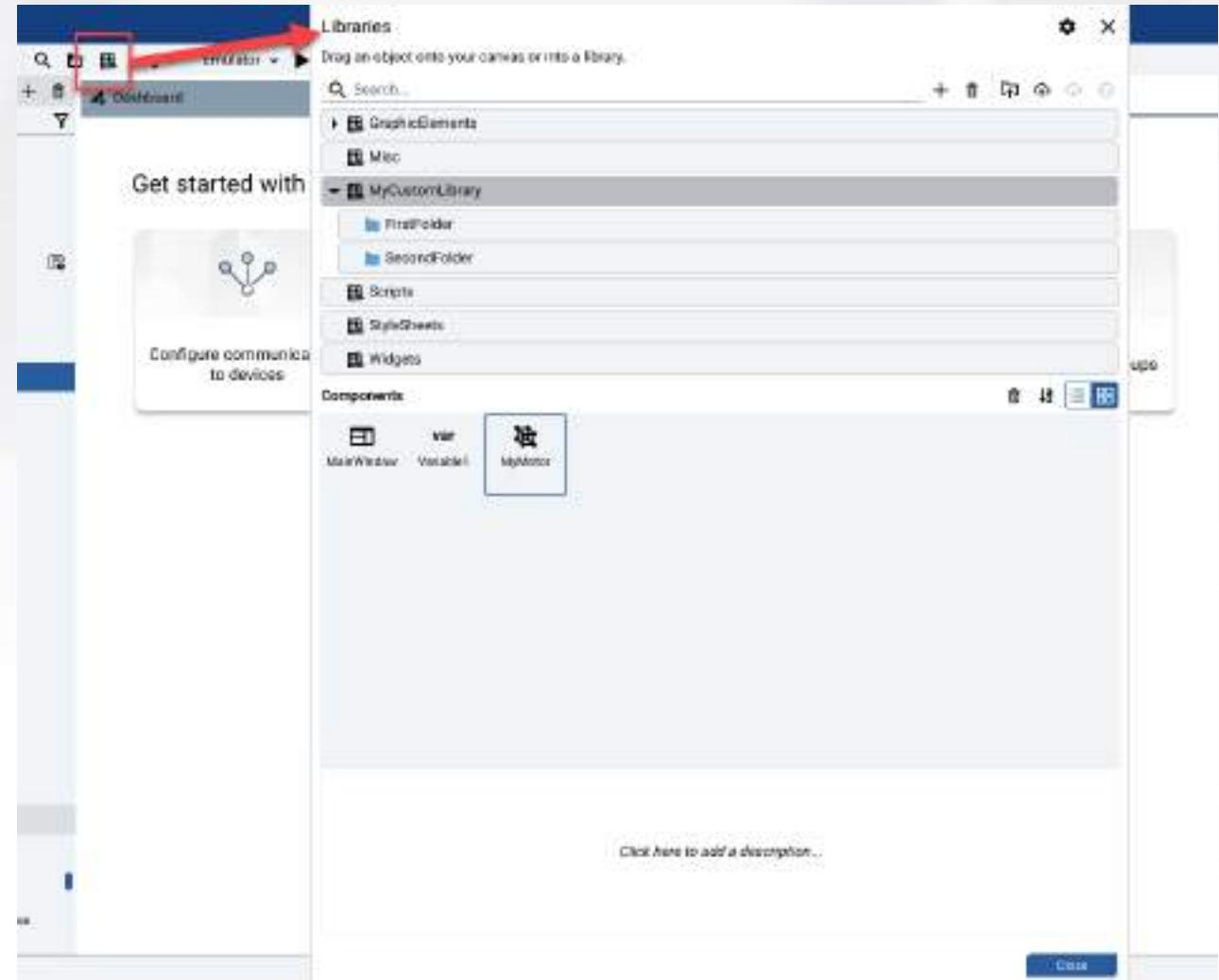


# Template Library



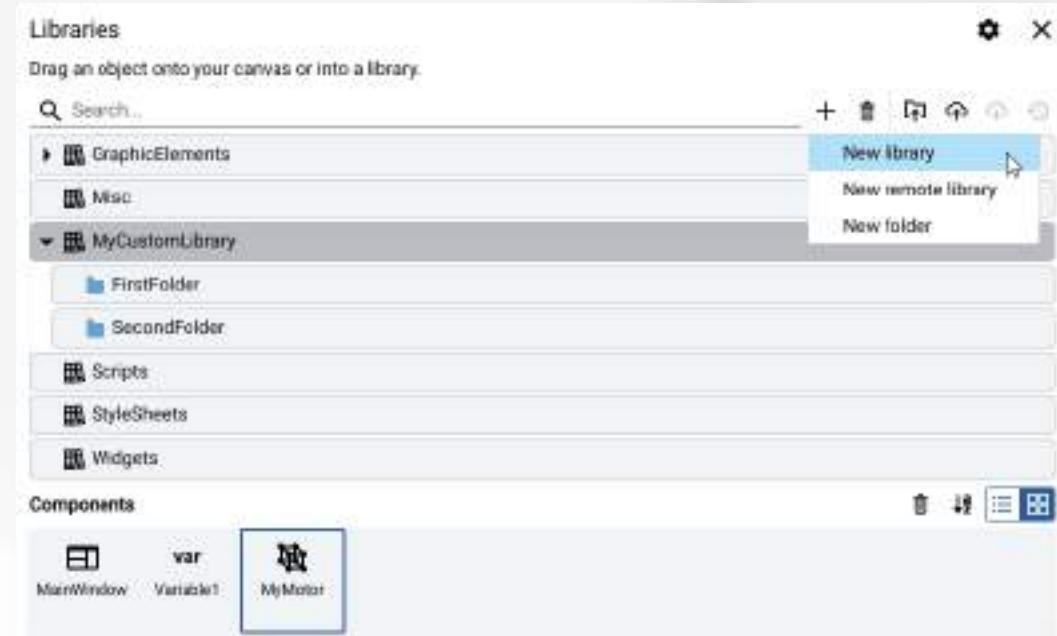
# Template Library

- FactoryTalk Optix contains a good number of default libraries
- Default libraries cannot be modified (read-only)
- User can create unlimited libraries
  - Each library can have multiple subfolders
  - Libraries can contain any kind of element
- Libraries are shared across projects
- Libraries can be versioned with GIT
- Custom libraries are saved in the Documents folder
  - C:\Users\User\Documents\Rockwell Automation\FactoryTalk Optix\Libraries



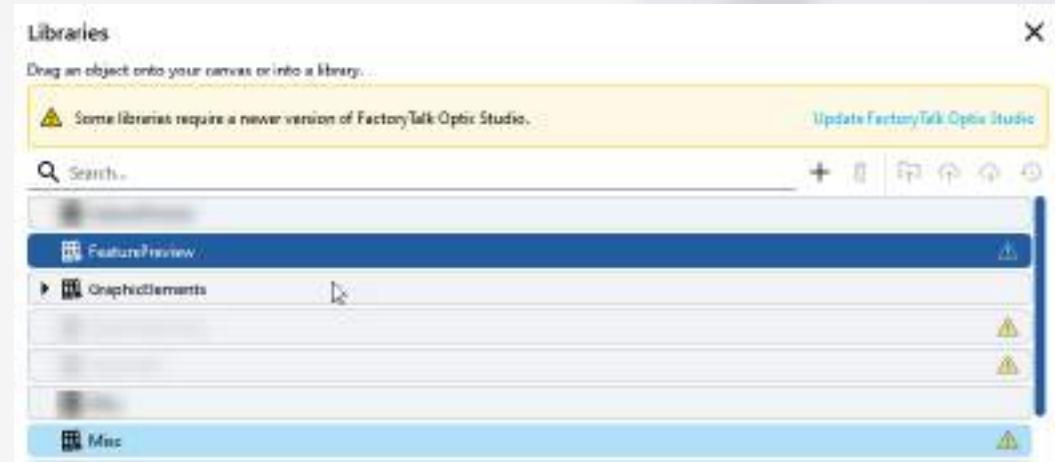
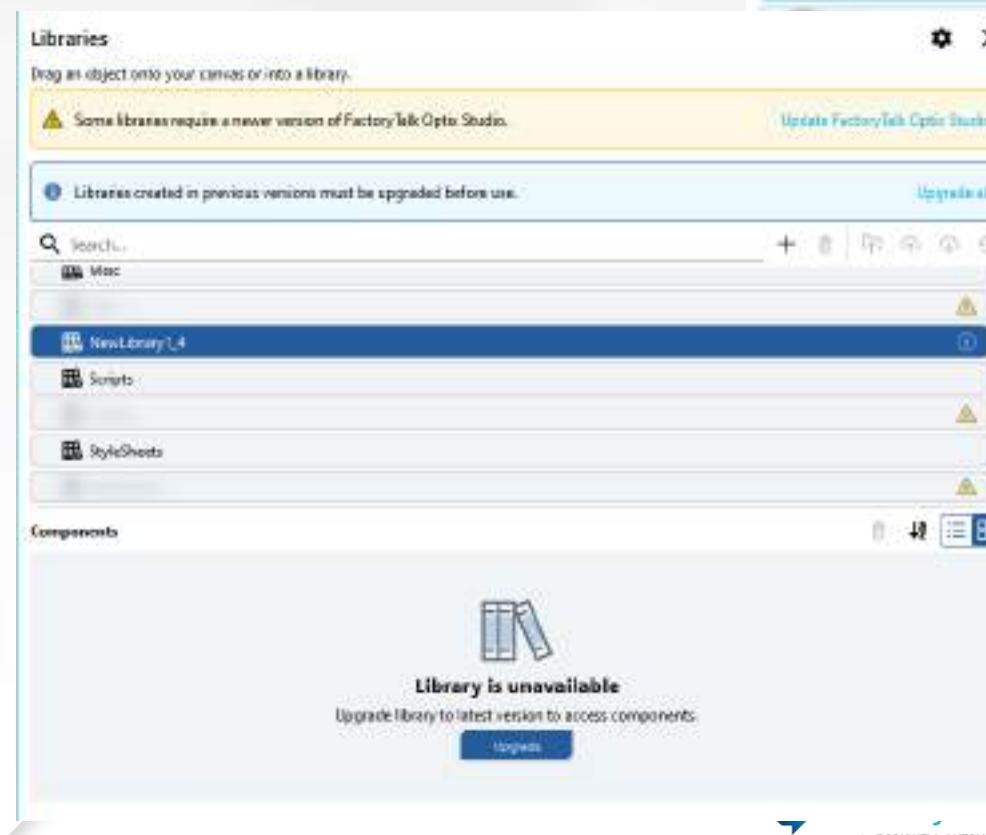
# Template Library

- **New library:** create a new local library
  - Any new library can be versioned by using the dedicated buttons
- **New remote library:** pulls an existing library from a remote GIT provider
- **New folder:** create a subfolder in the selected library
- Version control history works for Template Library too (see the Version Control chapter)



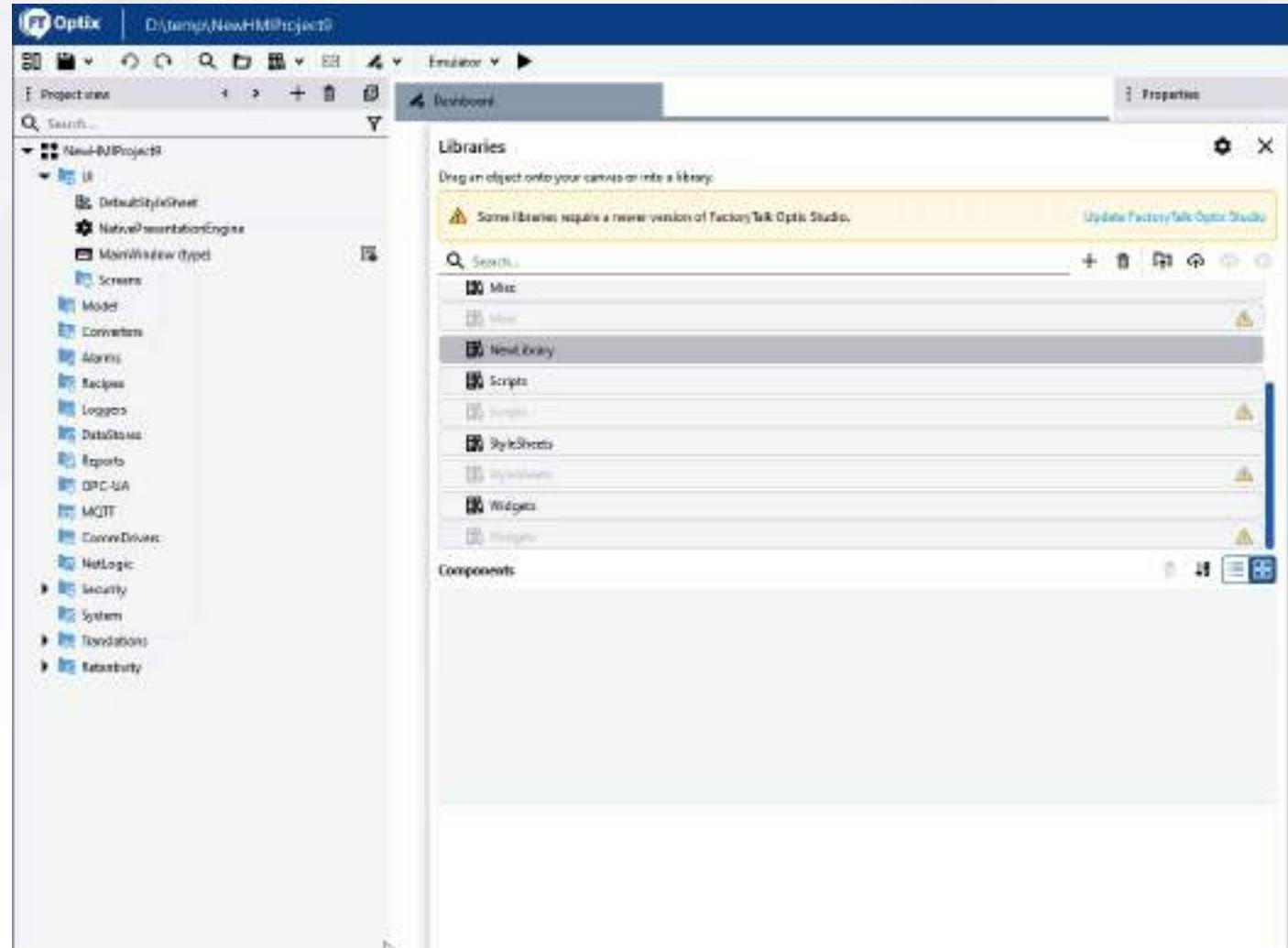
# Template Library

- Libraries are only compatible with the same version of Optix that was used to create them
  - Libraries can be upgraded to a newer version if needed
  - Libraries cannot be downgraded



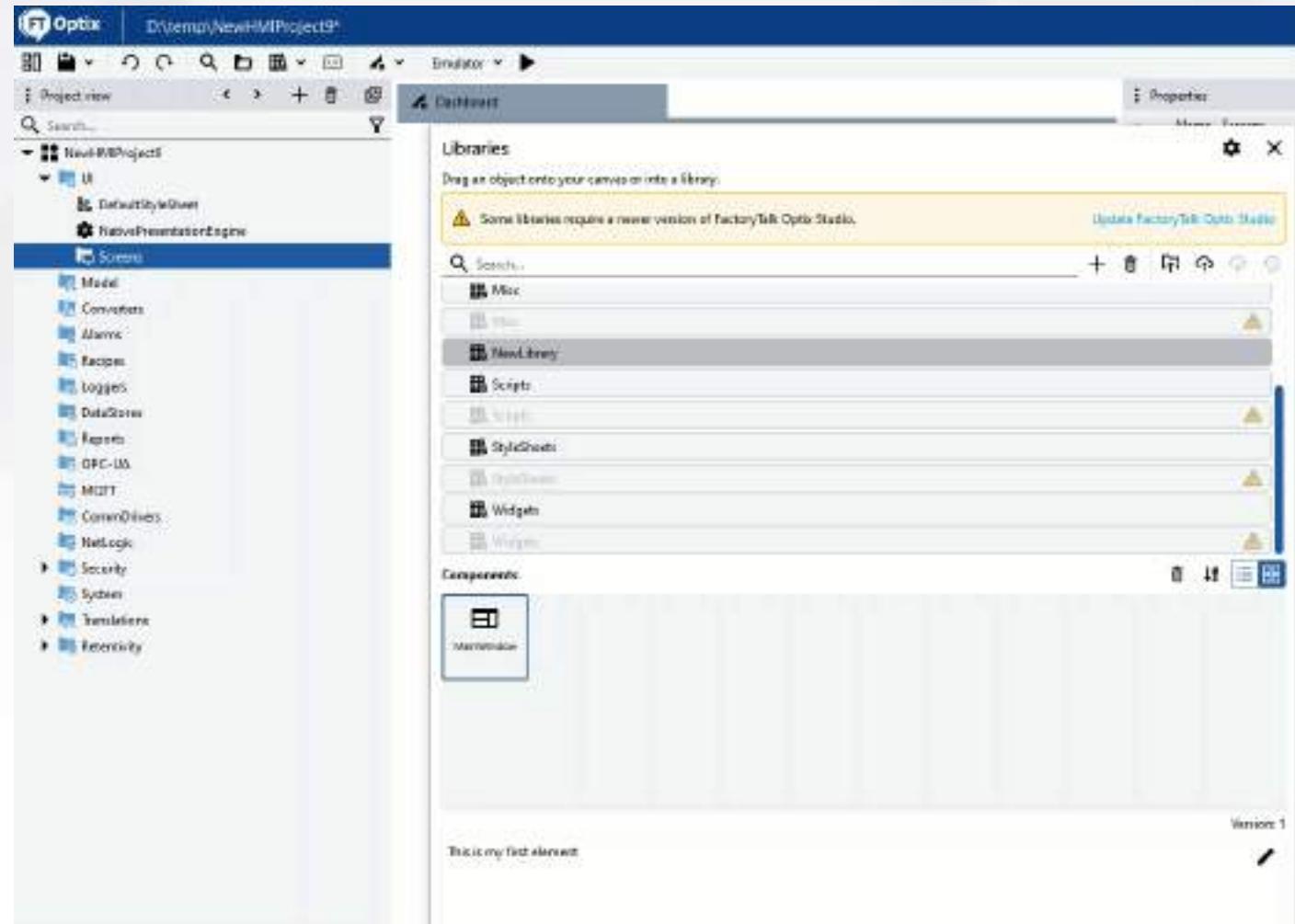
# Template Library

- **Add an element to the library:** drag and drop the element from the project to the Template Library
  - Additional dependencies or types are imported automatically
  - Each element can have a description
- **Import an element from the library:** drag and drop the element from the library to the project
  - Additional dependencies or types are imported automatically
- **Resolving conflicts:** when importing or exporting any element, if the same type already exist, conflict resolution is prompt to the user



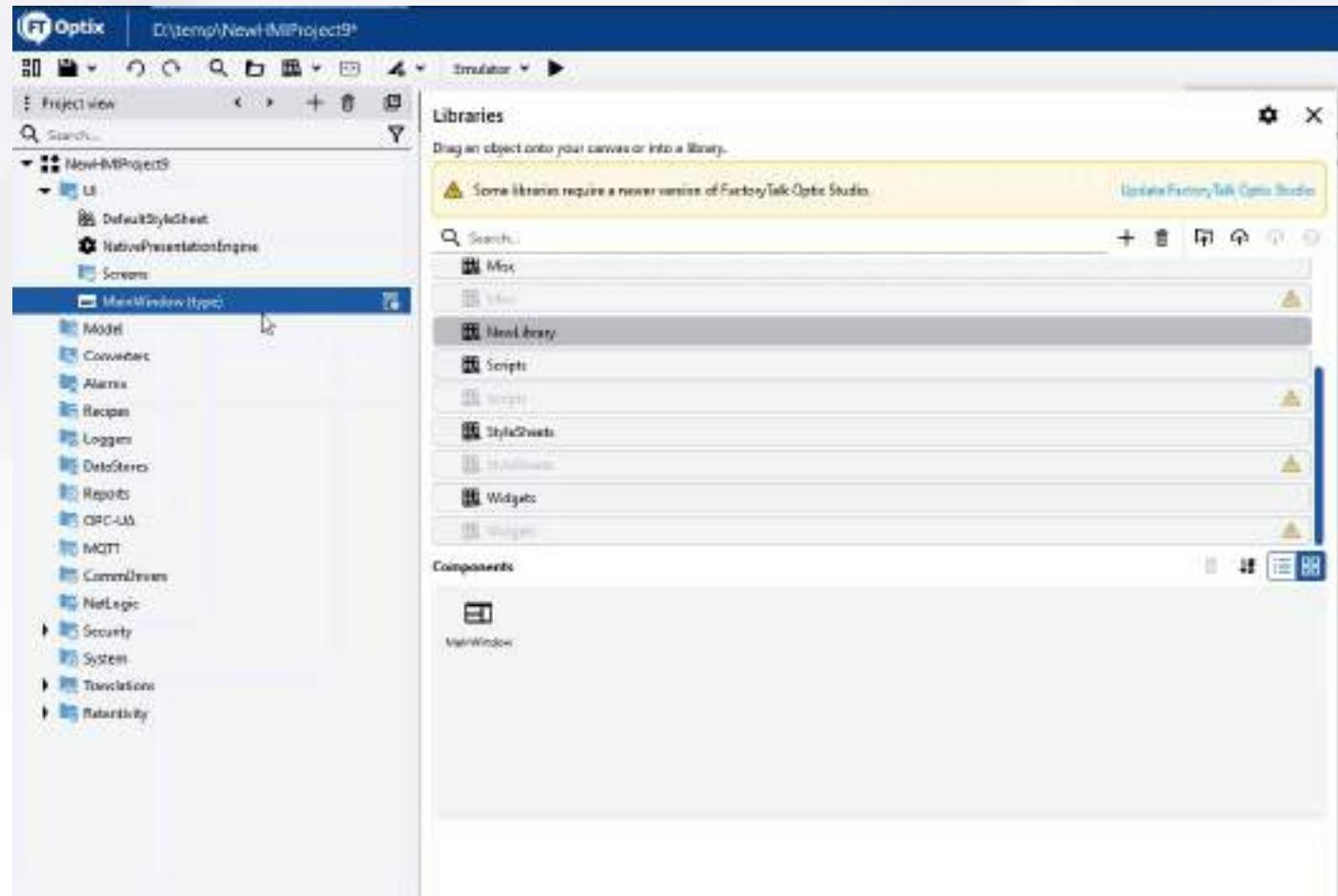
# Template Library

- **Add an element to the library:** drag and drop the element from the project to the Template Library
  - Additional dependencies or types are imported automatically
  - Each element can have a description
- **Import an element from the library:** drag and drop the element from the library to the project
  - Additional dependencies or types are imported automatically
- **Resolving conflicts:** when importing or exporting any element, if the same type already exist, conflict resolution is prompt to the user



# Template Library

- **Add an element to the library:** drag and drop the element from the project to the Template Library
  - Additional dependencies or types are imported automatically
  - Each element can have a description
- **Import an element from the library:** drag and drop the element from the library to the project
  - Additional dependencies or types are imported automatically
- **Resolving conflicts:** when importing or exporting any element, if the same type already exist, conflict resolution is prompt to the user

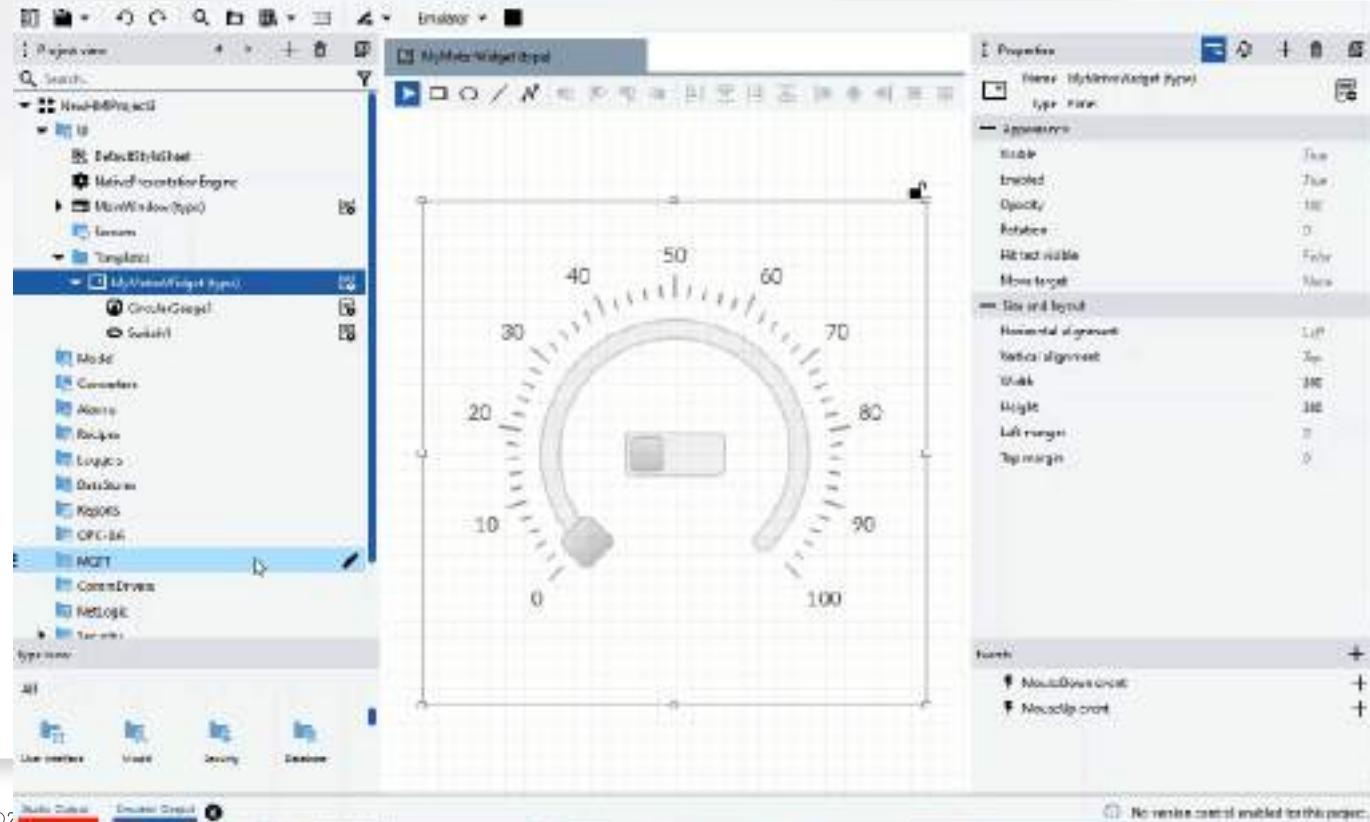


# UI Objects Protection



# UI Objects Protection

- UI Types can be protected to avoid accidental changes
  - Does not encrypt the content of the YAML file or prevents element manipulation from YAML source file
  - Does not encrypt the NetLogic in the UI Object (if any)
  - Used to avoid unauthorized changes of the children elements of an UI Type (e.g: sharing the object with colleagues)
  - Only works on UI Types

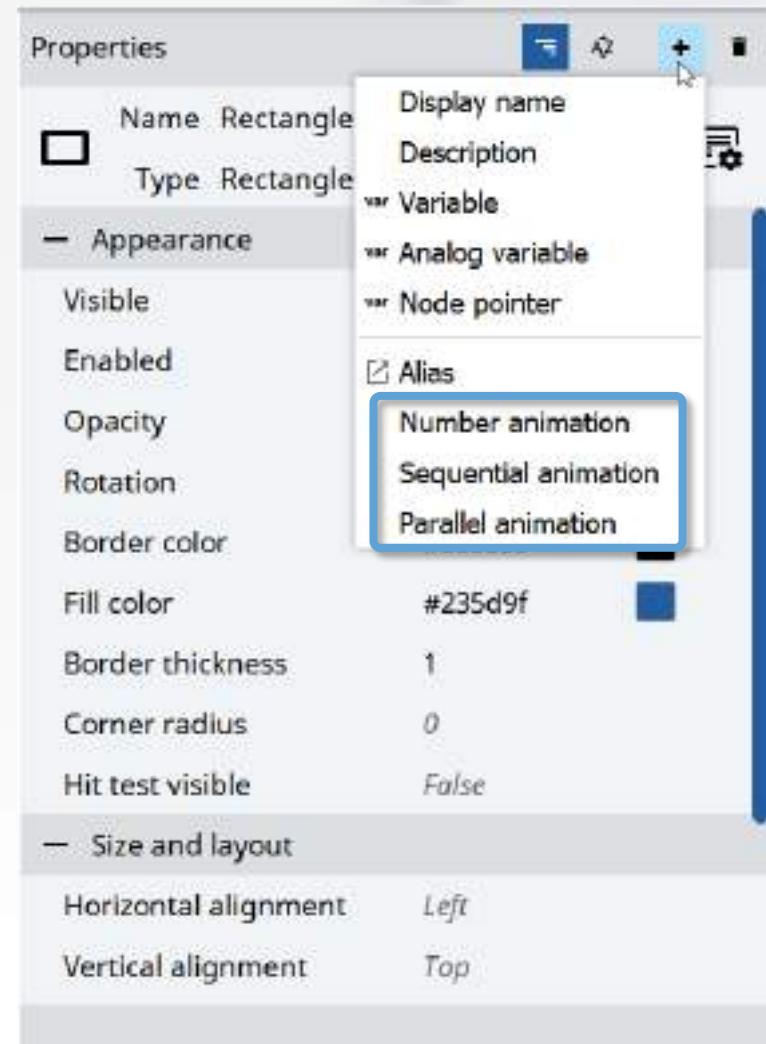


# Animations



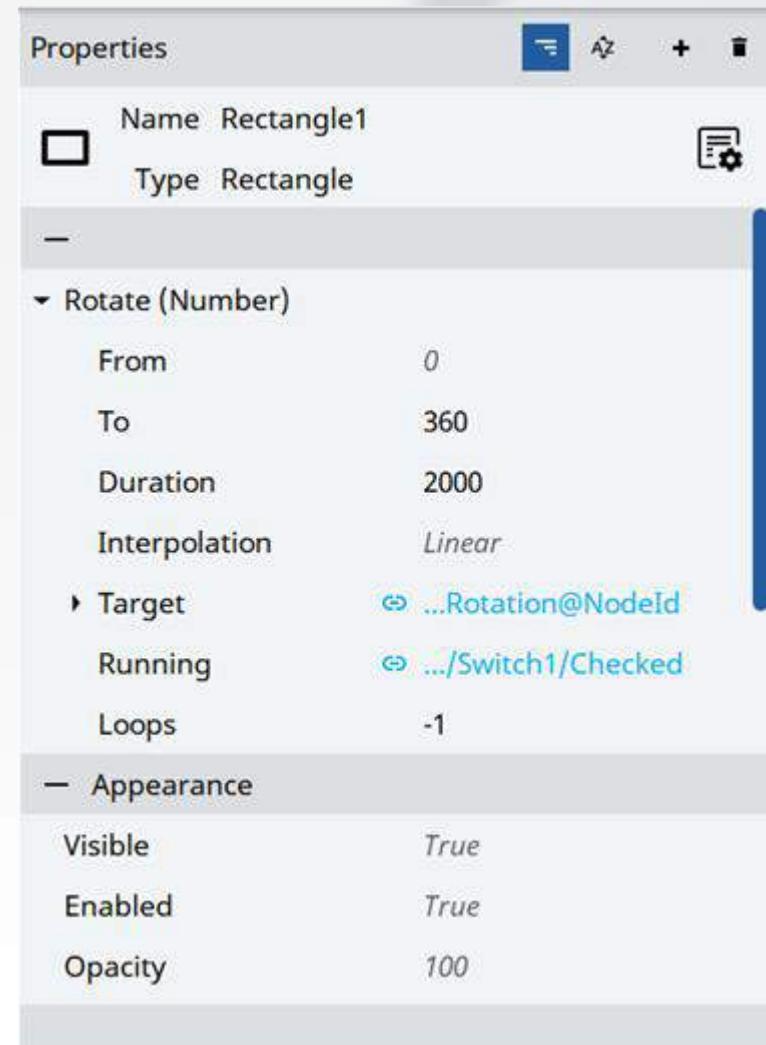
# Animations

- Using Dynamic Links some basic animations can be defined as **just linking a Variable to a property**
  - Visibility, Enabling, Resizing, Change color...
- For "**eye-catching**" Animations:
  - Number animation: single animation
  - Sequential animation: container of animations executed in sequence
  - Parallel animation: container of animations executed in parallel (at the same time)
  - Animation of Behavior: allow to animate a property linked to a Variable



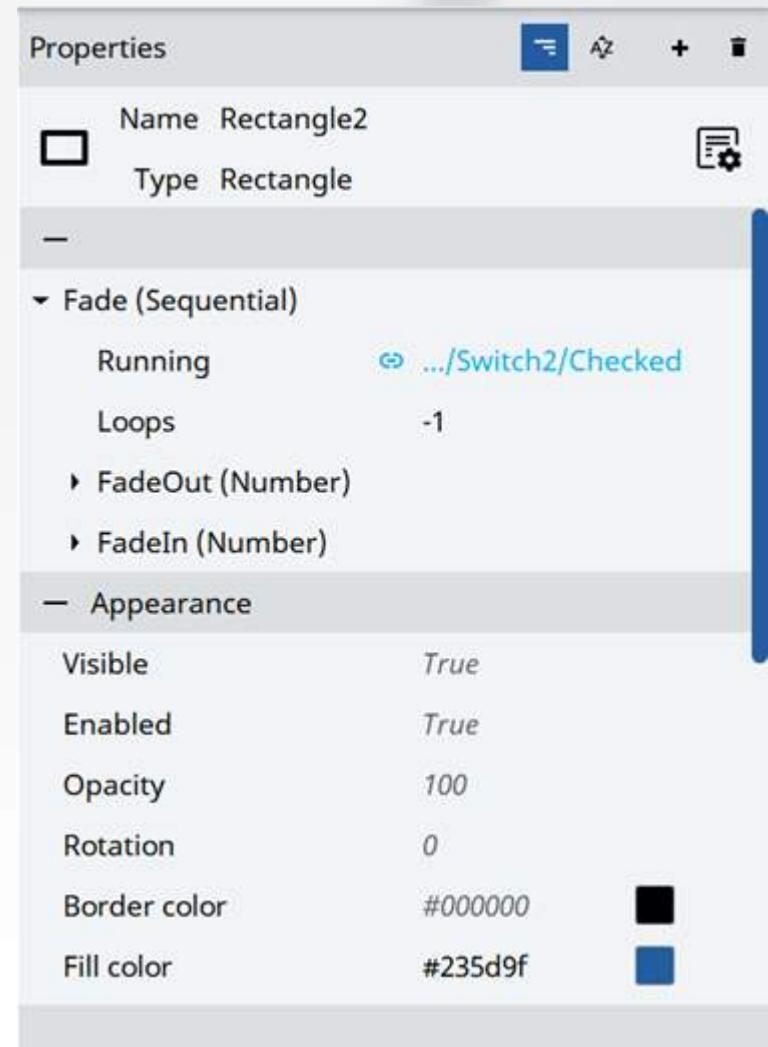
# Animations

- Using Dynamic Links some basic animations can be defined as **just linking a Variable to a property**
  - Visibility, Enabling, Resizing, Change color...
- For "**eye-catching**" Animations:
  - Number animation: single animation
  - Sequential animation: container of animations executed in sequence
  - Parallel animation: container of animations executed in parallel (at the same time)
  - Animation of Behavior: allow to animate a property linked to a Variable



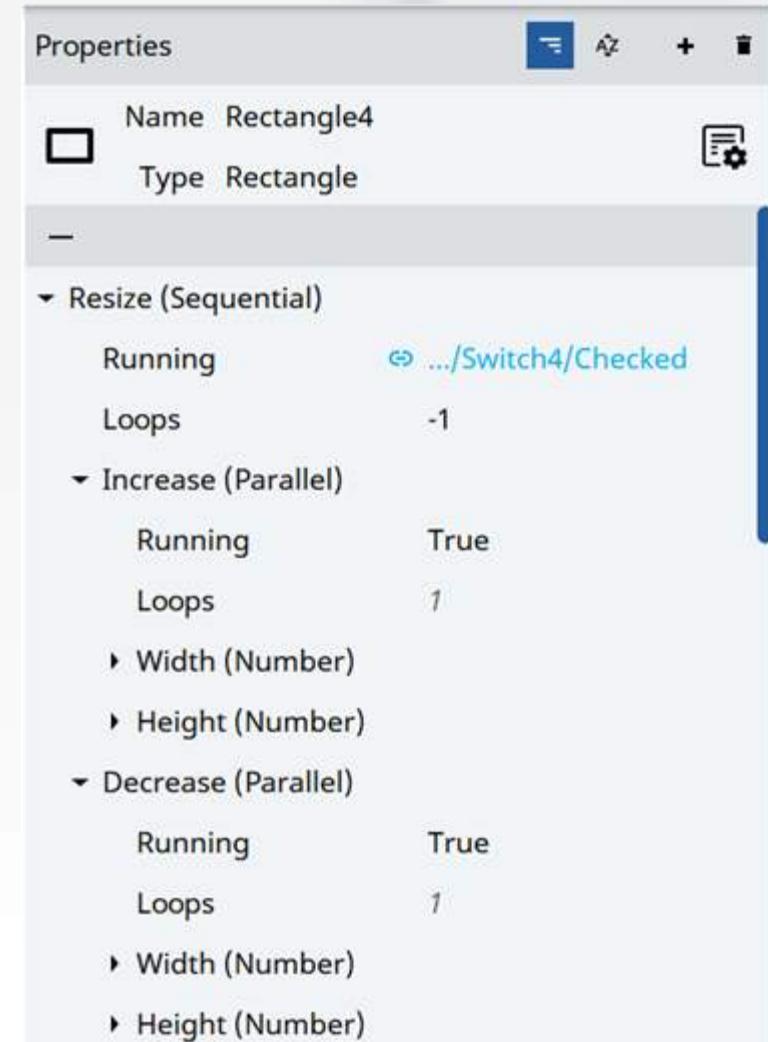
# Animations

- Using Dynamic Links some basic animations can be defined as **just linking a Variable to a property**
  - Visibility, Enabling, Resizing, Change color...
- For "**eye-catching**" Animations:
  - Number animation: single animation
  - Sequential animation: container of animations executed in sequence
  - Parallel animation: container of animations executed in parallel (at the same time)
  - Animation of Behavior: allow to animate a property linked to a Variable



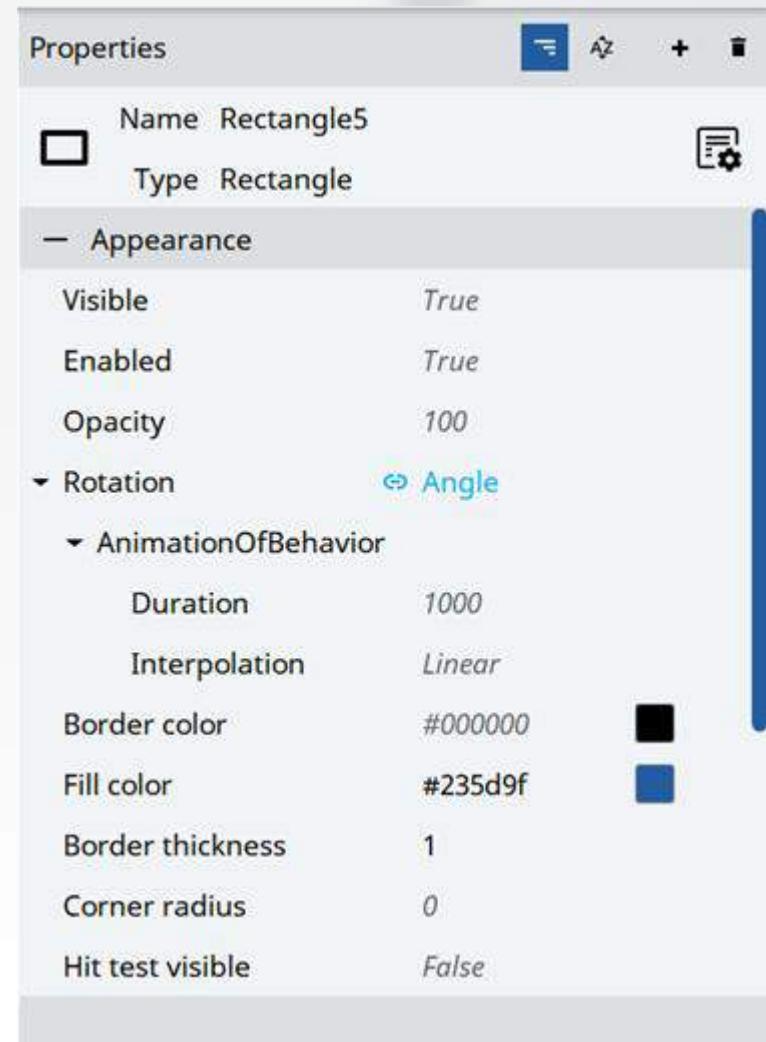
# Animations

- Using Dynamic Links some basic animations can be defined as **just linking a Variable to a property**
  - Visibility, Enabling, Resizing, Change color...
- For "**eye-catching**" Animations:
  - Number animation: single animation
  - Sequential animation: container of animations executed in sequence
  - Parallel animation: container of animations executed in parallel (at the same time)
  - Animation of Behavior: allow to animate a property linked to a Variable



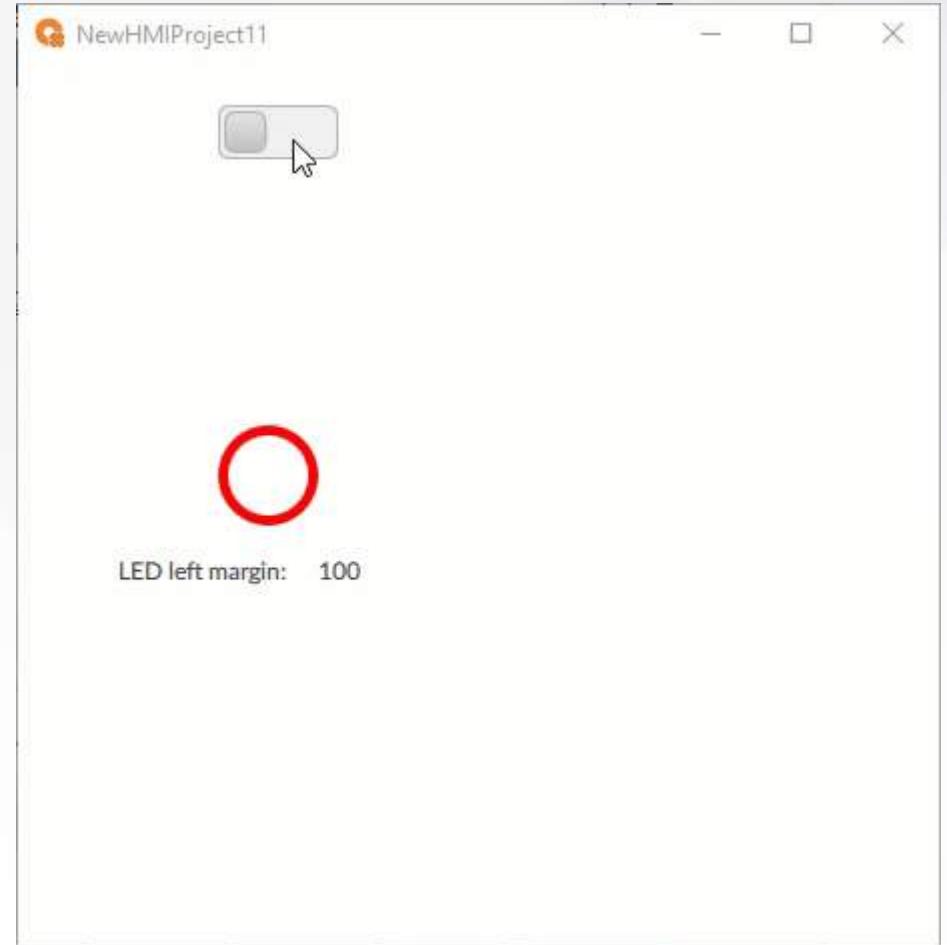
# Animations

- Using Dynamic Links some basic animations can be defined as **just linking a Variable to a property**
  - Visibility, Enabling, Resizing, Change color...
- For "**eye-catching**" Animations:
  - Number animation: single animation
  - Sequential animation: container of animations executed in sequence
  - Parallel animation: container of animations executed in parallel (at the same time)
  - Animation of Behavior: allow to animate a property linked to a Variable



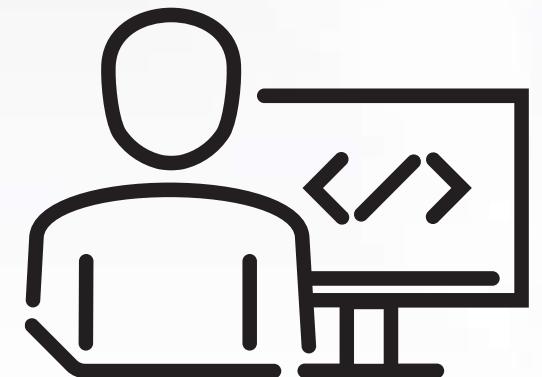
# Things to remember

- Animations are just an “overlay” added before rendering to the variable
- The “real” value of the variable is left untouched



# Hands-on session

- Add a Panel Loader object and add one or more buttons that call the method "Change Panel" to change the panel loaded
- Create a button object that calls the method "Open Dialog" to open a Dialog Box
- Define a Transitioned Event that executes a method like "Variable commands > Set Variable Value" or "Core commands > Close"



# Events



# Commands, methods and events

- Commands/Methods can be executed on Events triggered by objects

- Event Types:

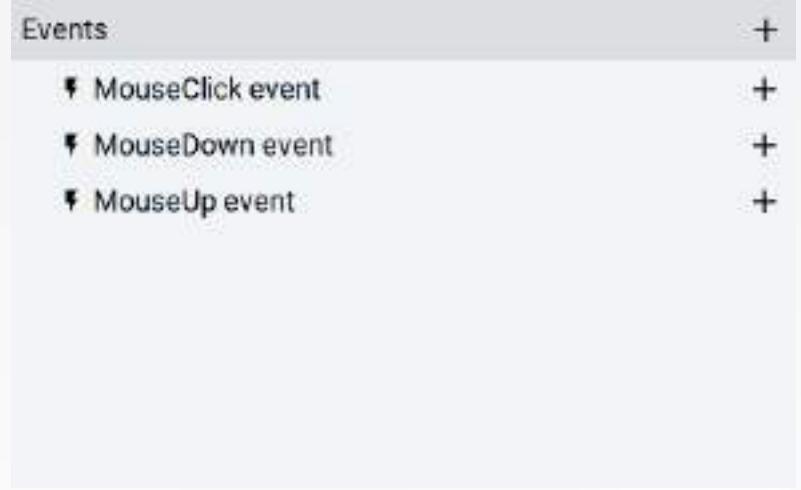
- **Native Event** (Event View): like "MouseClick", "MouseDown", "MouseUP" of a button

- **Changed Event, Transitioned Event, Range Transitioned Event** (Property Panel): custom event

- Method Types:

- Global Methods or **Commands**: like "Change User", or "Open Dialog"...

- Local **Methods exposed by objects**: like "Log" of Datalogger, or "Refresh" of Datalogger viewer...



## Events

- MouseClick event +
- MouseDown event +
- MouseUp event +

# Commands, methods and events

- Commands/Methods can be executed on Events triggered by objects

- Event Types:

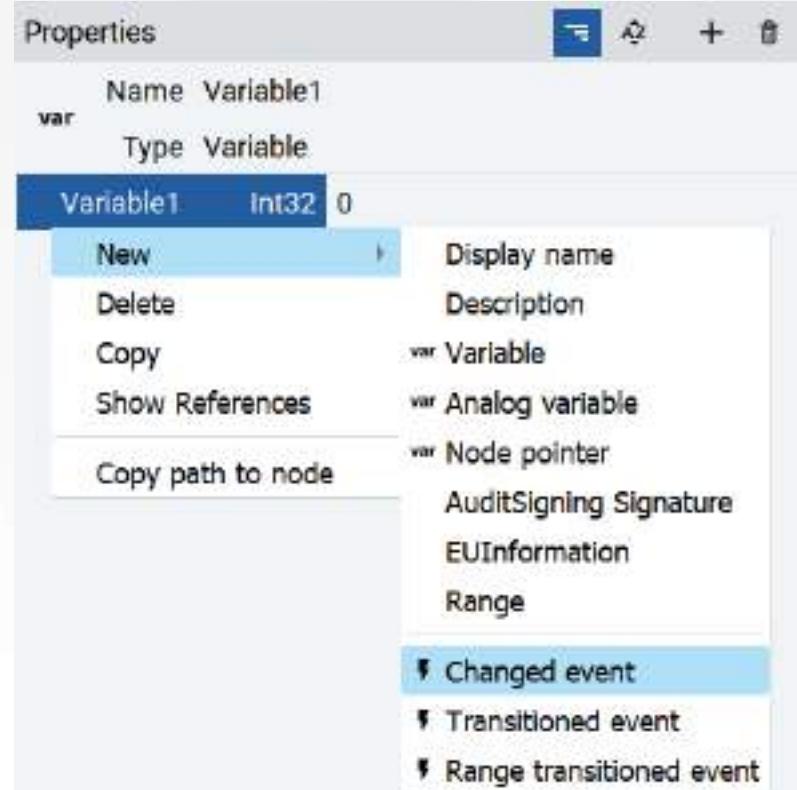
- **Native Event**(Event View): like "MouseClick", "MouseDown", "MouseUP" of a button

- **Changed Event, Transitioned Event, Range Transitioned Event**(Property Panel): custom event

- Method Types:

- Global Methods or **Commands**: like "Change User", or "Open Dialog"...

- Local **Methods exposed by objects**: like "Log" of Datalogger, or "Refresh" of Datalogger viewer...



# Commands, methods and events

- Commands/Methods can be executed on Events triggered by objects

- Event Types:

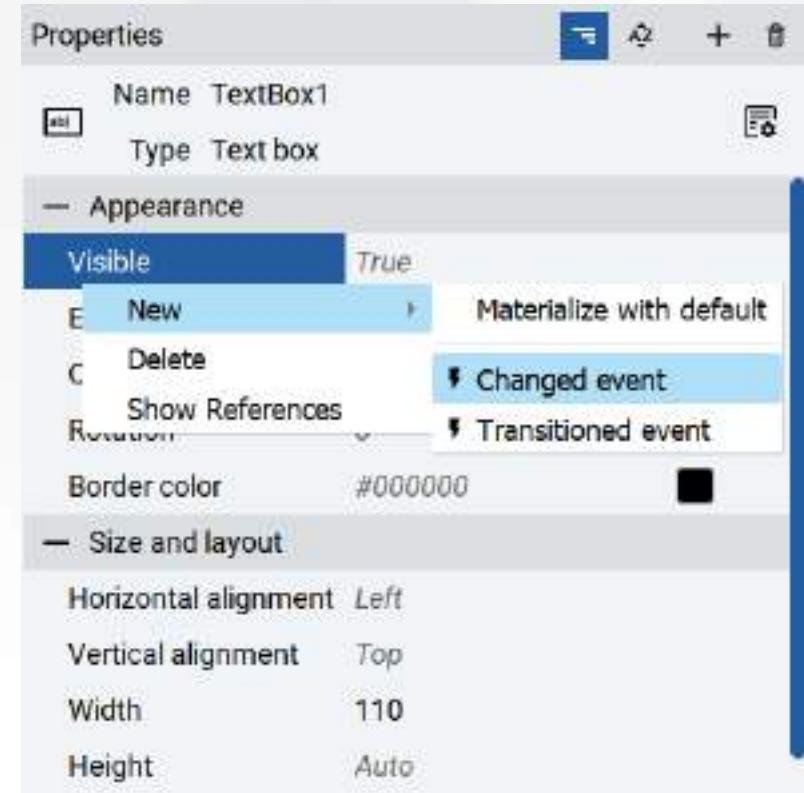
- **Native Event** (Event View): like "MouseClick", "MouseDown", "MouseUP" of a button

- **Changed Event, Transitioned Event, Range Transitioned Event** (Property Panel): custom event

- Method Types:

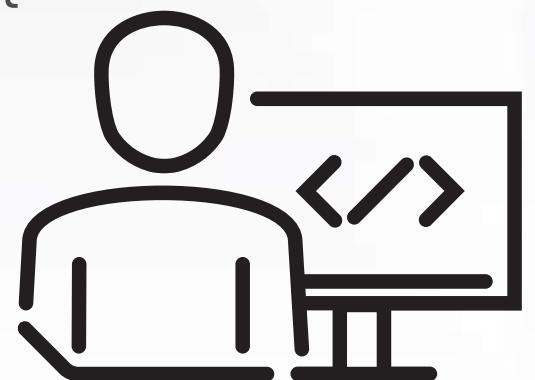
- Global Methods or **Commands**: like "Change User", or "Open Dialog"...

- Local **Methods exposed by objects**: like "Log" of Datalogger, or "Refresh" of Datalogger viewer...



# Hands-on session

- Create a button object that calls the method "Open Dialog" to open a Dialog Box
- Define a Transitioned Event that executes a method like "Variable commands > Set Variable Value" or "Core commands > Close"
- Define an event to display a ScreenSaver Screen after Idle Timeout
  - HINT: add the UISession object and assign Session in Presentation properties
  - Unzip Image file ra\_logo.svg or use any image
    - C:\Program Files\Rockwell Automation\FactoryTalk Optix\Studio\Help\en\downloads

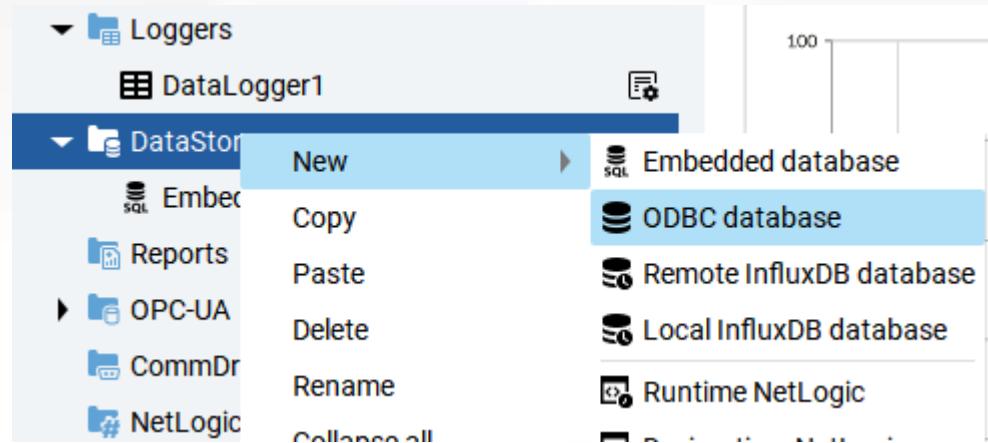


# DataStores



# DataStores

- A DataStore is a database connection
- Types of DataStores
  - **EmbeddedDatabase**: Records values of one or more variables at the same time
  - **ODBC database**: SQL based ODBC connection (SQL Server, MySQL or any derivative)
  - **Remote InfluxDB Database**: Remote PC Timeseries database connection (self-hosted)
  - **Local InfluxDB Connection**: Local PC Timeseries database connection (similar to an EmbeddedDatabase)



# Working with Databases

Objects that can exchange data with a datastore:

- ***Data grid, List box, Combo box*** and *Trend* objects  
query a table to display data at runtime
- *Data logger* and *Event logger* objects  
store data in a table
- *Recipe* object  
stores data in a table and execute a query to display data at runtime

# Working with databases Design-Time

- Emulator will create a copy of the project at runtime
- C:\Users\<USER>\AppData\Local\Rockwell Automation\FactoryTalk Optix\Emulator\Projects

- At Runtime the Embedded database is Locked.
- Path: \ApplicationFiles\<DB Name>.sqlite
- Can be manipulated with SQLite editors like SQLite Browsers, Dbeaver and more.

📁 PKI	11/14/2024 1:37 PM	File folder
📄 56db3d98fa3e3a8d3ae0e34c269688d6.sqlite	11/14/2024 1:39 PM	SQLITE File 8 KB
📄 AlarmsRetentivityStorage	11/14/2024 1:48 PM	Data Base File 92 KB
📄 DB_Example.sqlite	11/14/2024 1:48 PM	SQLITE File 8 KB
📄 SecurityRetentivityStorage	11/14/2024 1:48 PM	Data Base File 92 KB

GenericTableExporter

Name	GenericTableExporter
Type	NetLogic
CSVPath	AbsoluteResourceUri ...ktop\Recipe_Export.CSV <a href="#">Browse</a>
FieldDelimiter	String ,
Table	NodeId <a href="#">Mike_Recipe</a>
Query	String <a href="#">Complex Dynamic Link</a>
WrapFields	Boolean False

- Tables can be exported to CSV via script

# Data Grid, List Box and Combo Box

Data Grid is a tabular view of data at Runtime

Connect to Database, Folder of objects, structure, Enumeration

Brands		
ID	Brands	⋮
1	Electra	
2	Haro	
3	Heller	
4	Pure Cycles	
5	Ritchey	
6	Strider	

Name	dg_Brands
Type	Data grid
— Data grid	
Model	↳ BikeStores
Item kind	↳ DB_Brands
Kind	
Node class	Any
Query	SELECT * FROM brands
Auto refresh time	0000:00:30.000

- Model – DataStore or Folder or object
- Item kind (optional) – Design time structure
- Query – SQLite syntax
- Auto refresh time – how often data is queried – can be triggered on demand

Name	DB_Brands		
Type	Object		
brand_id	Int32	0	
brand_name	String	0	

# Data Grid - Columns

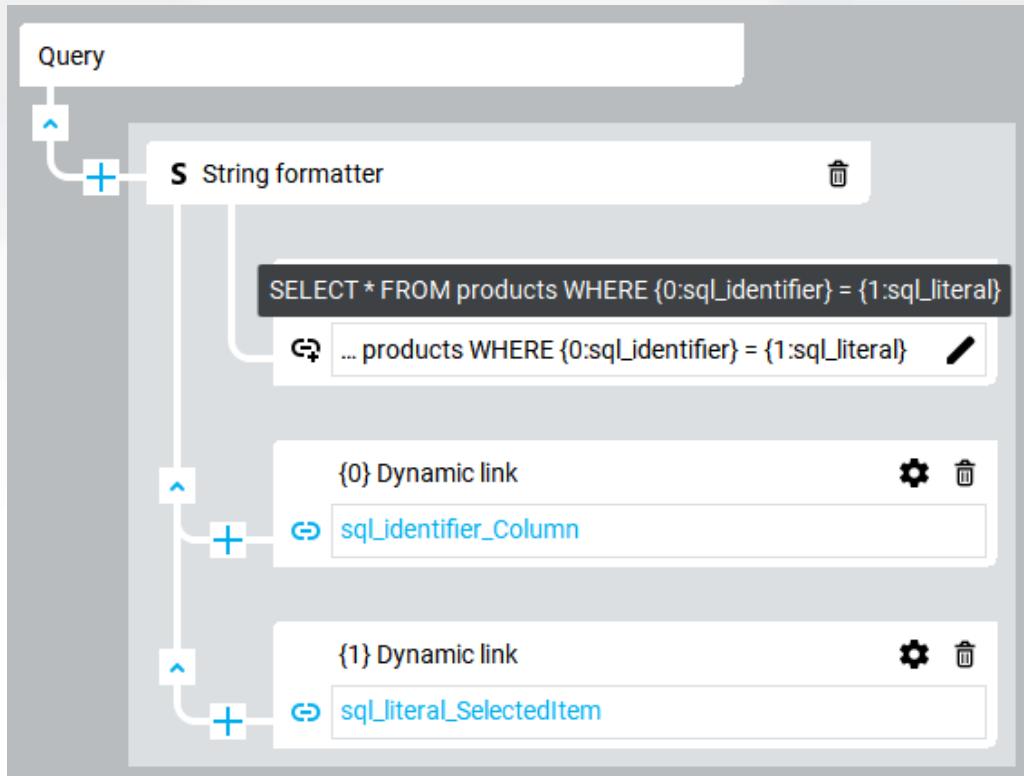
The screenshot shows the FactoryTalk configuration interface with the following details:

- Brands** (Table View): A table with columns ID, Name, and Actions. The row for "Electra" is selected.
- Columns** (Configuration):
  - DataGridColumn1**:
    - Data item template**:
      - Text**: `{Item}/brand_id`
      - Text color**: `#000000`
      - Title**: `ID`
      - Width**: `Auto`
      - Visible**: `True`
      - Header alignment**: `Left aligned`
      - Content alignment**: `Left`
      - Order by**: `{Item}/brand_id`
  - Aliases** (Configuration):
    - {Session}**
    - {Window}**
    - {Item} (DB\_BikeStores/UI/Screens/Select\_Bik**:
      - DB\_Brands**:
        - `var brand_id`
        - `var brand_name`

- **Text - {Item}/ColumnName**
  - For embedded Database
    - Browse into Aliases - {Item}/
    - Set Item kind to Datastore - Tables - columns
  - For ODBC Database
    - Browse into Aliases - {Item}/
    - Type in column name or create a model object for structure
- **Title - Column label**
- **Query - SQLite syntax**
- **Order by - At least one column in DataGrid needs to be set**

# Database Queries

- Query – SQLite syntax supports a subset of statements such as:
  - SELECT
  - DELETE
  - UPDATE
  - CREATE and DROP
  - JOIN
  - Many more...
- When using the String formatter use placeholders
    - {0:sql\_identifier} To define a column name for example
    - {1:sql\_literal} To define a number for example



# Database Queries

- There are 3 ways to Interact with a database

1. Loggers

- Data Logger or Event Logger can be set to "None" for OnDemand Writes or Periodic or OnChange

2. SQL Methods

- Use String Formatter to build query

3. Scripts

- See FeaturesDemo2 for examples

The screenshot shows the FactoryTalk configuration interface. On the left, there is a tree view under 'DataStores' for 'EmbeddedDatabase'. The nodes include 'InsertValues', 'Tables', 'Add column', 'Add table', 'Backup', 'Insert', 'Query', 'Remove column', 'Remove table', 'Rename column', 'Rename table', and 'Restore'. To the right, there are two sections for 'Events'. The top section shows a 'MouseClick event' with a 'Method' of '...ataStores/EmbeddedDatabase/Query' and an 'Input arguments' section containing a 'Query' of 'DELETE FROM DataLogger'. The bottom section shows another 'MouseClick event' with a 'Method' of '...ommands/VariableCommands/Set' and an 'Input arguments' section containing 'VariableToModify' with a value of '.../.../.../DataGridQuery/Query@NodeID', 'Value' as a 'String' of 'SELECT \* FROM DataLogger', and 'ArrayIndex' of '0'.

# Combo box and List box

The screenshot shows the configuration interface for a 'ComboBox' component named 'ComboBox1'. The properties are listed as follows:

- Name: ComboBox1
- Type: Combo box
- Model: BikeStores  
SELECT \* FROM brands
- Mode: Normal
- Display value path: {Item}/brand\_name
- Selected value path: BaseDataType {Item}/brand\_id
- Selected item: BaseDataType brand\_id
- Selected value: brand\_id
- Allow deselection: False

Below the properties, a preview window displays a list of bicycle brands: Heller, Electra, Haro, Heller (selected), Pure Cycles, Ritchey, Strider, Sun Bicycles, and Surly.

## The most flexible of objects

Similar to DataGrid but has additional functions

- Model – Datastore or Folder or Object
- Query – SQLite elect statement, use string formatter for advanced select
- Mode – Combo box – Normal, Search or Editable
- Display Value Path – Values to display
- Selected Value Path – Values to select
- Selected Item – Nodeld of selected (nodeld is the unique ID reference)
- Selected Value – Value of selected

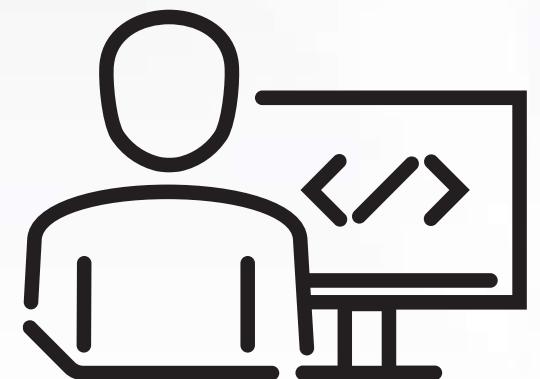
# Queries limitations

- Support only for Standard SQL 2003 syntax
  - No support for specific dialects like SQL Server, MySQL or Flux
- List of supported clauses:
  - [SQL queries](#)
  - [Influx database limitations](#)

Operator	Example
IN	<code>SELECT * FROM Table1 WHERE Column1 IN (10, 20, 30)</code>
BETWEEN	<code>SELECT * FROM Table1 WHERE Column1 BETWEEN 100 AND 200</code>
LIKE	<code>... WHERE column1 LIKE '%a'</code> <code>... WHERE column1 LIKE 'abc'</code> <code>... WHERE column1 LIKE 'Hello!Apples' ESCAPE '\'</code>
EXISTS	<code>... WHERE EXISTS (SELECT Table1.Column1 FROM Table2)</code>
IS	<code>... WHERE column1 IS NULL</code>
NOT	<code>... WHERE column1 IS NOT NULL</code> <code>... WHERE column1 NOT IS (10, 20)</code>

# Hands-on session

- Create a project with an EmbeddedDatabase and log some data
- Configure a DataGrid to show data from a DataBase
- Configure Combo box to filter by values
- Backup and Restore Embedded Database content



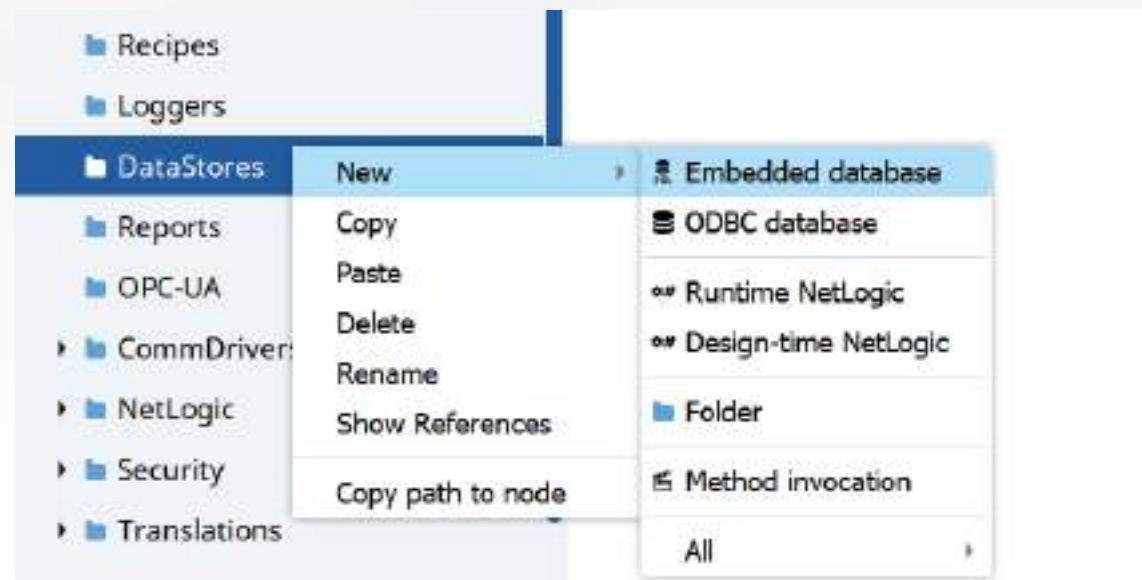


# Save historical data with loggers



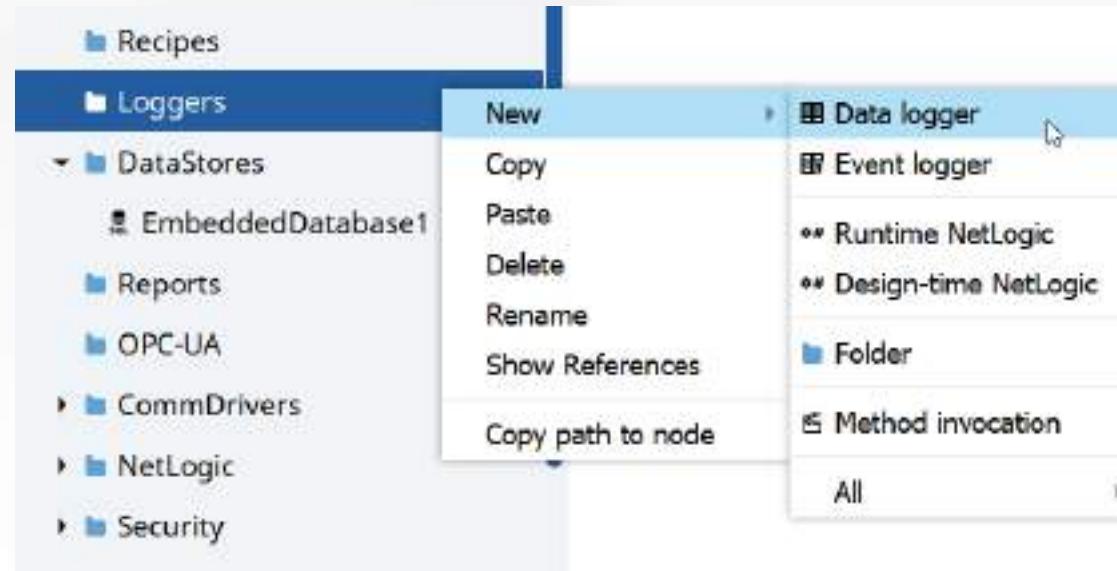
# Loggers

- A logger is an object that records data and stores them in a database
- Types of logger
  - **Data logger:** Records values of one or more variables at the same time
  - **Event logger:** Records the properties of one or more events



# Loggers

- A logger is an object that records data and stores them in a database
- Types of logger
  - **Data logger:** Records values of one or more variables at the same time
  - **Event logger:** Records the properties of one or more events



# Data logger

The screenshot shows the FactoryTalk DataLogger1 configuration interface. The left sidebar displays a project tree under 'MyProject':

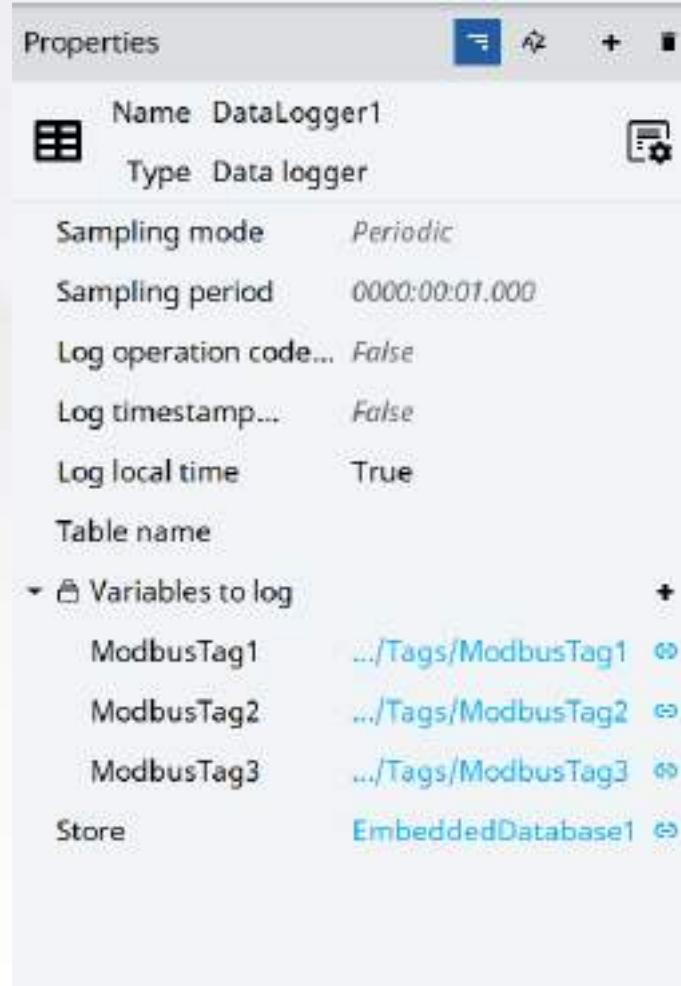
- UE
- Model
- Converters
- Alarms
- Recipes
- Loggers
- DataStores
- Reports
- OPC-UA
- CommDrivers
  - ModbusDriver1
    - ModbusStation1
      - Tags
        - ModbusTag1 (checked)
        - ModbusTag2 (checked)
        - ModbusTag3 (checked)
      - Types
- NetLogic

- At every sampling event the Datalogger create a new record into a table with the following structure:

Date/Time	Variable 1	Variable 2	...	Variable N
hh:mm:ss	value	value	...	value
hh:mm:ss	value	value	...	value

- Select Variables to record
  - Model tags, CommDrivers Tag, Objects properties...
- Configure Datalogger properties
  - Sampling Mode: Periodic/Change in Value
  - Sampling Period: dd:hh:mm:ss.fff
  - Table name (optional)
  - Store: DataStore used to save historical data

# Data logger



- At every sampling event the Datalogger creates a new record into a table with the following structure:

Date/Time	Variable 1	Variable 2	...	Variable N
hh:mm:ss	value	value	...	value
hh:mm:ss	value	value	...	value

- Select Variables to record
  - Model tags, CommDrivers Tag, Objects properties...
- Configure Datalogger properties
  - Sampling Mode: Periodic/Change in Value
  - Sampling Period: dd:hh:mm:ss.fff
  - Table name (optional)
  - Store: DataStore used to save historical data

# Data logger "viewer"

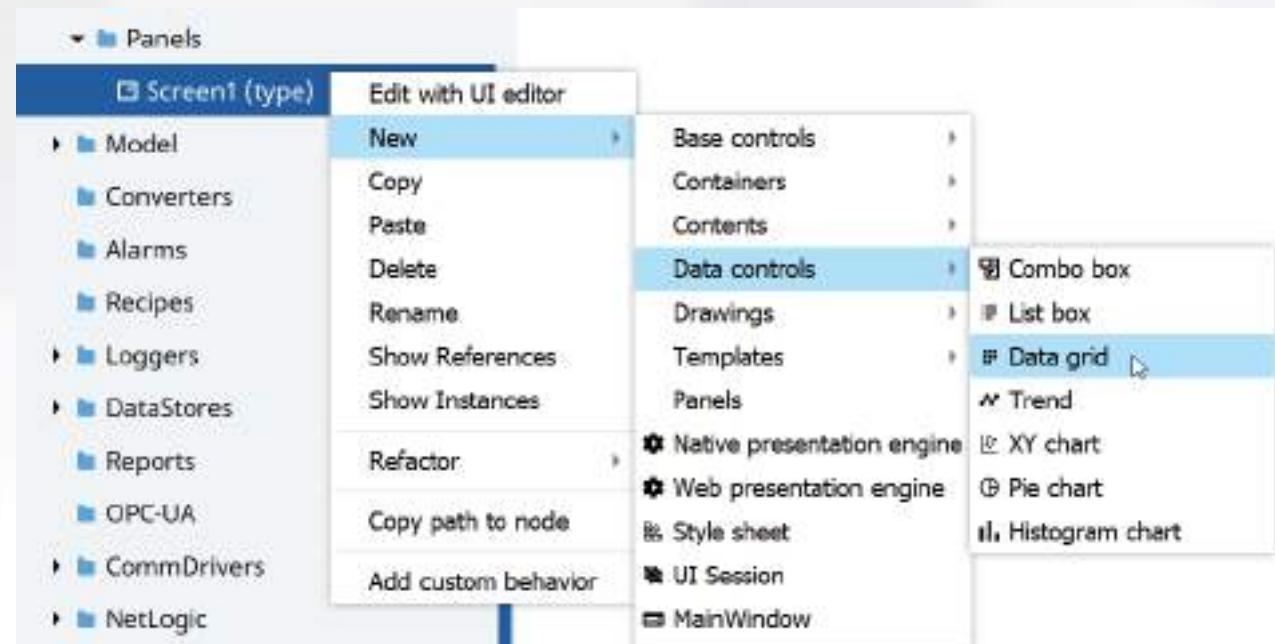
1. Add a "Data grid" object by New > Data controls > Data grid
2. Drag&Drop the Datalogger to the Data grid object

- The object can be customized through Properties pane

- Query
- Columns format
- Sorting

- Export

- Data logger can be exported using the script "Data Logger exporter" available into Template Library. This script can be associated with a button event



# Data logger "viewer"

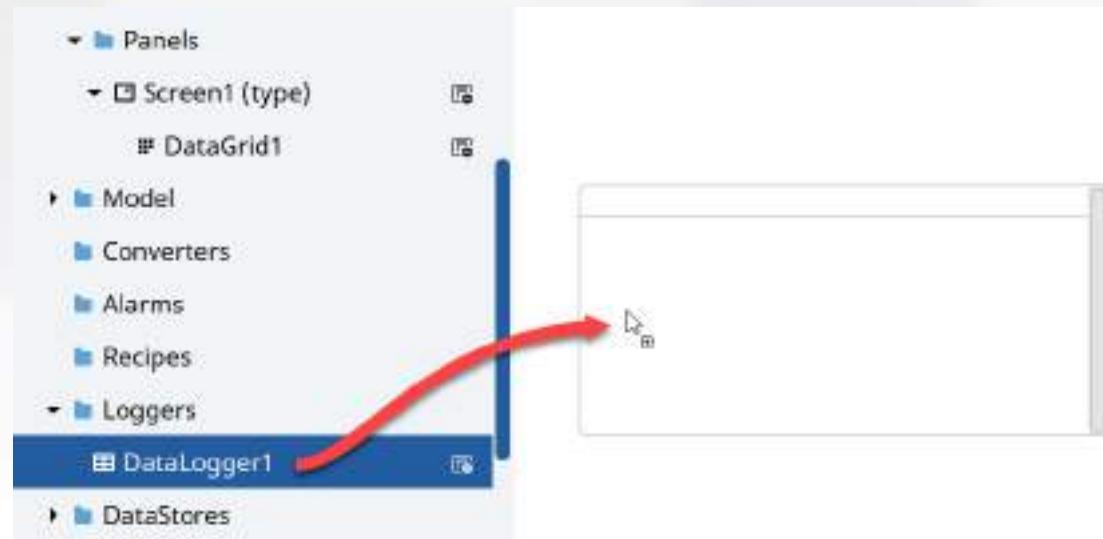
1. Add a "Data grid" object by New > Data controls > Data grid
2. Drag&Drop the Datalogger to the Data grid object

- The object can be customized through Properties pane

- Query
- Columns format
- Sorting

- Export

- Data logger can be exported using the script "Data Logger exporter" available into Template Library. This script can be associated with a button event



# Data logger "viewer"

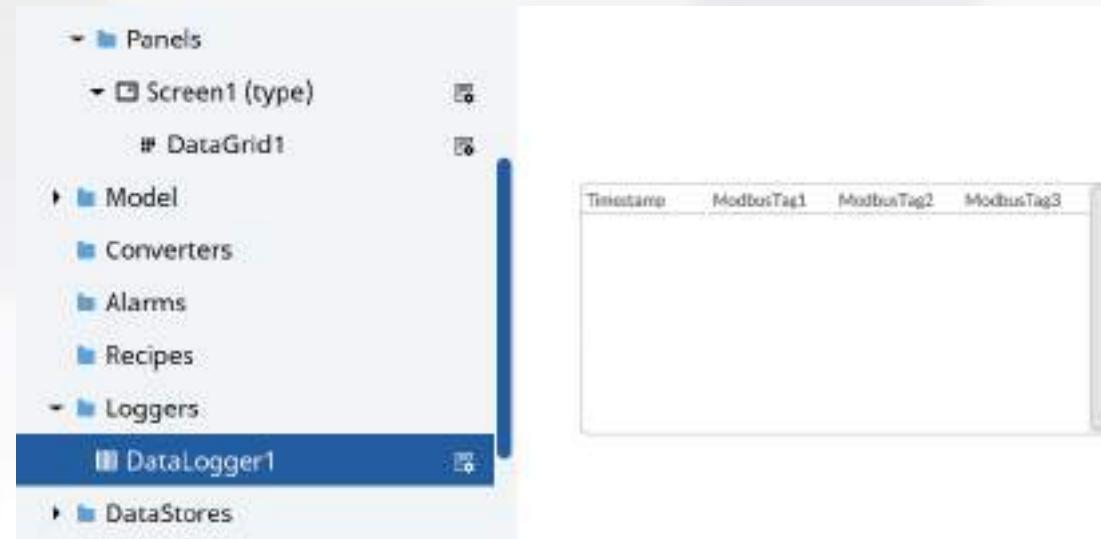
1. Add a "Data grid" object by New > Data controls > Data grid
2. Drag&Drop the Datalogger to the Data grid object

- The object can be customized through Properties pane

- Query
- Columns format
- Sorting

- Export

- Data logger can be exported using the script "Data Logger exporter" available into Template Library. This script can be associated with a button event

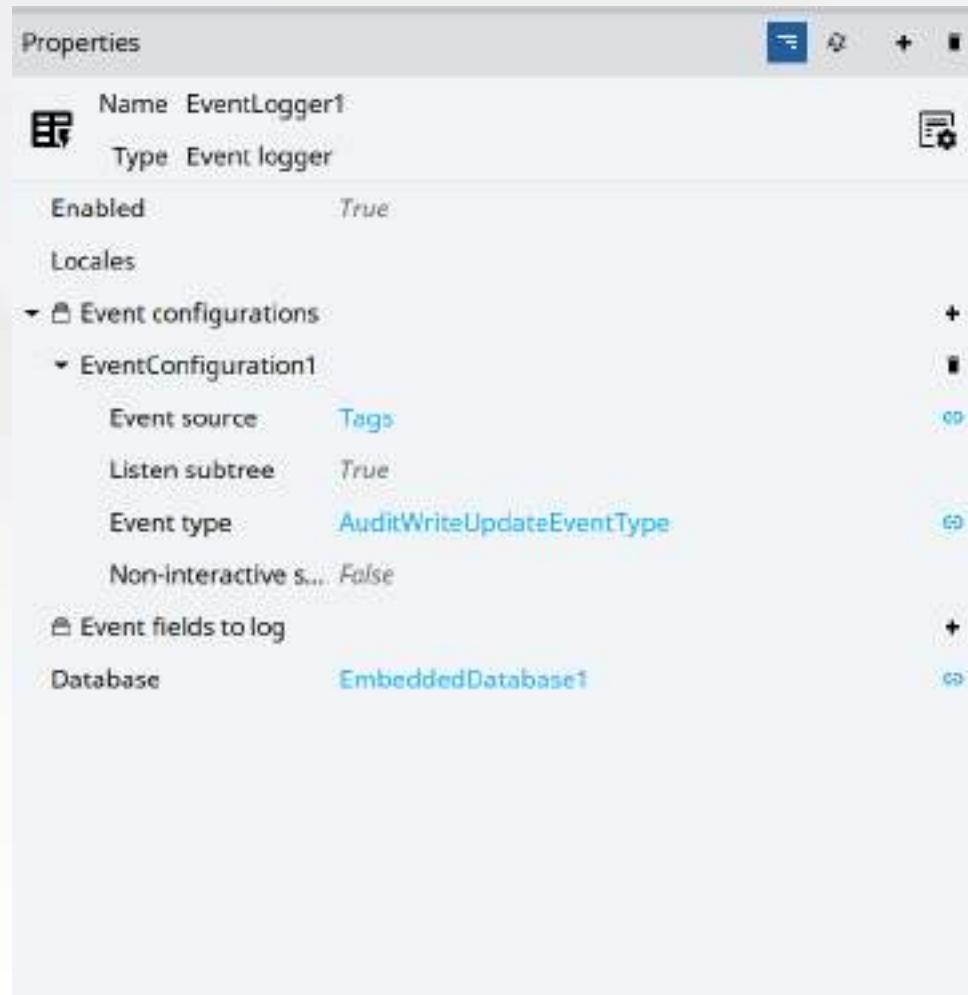


# Hands-on session

- Define a new Datalogger with a periodic sampling of 5 seconds
- Configure a Grid to show the datalogger table stored on the database
- Add a button to manually trigger a new record
- Add a button to refresh the grid
- Add a switch to start/stop the sampling



# Event logger



- At every event the Event Logger creates a new record into the table with structure defined into "Event fields to log"

Time	Client User ID	Source Node	Old Value	New Value
hh:mm:ss	Guest	Variable1	0	5
hh:mm:ss	Admin	Variable2	2	8

- Configure the Source and Event to log
- Select Event fields to log
- Define the Datastore

# Event logger

The screenshot shows the configuration interface for a logger named 'EventLogger1'. The left pane displays a tree structure of event types:

- BaseEventType**: Contains checkboxes for 'EventId' and 'SourceNode' (which is checked).
- AuditEventType**: Contains checkboxes for 'ActionTimeStamp', 'ClientAuditEntryId', 'ClientUserId' (which is checked), 'ServerId', and 'Status'.
- AuditUpdateEventType**: Contains checkboxes for 'AttributeId', 'IndexRange', 'NewValue' (which is checked), and 'OldValue' (which is checked).

- At every event the Event Logger creates a new record into the table with structure defined into "Event fields to log"

Time	Client User ID	Source Node	Old Value	New Value
hh:mm:ss	Guest	Variable1	0	5
hh:mm:ss	Admin	Variable2	2	8

- Configure the Source and Event to log
- Select Event fields to log
- Define the Datastore

# Event logger

The screenshot shows the configuration properties for an 'Event logger' named 'EventLogger1'. Key settings include:

- Enabled:** True
- Locales:** None
- Event configurations:** One configuration named 'EventConfiguration1' is selected:
  - Event source:** Tags
  - Listen subtree:** True
  - Event type:** AuditWriteUpdateEventType
  - Non-interactive s...:** False
- Event fields to log:** A list of fields and their mappings:
  - Time: UtcTime ..../EventArgs/Time
  - ClientUserId: String ..../EventArgs/ClientId
  - SourceNode: String ..../EventArgs/SourceNode@BrowseName
  - OldValue: Int32 ..../EventArgs/OldValue@Value
  - NewValue: Int32 ..../EventArgs/NewValue@Value
- Database:** EmbeddedDatabase1

- At every event the Event Logger creates a new record into the table with structure defined into "Event fields to log"

Time	Client User ID	Source Node	Old Value	New Value
hh:mm:ss	Guest	Variable1	0	5
hh:mm:ss	Admin	Variable2	2	8

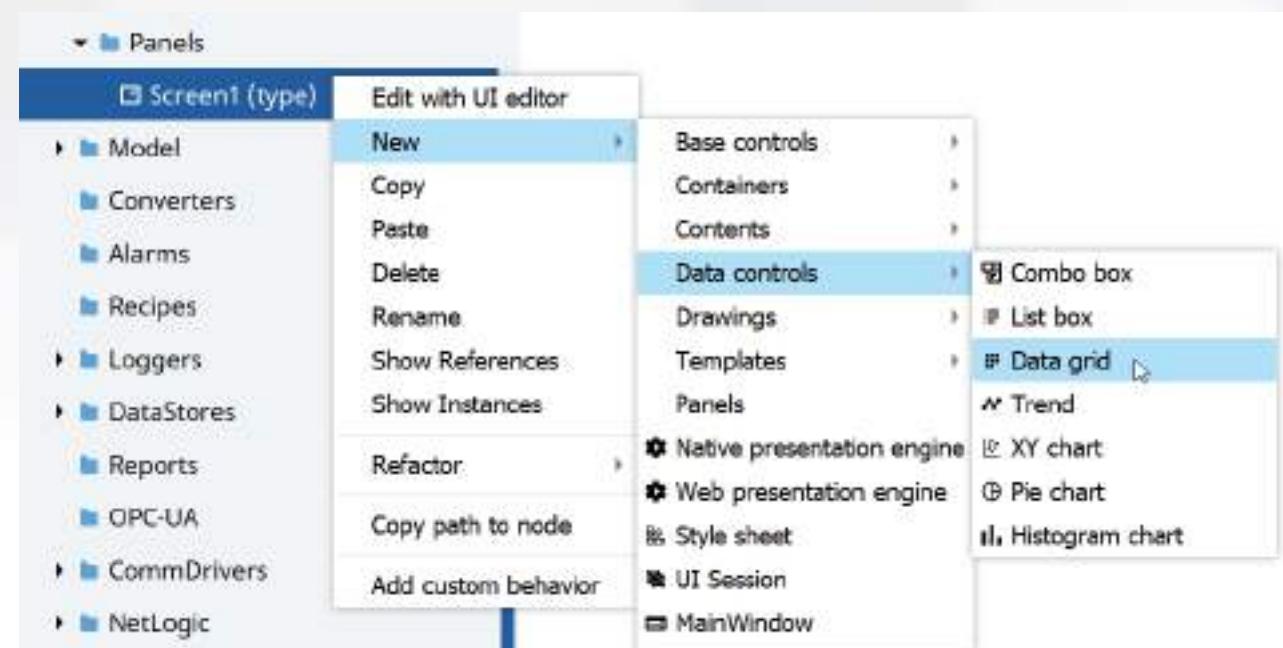
- Configure the Source and Event to log
- Select Event fields to log
- Define the Datastore

# Event logger "viewer"

1. Add a "Data grid" object by New > Data controls > Data grid
2. Drag&Drop the Event Logger to the Data grid object

- The object can be customized through Properties pane

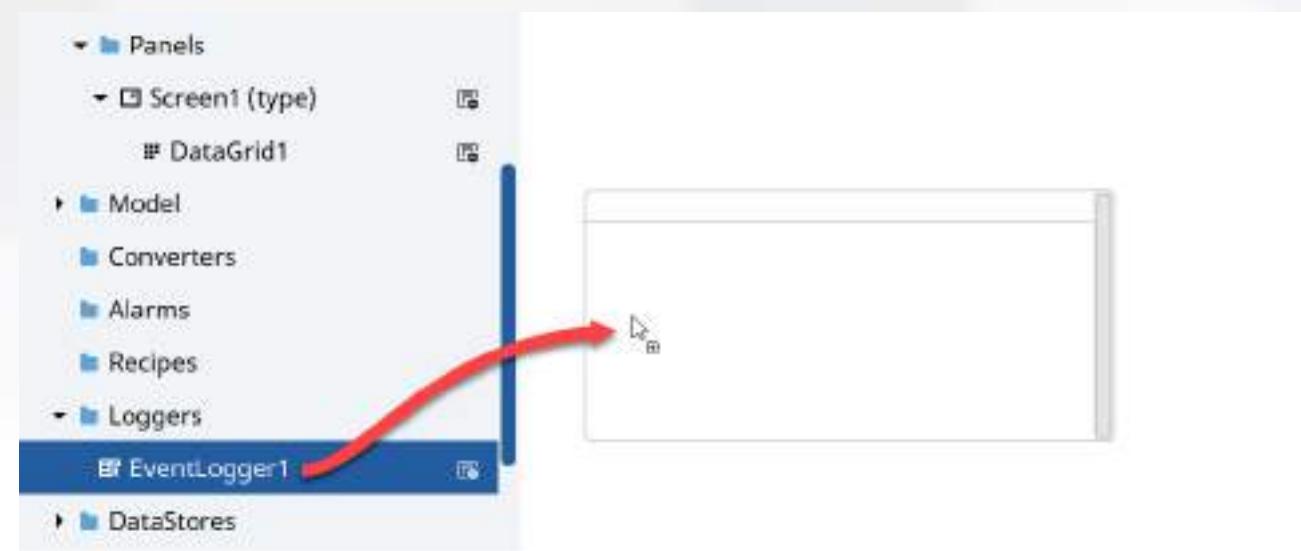
- Query
- Columns format
- Sorting



# Event logger "viewer"

1. Add a "Data grid" object by New > Data controls > Data grid
2. Drag&Drop the Event Logger to the Data grid object

- The object can be customized through Properties pane
  - Query
  - Columns format
  - Sorting

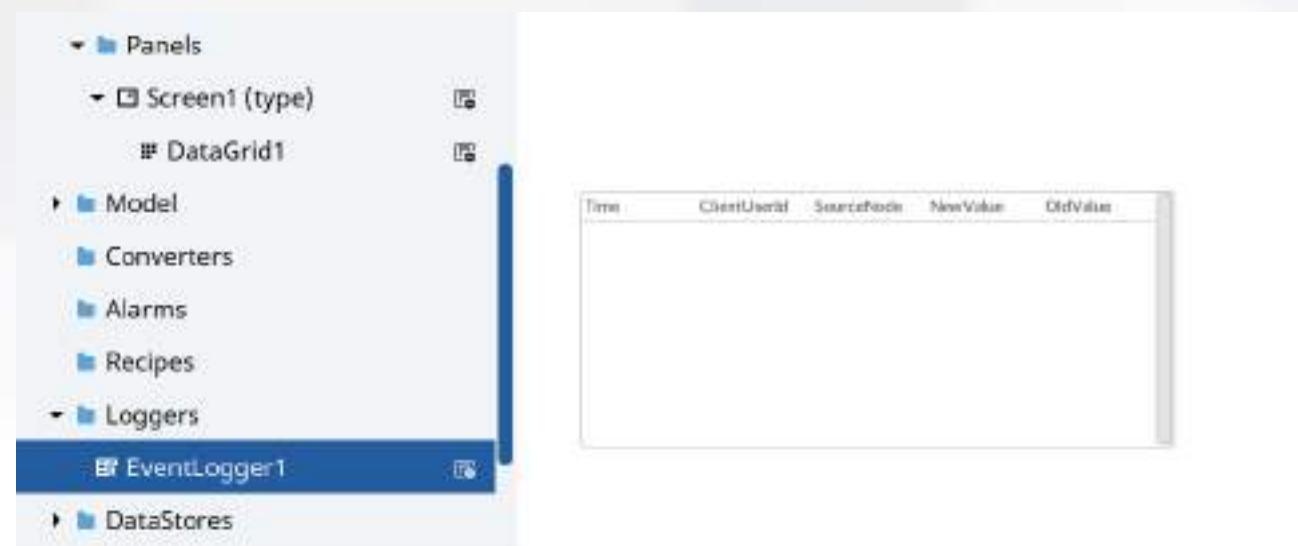


# Event logger "viewer"

1. Add a "Data grid" object by New > Data controls > Data grid
2. Drag&Drop the Event Logger to the Data grid object

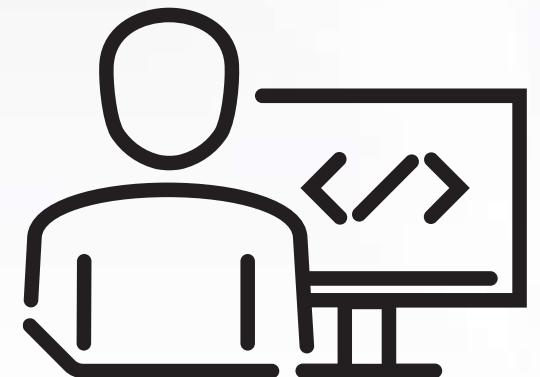
- The object can be customized through Properties pane

- Query
- Columns format
- Sorting



# Hands-on session

- Configure an Event Logger to trace changes made to variables
- Add a Grid to show the Event Logger table
- Add a button to refresh the Grid



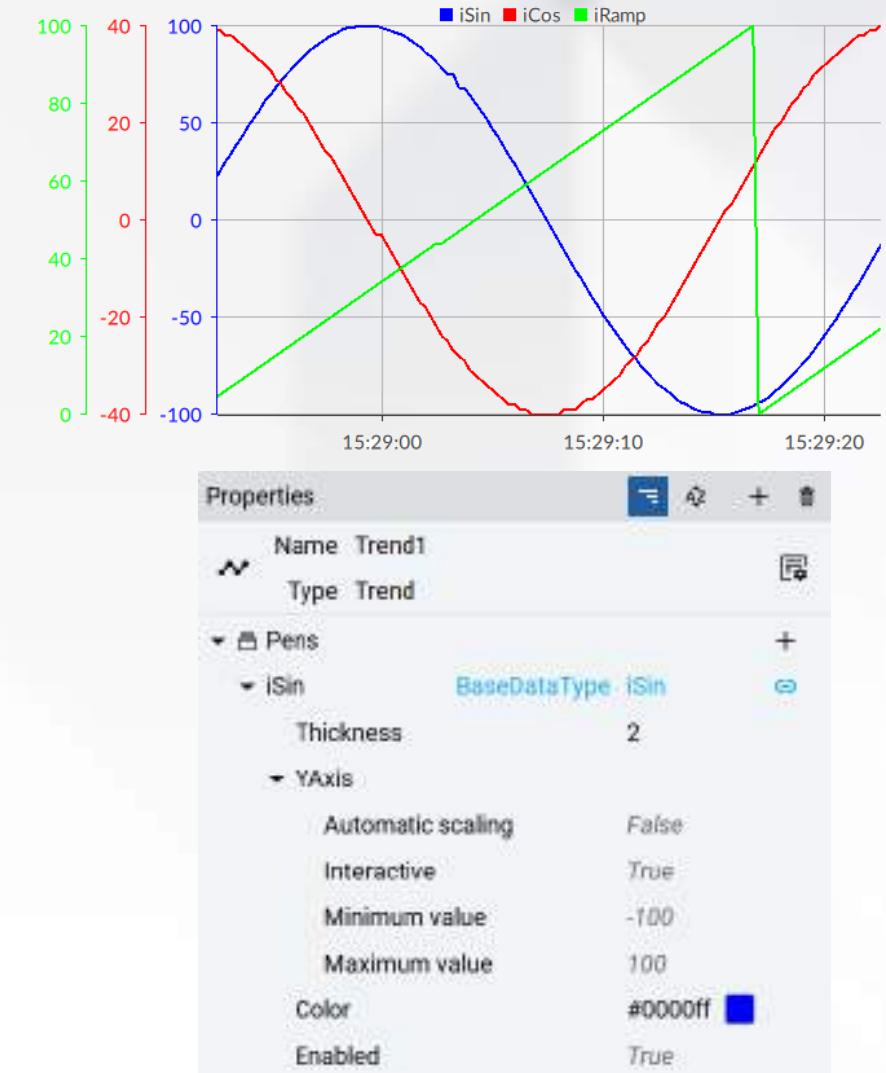


# Represent data by Trends



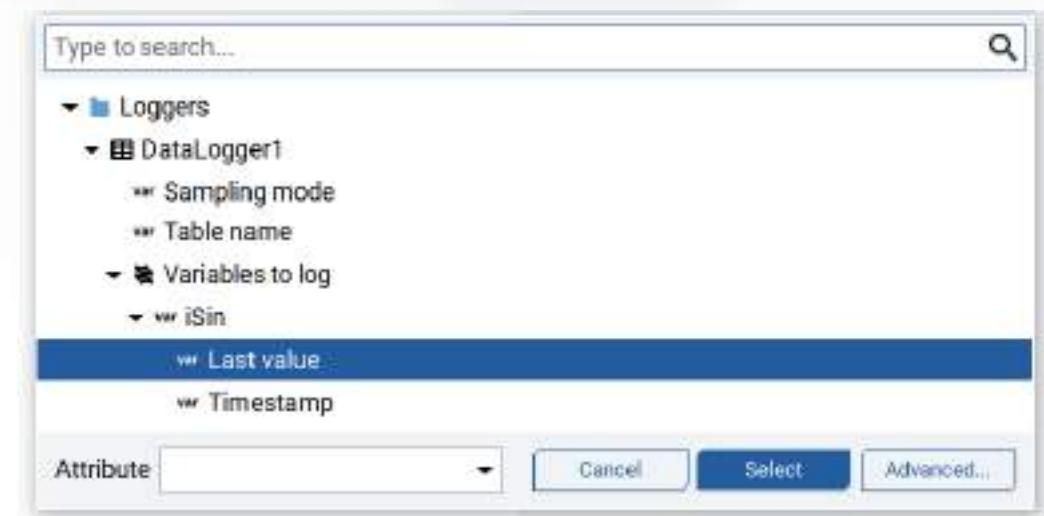
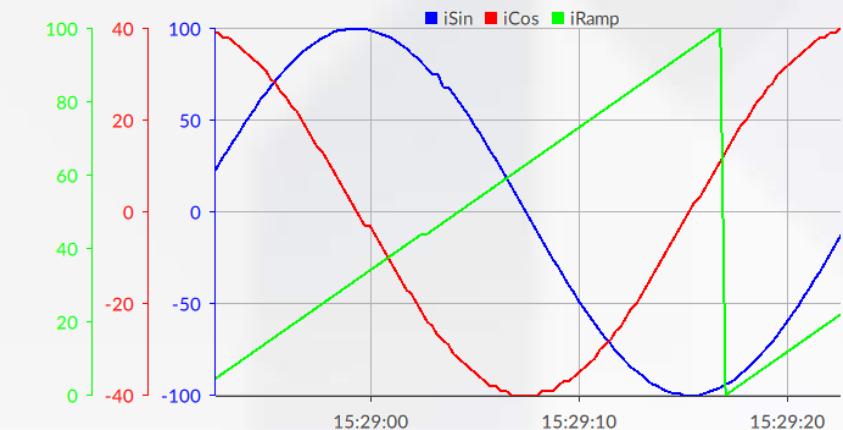
# Real-time Trends

- Trend widget is available under the "Data Controls" group
- Real-time Trend = data showed in the Trend window, will be purged on the exit
- To configure a Trend as a Real-time one, just link a Pen with a Variable
- Optionally, a Y-Axis for each Pen can be configured



# Historical Trends

- Historical Trends = when the Trend window is loaded, a query is executed to retrieve historical data
- To configure a Trend as an Historical one:
  - a DataLogger is needed to save historical data
  - Pens should be associated to  
DataLogger > Variables to Log > {name} > Last Value

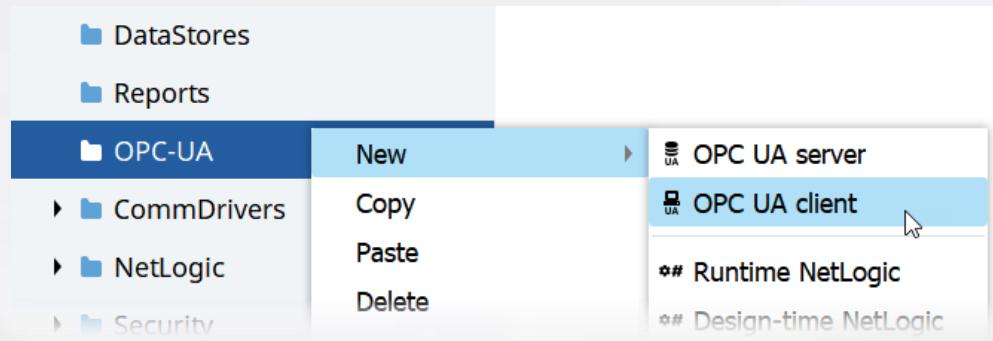


A large, abstract graphic consisting of a series of blue dots arranged in a wave-like, undulating pattern across the entire background of the slide.

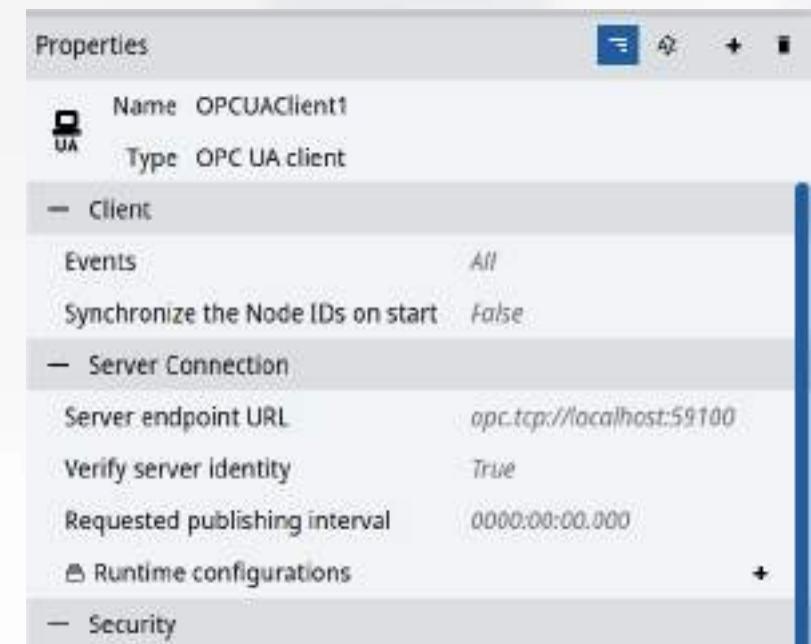
# Exchange data via OPC-UA Client



# OPC-UA Client: basic configuration

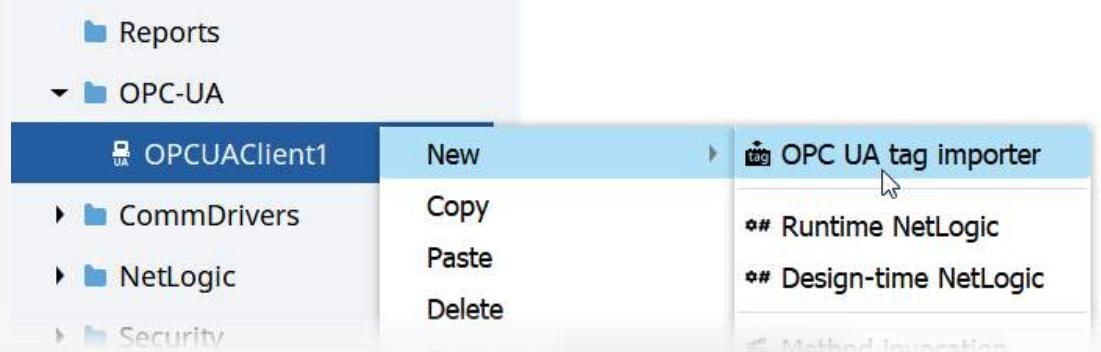


- *Events*: If not interested to read OPC-UA Server Events, set to None
- *Synchronize Node IDs on start*: In case node id of OPC-UA Server changes
- *Server Endpoint URL*: URL of OPC-UA Server to connect to
- *Requested publishing interval*: defines the rate that the OPC-UA Server returns data change notifications to the OPC UA client



# OPC-UA Client: tag importer

- Like any other Tag Importer
- Allow to import/synchronize tags exposed by an OPC-UA Server



OPC-UA > OPCUAClient1 > OPCUATagImporter				Type to search...	Search icon	Filter icon	Sort icons	Clear icon	
	PLC Item	HMI Item	DataType	Status					
<input type="checkbox"/>	▶ Demo			New					
<input type="checkbox"/>	▶ DeviceSet			New					
<input type="checkbox"/>	▶ DeviceTopology			New					
<input type="checkbox"/>	NetworkSet			New					

# Hands-on session

- Launch the OPC UA Server (UaAnsiCServer)
- Add the OPC UA Client node to the project and configure the server endpoint
- Open the OPC UA tag importer and import the folder "Demo > BoilerDemo"
- Into the UI:
  - attach the variable "FillLevel" to a Text box
  - attach the variable "Temperature" to a Text box
  - add a button that calls the method "Fill"  
setting the argument "SetPoint" as variable
  - add a button that calls the method "Heat"  
setting the argument "SetPoint" as variable



# OPC-UA Client: security configuration

- Verify server identity
  - if True, accept only CA-issued certificates
  - if False, accept Self-signed certificates**
- Minimum message security mode
  - Security Profile: none, Sign, Sign+Encrypt
- Minimum security policy
- Client Certificate File
  - "der" file stored in "...\\ProjectFiles\\PKI\\Own\\Client"
- Client Private key file
  - ".pem" file stored in "...\\ProjectFiles\\PKI\\Own\\Client"
- Runtime, **will NOT automatically generate** a self-signed certificate for the Client so it's necessary to create one manually



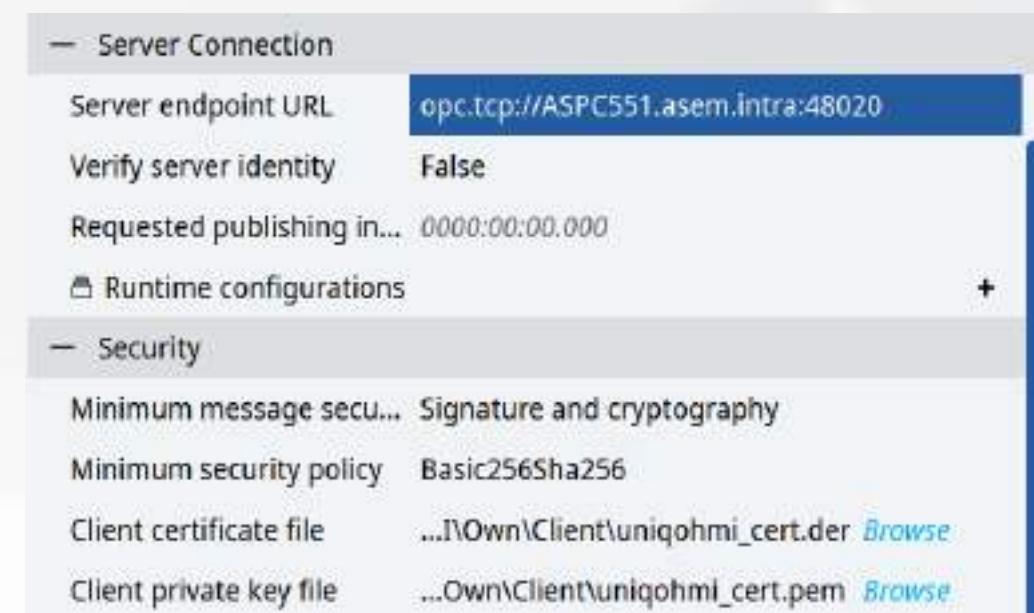
— Client	
Events	All
Synchronize the Node IDs on start	False
— Server Connection	
Server endpoint URL	opc.tcp://localhost:59100
Verify server identity	True
Requested publishing interval	0000:00:00.000
— Runtime configurations	
— Security	
Minimum message security mode	None
Minimum security policy	None
Client certificate file	<a href="#">Browse</a>
Client private key file	<a href="#">Browse</a>

# OPC-UA Client: connection using certificates

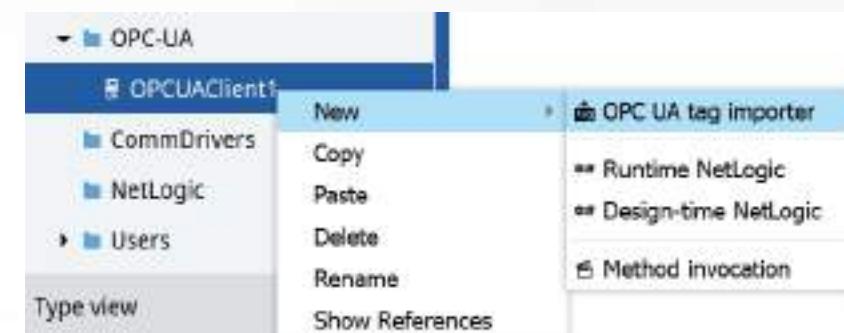
- Connection to a 3rd part OPC-UA Server

## 1. Configure OPC-UA Client Settings:

- Server endpoint URL
- Set "Verify server identity" to False  
if OPC-UA Server uses a self-signed certificate
- Set Security Profile and Security Policy
- Import self-signed Client Certificate file  
and Client Private Key file



## 2. Import tags adding the object OPC UA tag importer



# OPC-UA Client: connection using certificates

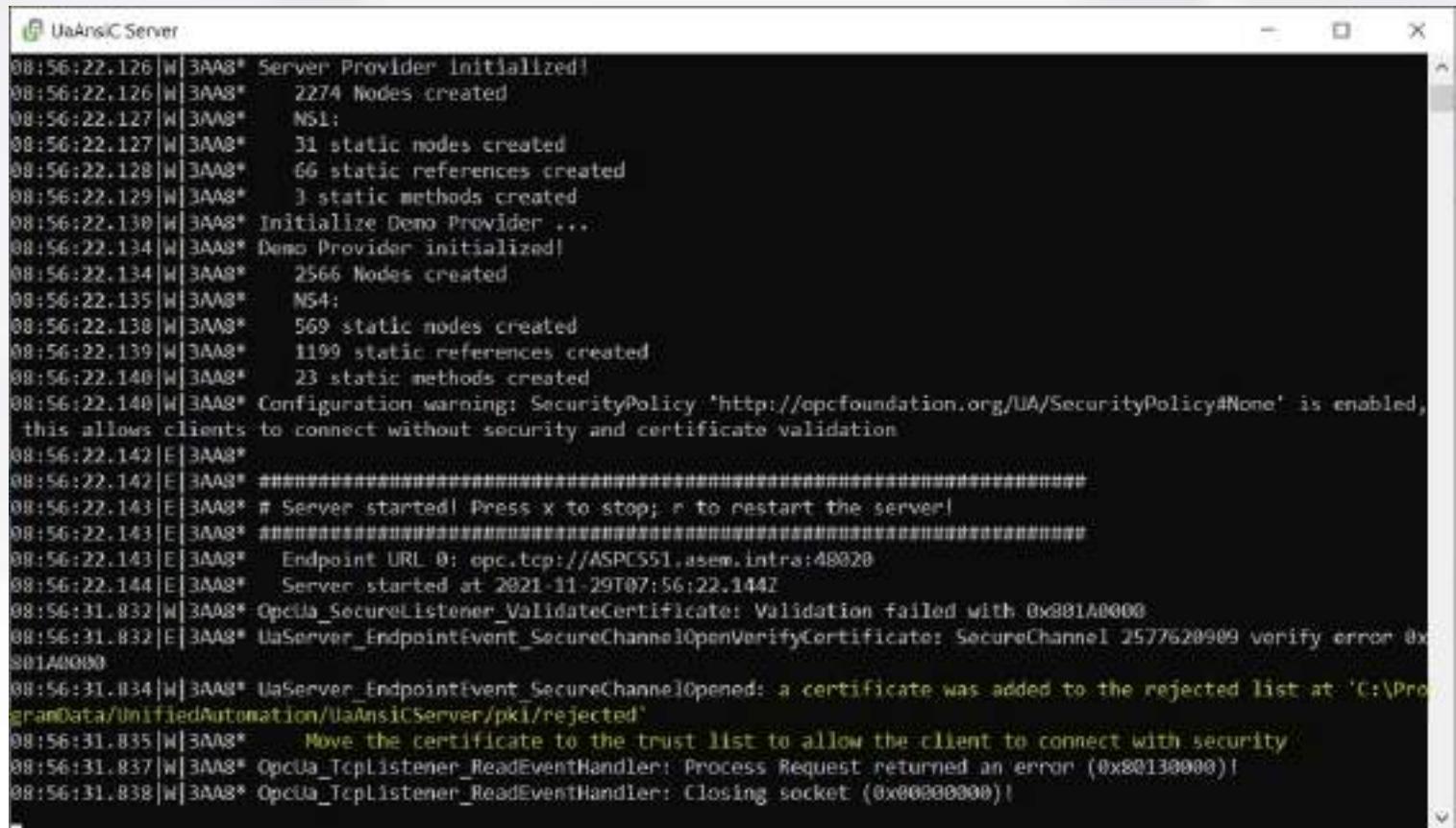
- OPC-UA Client sends his certificate to the Server,  
but in this case, the Server automatically set the certificate into the Rejected folder so  
**"BadSecurityChecksFailed"** is triggered

Code	Timestamp	Category	Object	Message
43	2021-11-29 08:54:31.967	TaskConsumer		Exception caught: Connection failed with error BadSecurityChecksFailed

Q Studio Output

# OPC-UA Client: connection using certificates

- In this case,  
we need to go to the  
OPC-UA Server and  
manually move  
the Client certificate  
from Rejected  
to Trusted folder
- After this operation,  
the Client will be able to  
connect to Server to Browse  
tag for import and  
exchange data.



```
UaAmsiC Server
08:56:22.126|W|3AA8* Server Provider initialized
08:56:22.126|W|3AA8* 2274 Nodes created
08:56:22.127|W|3AA8* NS1:
08:56:22.127|W|3AA8* 31 static nodes created
08:56:22.128|W|3AA8* 66 static references created
08:56:22.129|W|3AA8* 3 static methods created
08:56:22.130|W|3AA8* Initialize Demo Provider ...
08:56:22.134|W|3AA8* Demo Provider initialized
08:56:22.134|W|3AA8* 2566 Nodes created
08:56:22.135|W|3AA8* NS4:
08:56:22.138|W|3AA8* 569 static nodes created
08:56:22.139|W|3AA8* 1199 static references created
08:56:22.140|W|3AA8* 23 static methods created
08:56:22.140|W|3AA8* Configuration warning: SecurityPolicy 'http://opcfoundation.org/UA/SecurityPolicy#None' is enabled,
this allows clients to connect without security and certificate validation
08:56:22.142|E|3AA8*
08:56:22.142|E|3AA8* #####
08:56:22.143|E|3AA8* # Server started! Press x to stop; r to restart the server!
08:56:22.143|E|3AA8* #####
08:56:22.143|E|3AA8* Endpoint URL 0: opc.tcp://ASPCS51.asem.intra:48928
08:56:22.144|E|3AA8* Server started at 2021-11-29T07:56:22.144Z
08:56:31.832|W|3AA8* OpcUa_SecureListener_ValidateCertificate: Validation failed with 0x801A0000
08:56:31.832|E|3AA8* UaServer_EndpointEvent_SecureChannelOpenVerifyCertificate: SecureChannel 2577628909 verify error 0x
801A0000
08:56:31.834|W|3AA8* UaServer_EndpointEvent_SecureChannelOpened: a certificate was added to the rejected list at 'C:\Pro
gramData\UnifiedAutomation\UaAmsiCServer\pki\rejected'
08:56:31.835|W|3AA8* Move the certificate to the trust list to allow the client to connect with security
08:56:31.837|W|3AA8* OpcUa_TcpListener_ReadEventHandler: Process Request returned an error (0x80130000) !
08:56:31.838|W|3AA8* OpcUa_TcpListener_ReadEventHandler: Closing socket (0x00000000)!
```

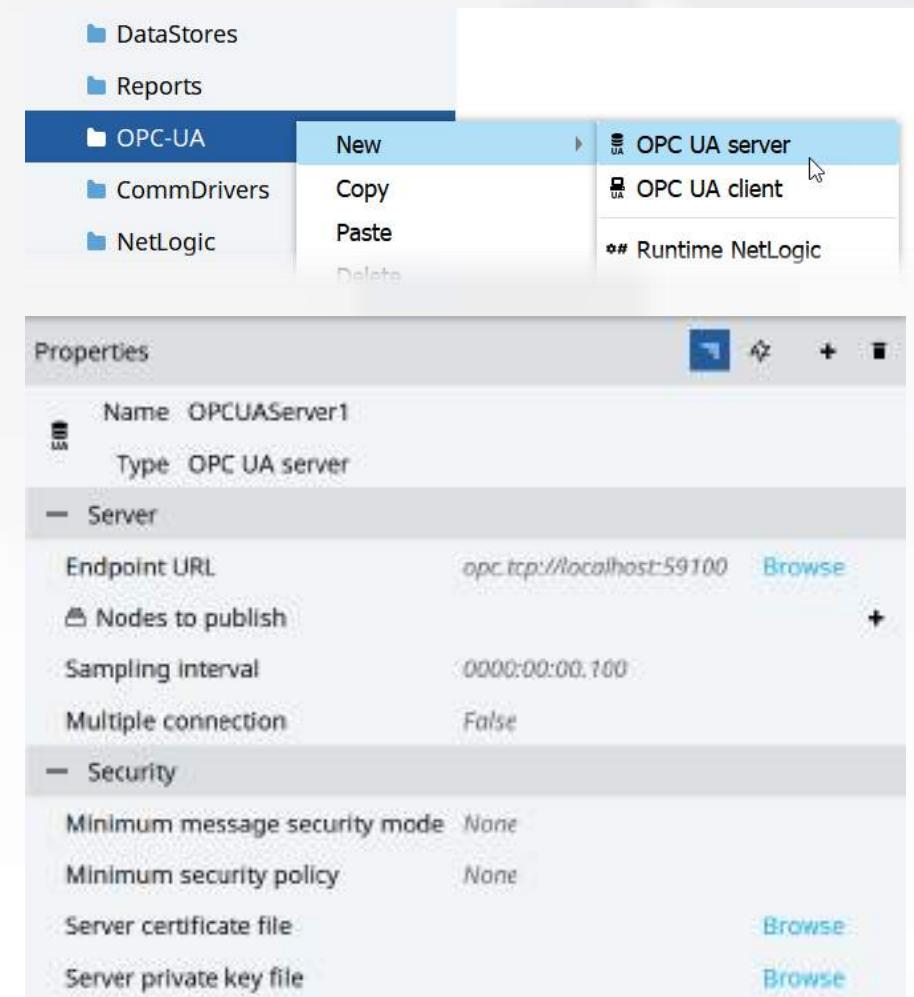


# Expose data as OPC-UA Server



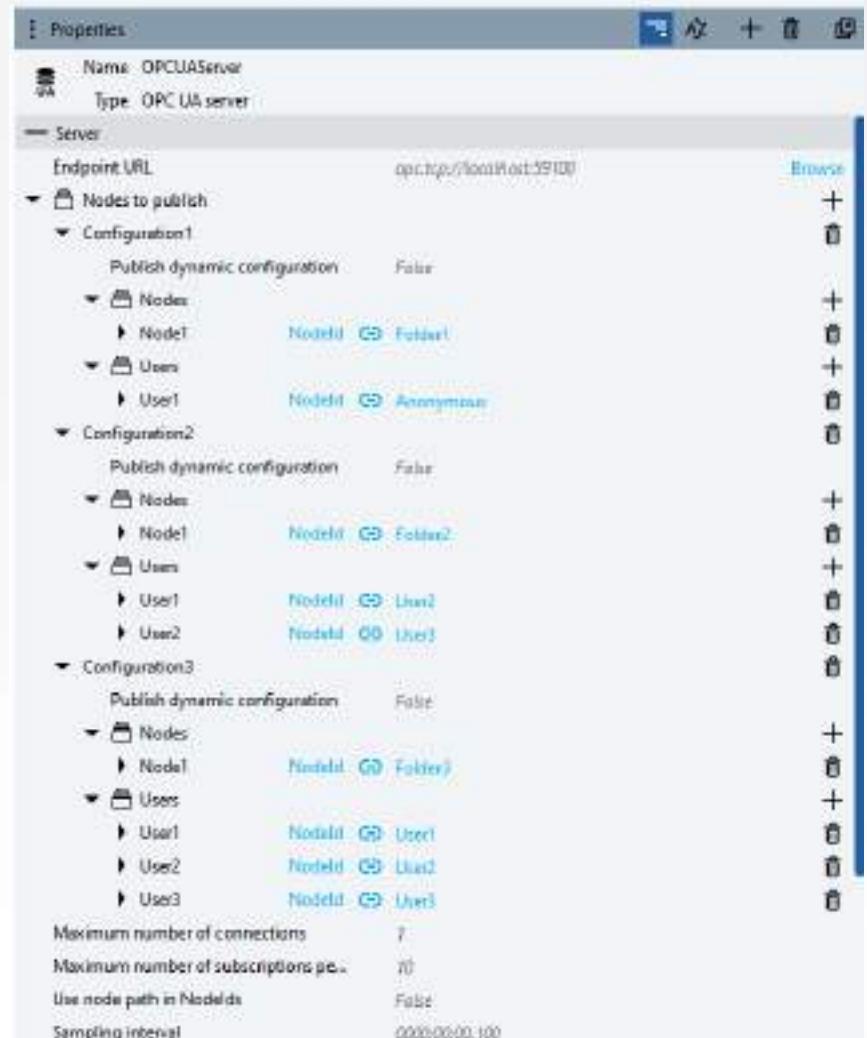
# OPC-UA Server: basic configuration

- *Endpoint URL*
  - protocol, IP/hostname, port used by Server
- *Nodes to publish*
  - by default, the entire project will be published
- *Sampling Interval*
  - Defines a “best effort” cyclic rate that the Server uses to sample the item from its source.  
i.e. the Communication Driver tag polling rate
- *Multiple connections*
  - Allow multiple OPC-UA Clients to connect  
then OPC-UA Server cost 2 tokens instead of 1
- *Security settings*
  - See: [Appendix: See the light about OPC-UA Security](#)
  - See: [Appendix: Using OPC-UA certificates](#)
  - See: [Appendix: Generating OPC-UA certificates](#)



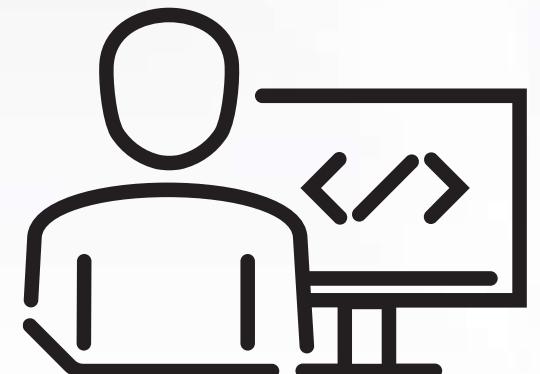
# OPC-UA Server: Nodes to publish

- Specific users can be limited to specific nodes of the project
- When configuring a set of nodes to publish, a user entry must be provided (can be any project user or “Anonymous”)
- Multiple nodes can be assigned to multiple users
- Currently no way to set a node in “read-only” mode
- If no entry is created in the **Nodes to publish** collection, then the whole project is exposed via OPC-UA



# Hands-on session

- Add the OPC UA Server node to the project
- Configure the OPC UA Server to expose the entire project
- Launch the Emulator
- Launch the OPC UA Client «UA Expert» and connect to the Emulator
- Browse some variables or call a method exposed like the "Log" method of a Datalogger

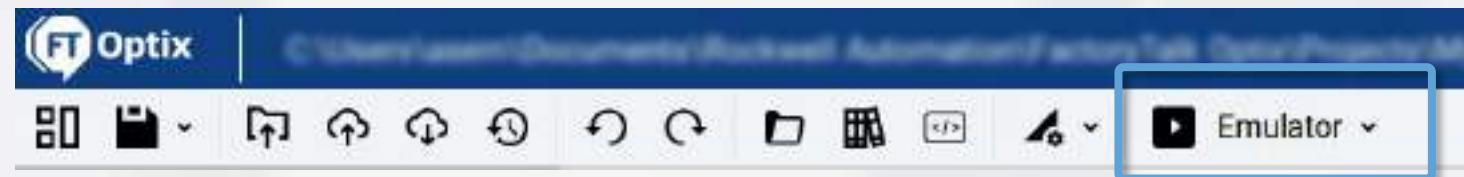


# Project deployment



# Execute the project with emulator

- Allow the execution of the project from Studio.



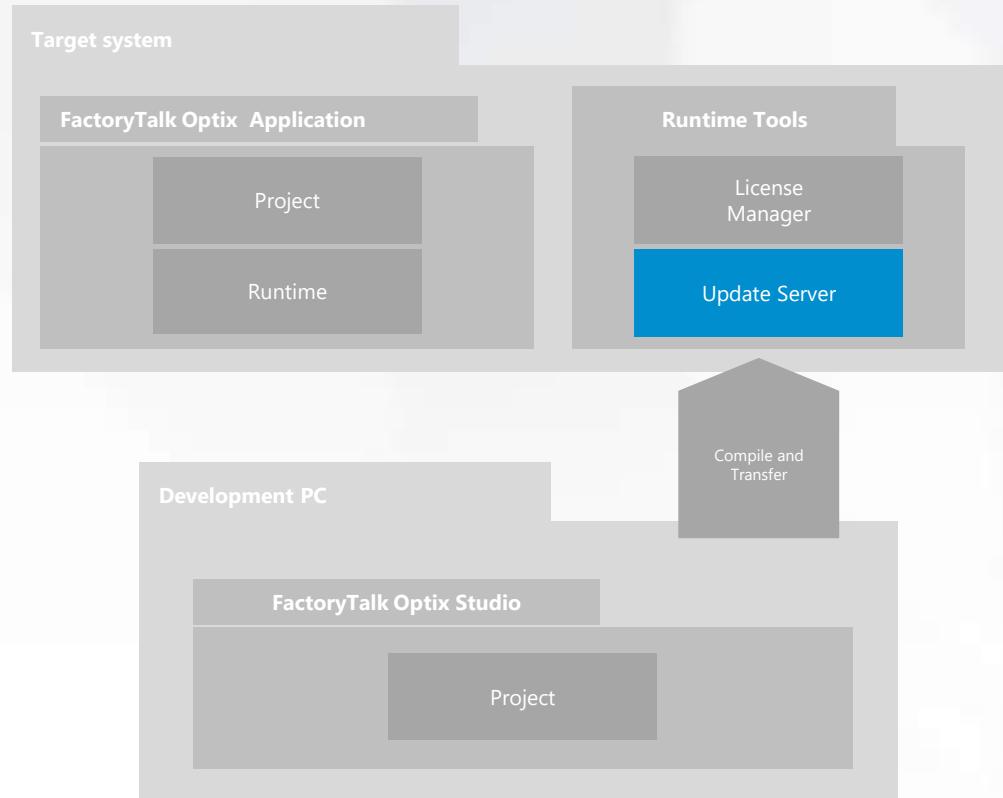
- Emulator folder: %localappdata%\Rockwell Automation\FactoryTalk Optix\Emulator\
- Emulator output is useful to check license token in use by the Runtime

Code	Timestamp	Category	Object	Message
①	2022-09-27 18:19:56.572	FTOptixRuntime		Starting project MyProject
①	2022-09-27 18:19:56.572	FTOptixRuntime		FTOptixRuntime is currently using 3 license token(s) ↴ Component: Native pres... ↴ Component: Retentivity storage consu...
①	2022-09-27 18:19:56.572	FTOptixRuntime		No license tokens found ↴ FTOptixRuntime will be closed in: 120 minutes

# Transfer the project to local target

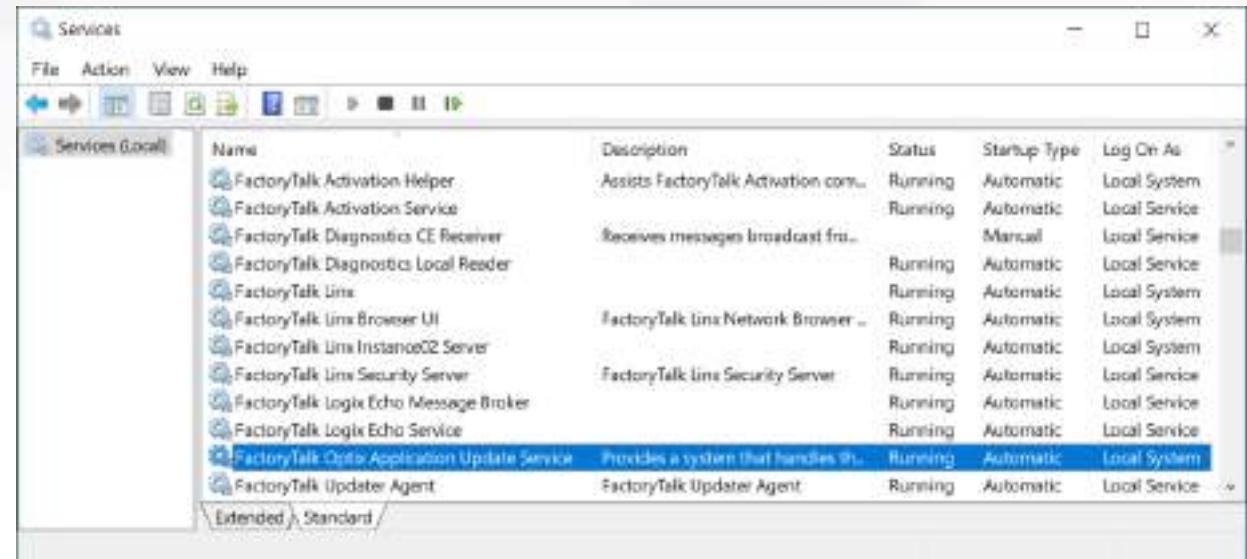
1. Target must have "Runtime Tools" installed and "Update Server" running

- **Optix Panel:** Runtime Tools are already installed in production and Update the Server running as a service
- **iPC:** Runtime Tools must be installed manually. Update Server will be installed as Service and set to automatically startup when Windows starts



# Transfer the project to local target

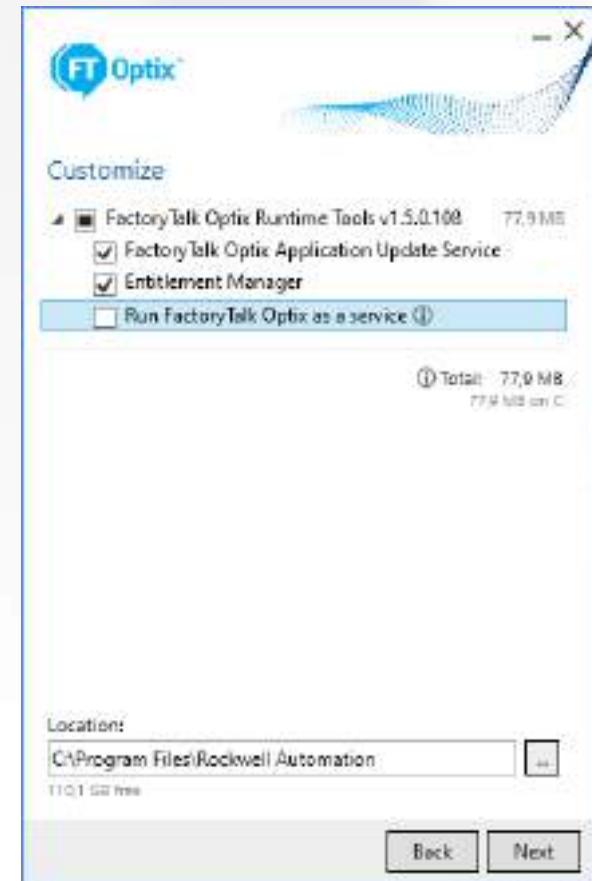
1. Target must have "Runtime Tools" installed and "Update Server" running
  - **Optix Panel:** Runtime Tools are already installed in production and Update the Server running as a service
  - **iPC:** Runtime Tools must be installed manually. Update Server will be installed as Service and set to automatically startup when Windows starts
- On **OptixPanels**, the UpdateServer is automatically upgraded by FactoryTalk Optix Studio if needed (downgrade not supported)
- On **iPC**, the UpdateServer can be upgraded or downgraded by the user as needed
- FactoryTalk Optix version must be **same or newer** version than the UpdateServer version (no backward compatibility)



Services						
	Name	Description	Status	Startup Type	Log On As	More
	FactoryTalk Activation Helper	Assists FactoryTalk Activation com...	Running	Automatic	Local System	
	FactoryTalk Activation Service		Running	Automatic	Local Service	
	FactoryTalk Diagnostics CE Receiver	Receives messages broadcast from...	Running	Manual	Local Service	
	FactoryTalk Diagnostics Local Reader		Running	Automatic	Local Service	
	FactoryTalk Lims		Running	Automatic	Local System	
	FactoryTalk Lims Browser UI	FactoryTalk Lims Network Browser ...	Running	Automatic	Local System	
	FactoryTalk Lims Instance02 Server		Running	Automatic	Local System	
	FactoryTalk Lims Security Server	FactoryTalk Lims Security Server	Running	Automatic	Local Service	
	FactoryTalk Logix Echo Message Broker		Running	Automatic	Local Service	
	FactoryTalk Logix Echo Service		Running	Automatic	Local Service	
	FactoryTalk Orts Application Update Service	Provides a system that handles th...	Running	Automatic	Local System	
	FactoryTalk Updater Agent	FactoryTalk Updater Agent	Running	Automatic	Local Service	

# Run the project as Windows service

- Configure the RuntimeTools in Service mode during setup
  - Only available on Windows IPCs
  - Project cannot contain a NativePresentationEngine
  - WebPresentationEngine is allowed but not strictly needed
  - Useful when the application does not have any UI (data gateway) or when the user interface should only be accessed remotely



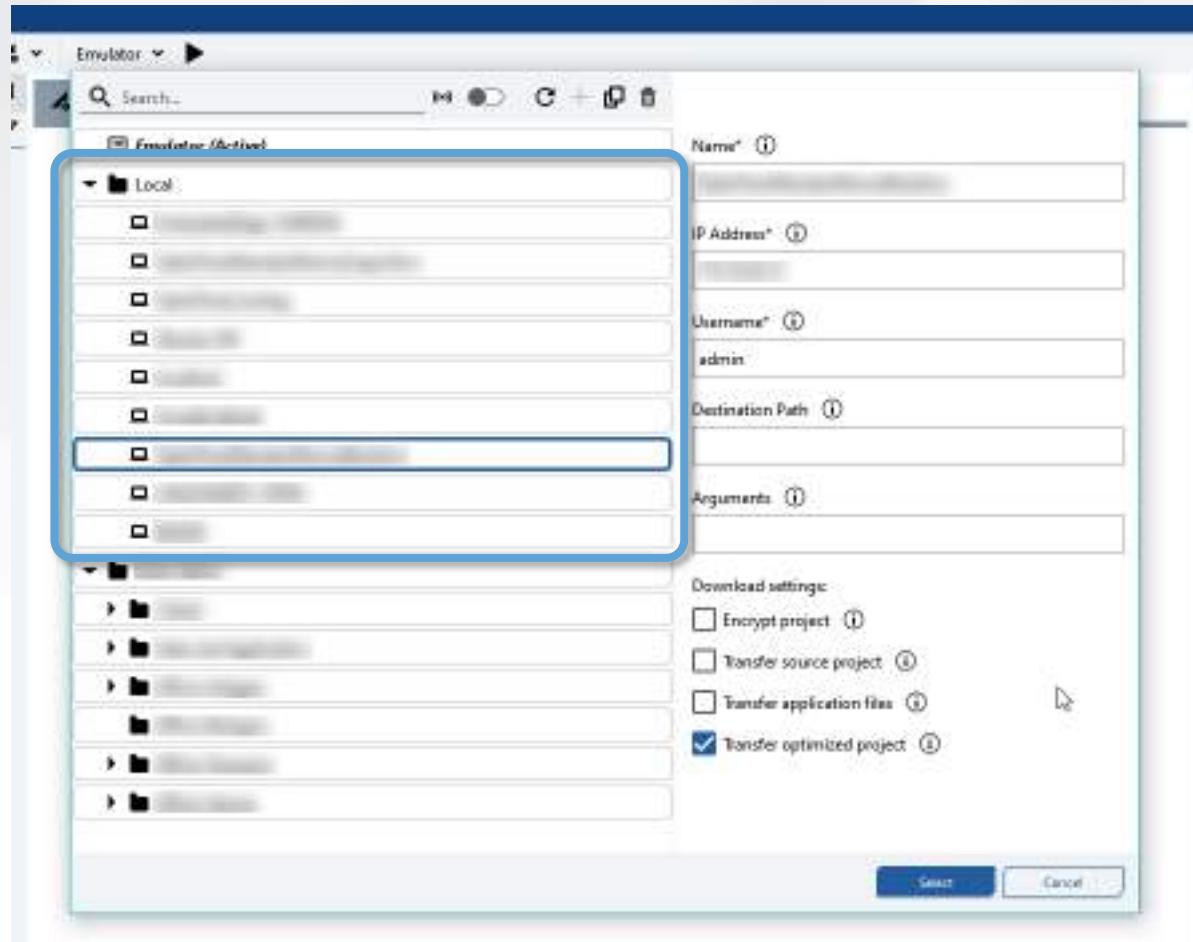
# Transfer the project to local target

## 2. Configure Target device in Studio

- Name,
- IP Address,
- Username,
  - for Optix Panels, it's the same admin user that access the System Manager
  - for iPC, it's a Windows user with rights to access OS folders
- Destination Path: optional

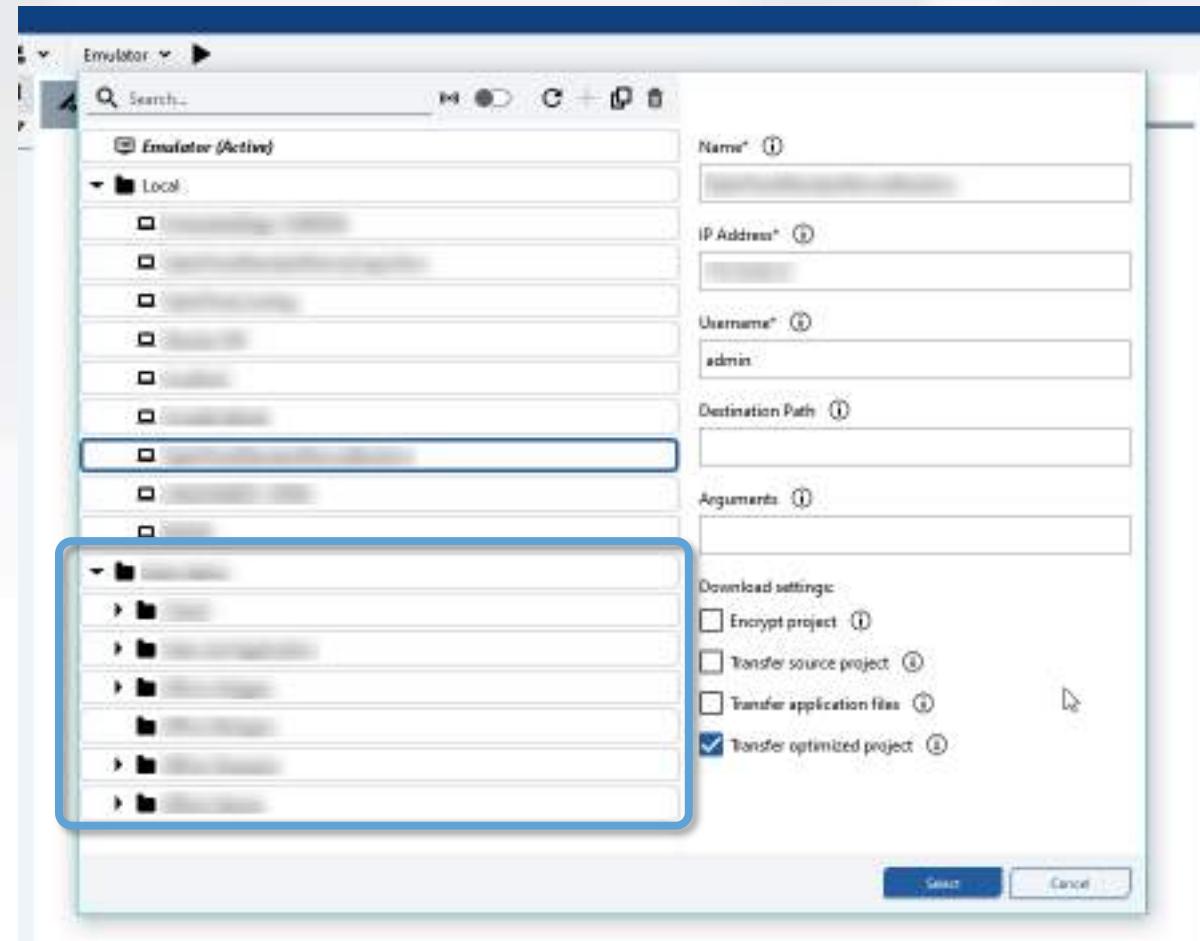
## 3. Transfer from Studio to Target

- set the Active path
- click the Play button



# Transfer the project to remote target (FactoryTalk Remote Access)

- If you are signed in FactoryTalk Hub and you have also access to a FactoryTalk Remote Access domain
- FactoryTalk Optix Studio, lists the devices available through FactoryTalk Remote Access
- Then you will be able to:
  - activate a VPN connection to the device and transfer the project
  - open the FTRA screen mirroring feature (VNC-like)



# Execute the project on target

4. Application is launched by Studio after download

- **HMI:** Application is also set to be executed automatically at HMI startup
- **iPC:** Application is also set to be automatically executed at Windows startup by using the shortcut of

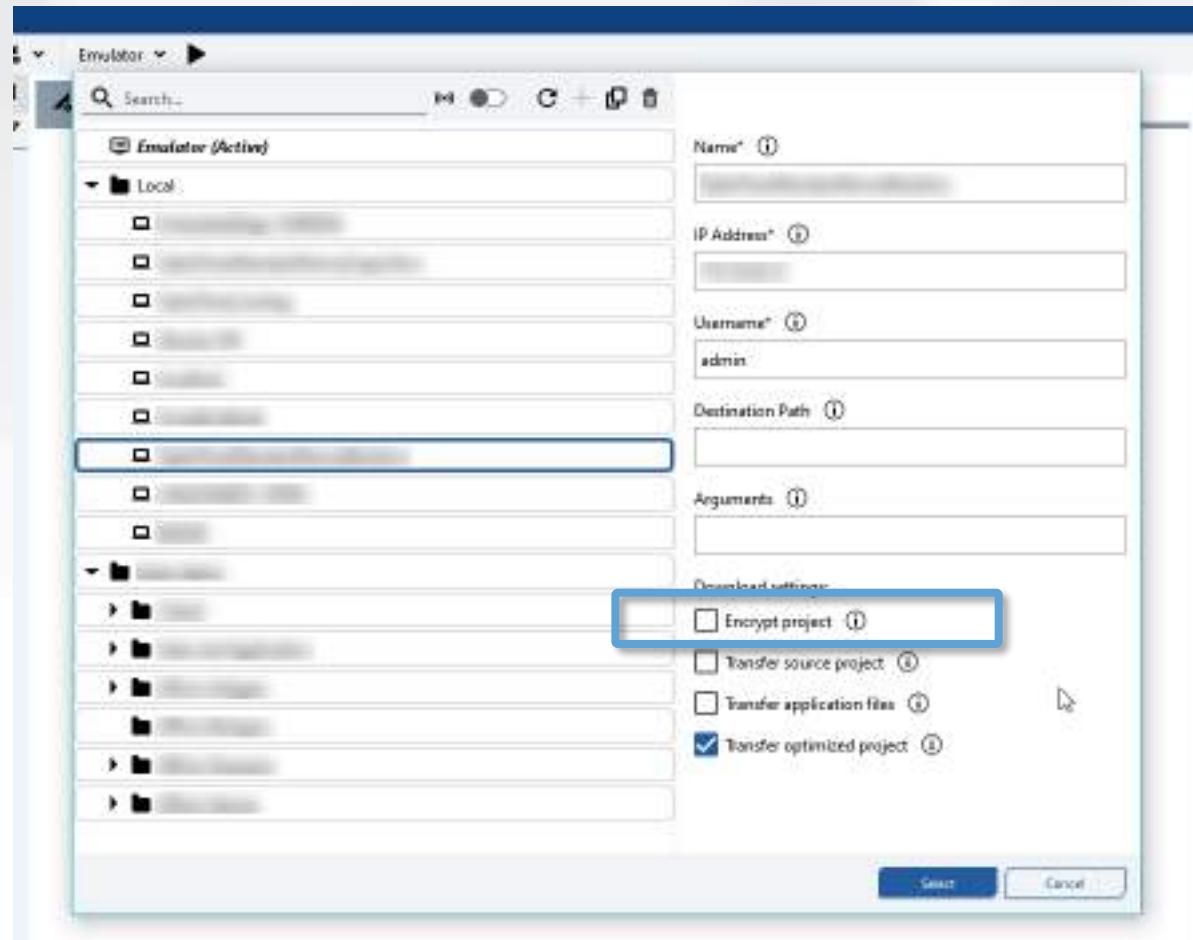
```
%localappdata%\Rockwell Automation\FactoryTalk  
Optix\Application\FTOptixRuntime.exe
```

into:

```
shell:startup
```

# Deployment options

- Project can be encrypted with a hardware-based key
  - All YAML files will be fully encrypted using a unique and hardware-based encryption key
  - Once project is encrypted, it cannot be moved to a different PC or Panel (as the encryption is hardware based)
- Project is encrypted and decrypted using the UpdateServer
  - Encryption is only supported when deploying the project from FactoryTalk Optix Studio
  - Project can be decrypted and uploaded (if source files are transferred) only by using the Update Server

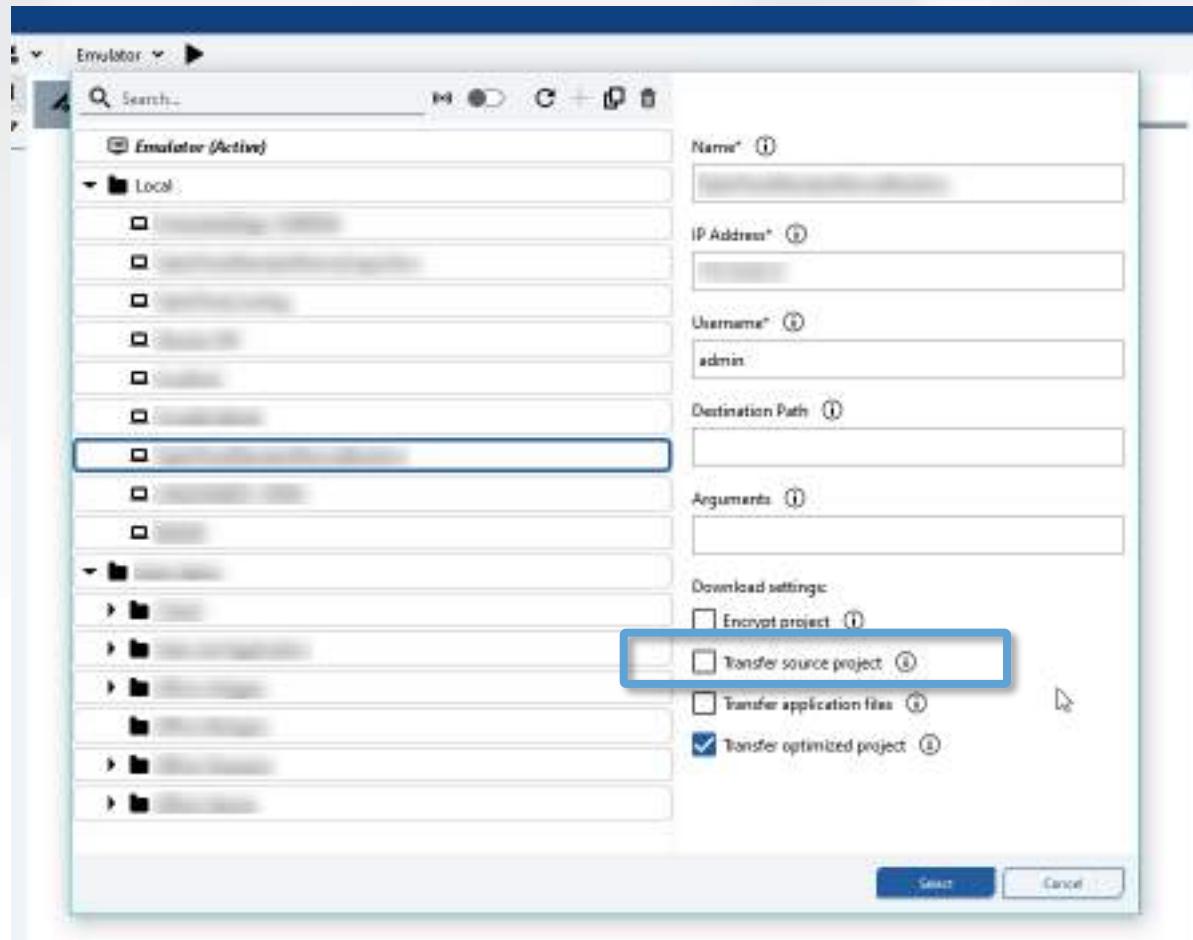


# Deployment options

- Project can be encrypted with a hardware-based key
    - All YAML files will be fully encrypted using a unique and hardware-based encryption key
    - Once project is encrypted, it cannot be moved to a different PC or Panel (as the encryption is hardware based)
    - Project is encrypted and decrypted using the UpdateServer
      - Encryption is only supported when deploying the project from FactoryTalk Optix Studio
      - Project can be decrypted and uploaded (if source files are transferred) only by using the Update Server

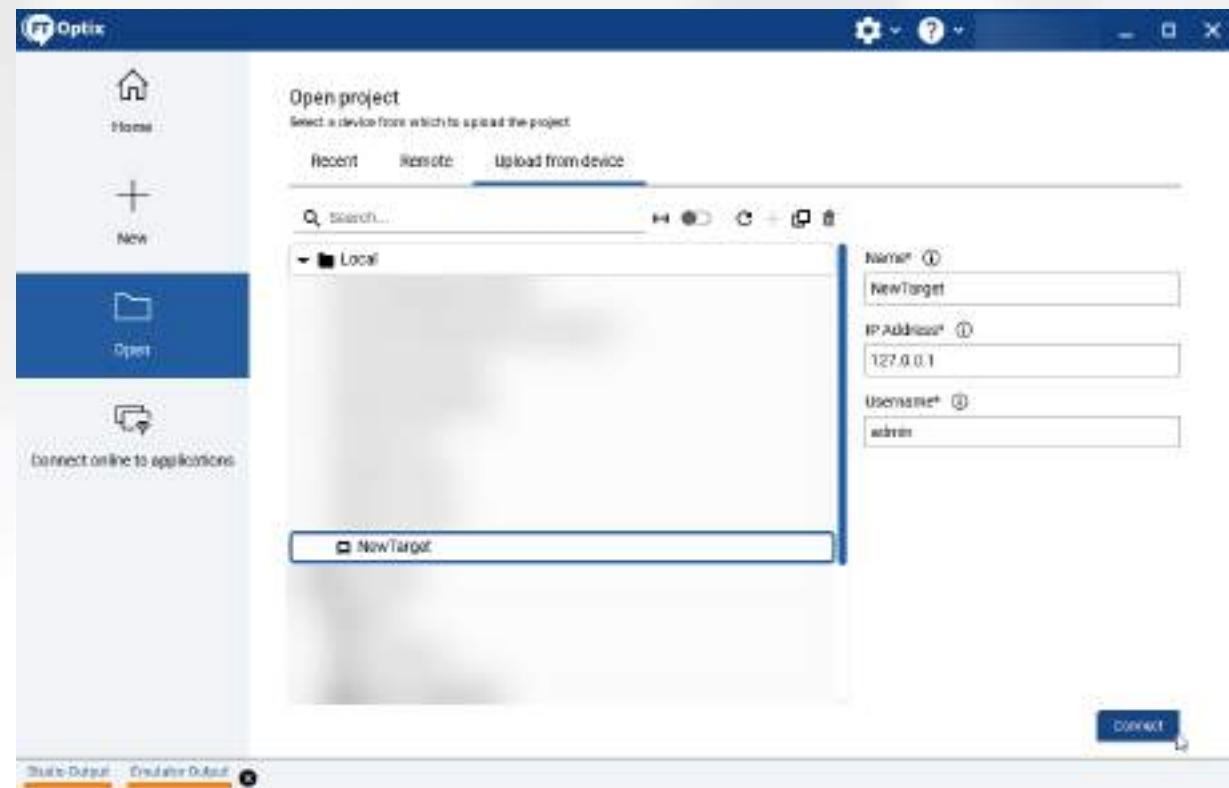
# Deployment options

- Source code of the application can be deployed to the device
  - This allows the project to be uploaded from the target device if needed



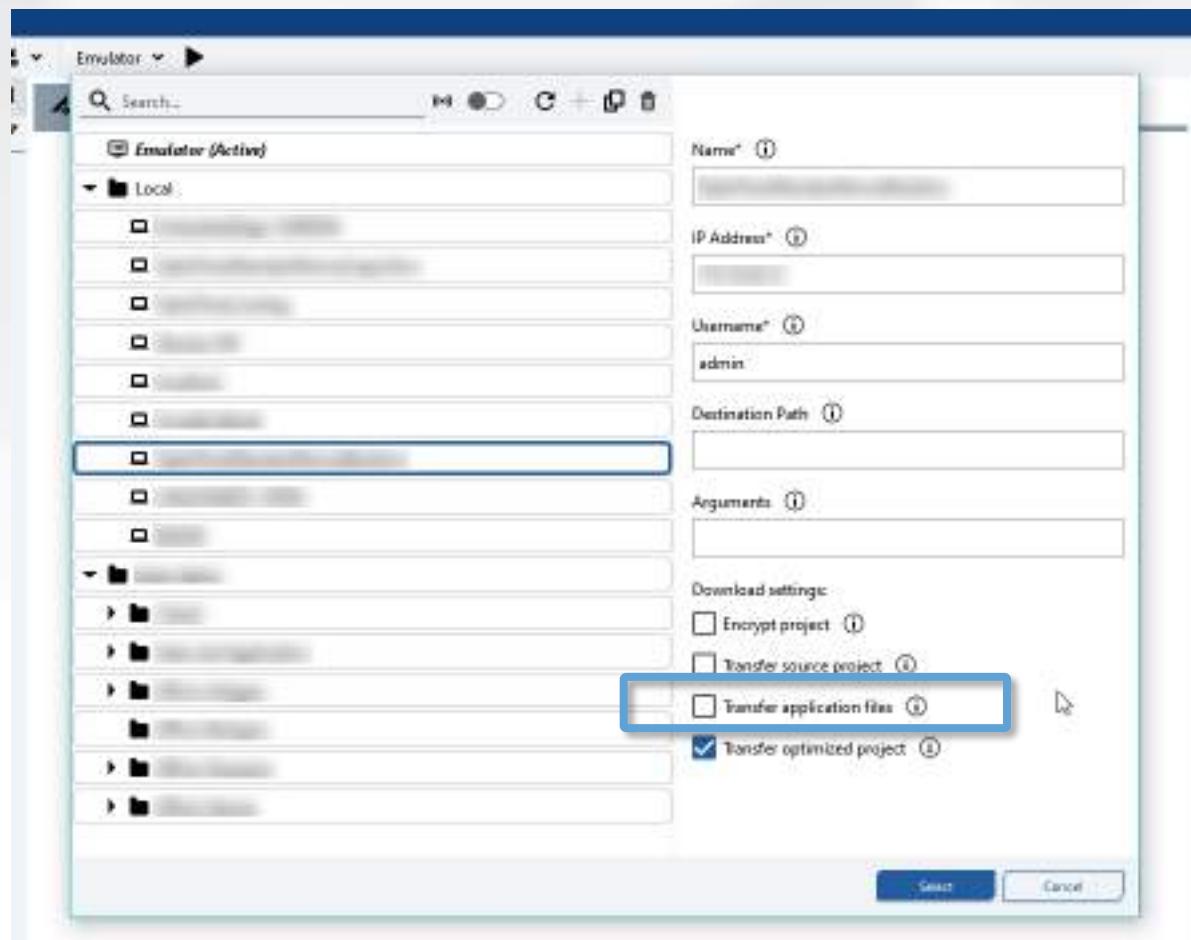
# Deployment options

- Source code of the project can be transferred to the target device
  - All YAML files, resources and NetSolution will be transferred to the target device
  - If the project sources were transferred to the device, the project can be uploaded later from the device using the dedicated menu in the welcome screen of FactoryTalk Optix IDE



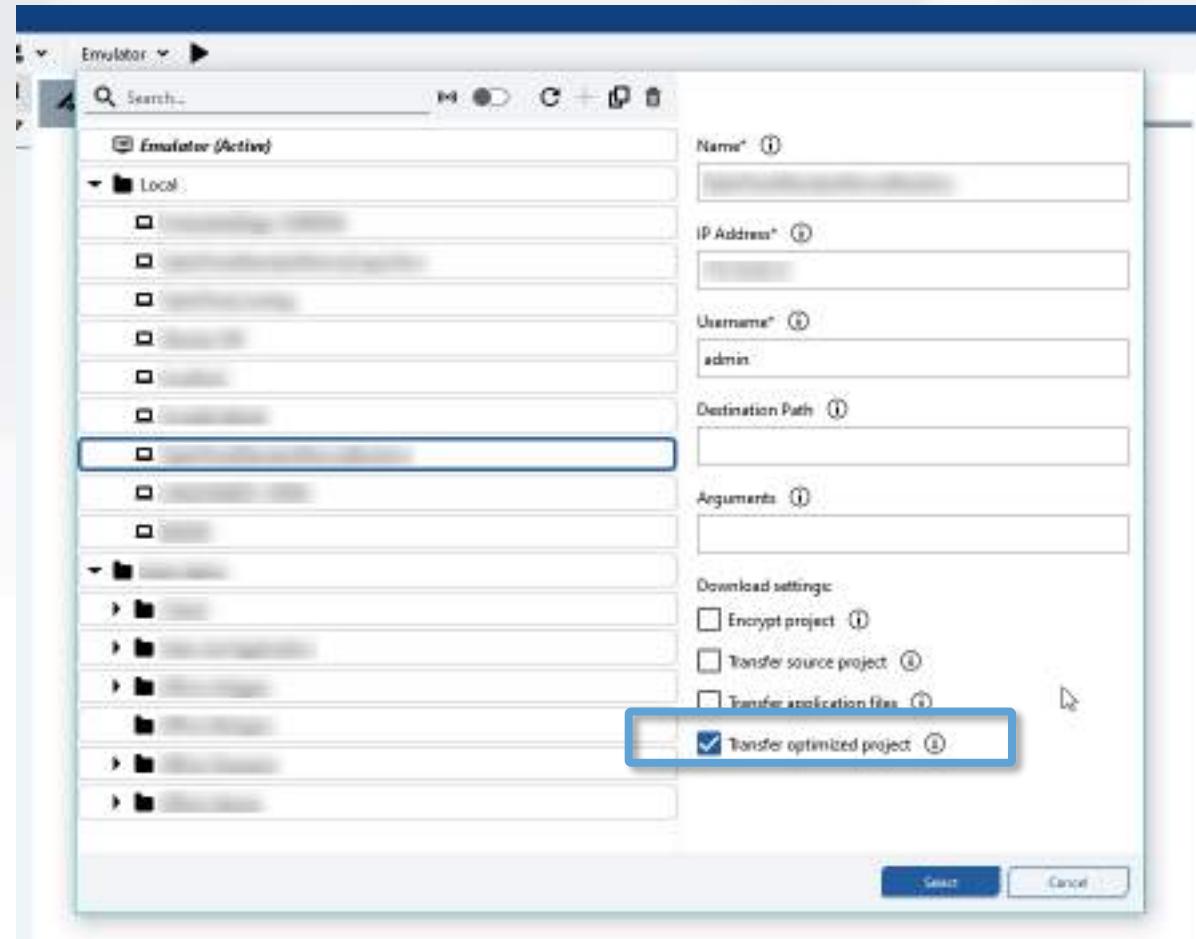
# Deployment options

- Application files on the target device can be overwritten
  - When a project is uploaded from a device, the files in the ApplicationDir can be uploaded too, this includes: retentivity, users, database, etc.
  - Once the files were uploaded, they can be transferred to the same or a different device



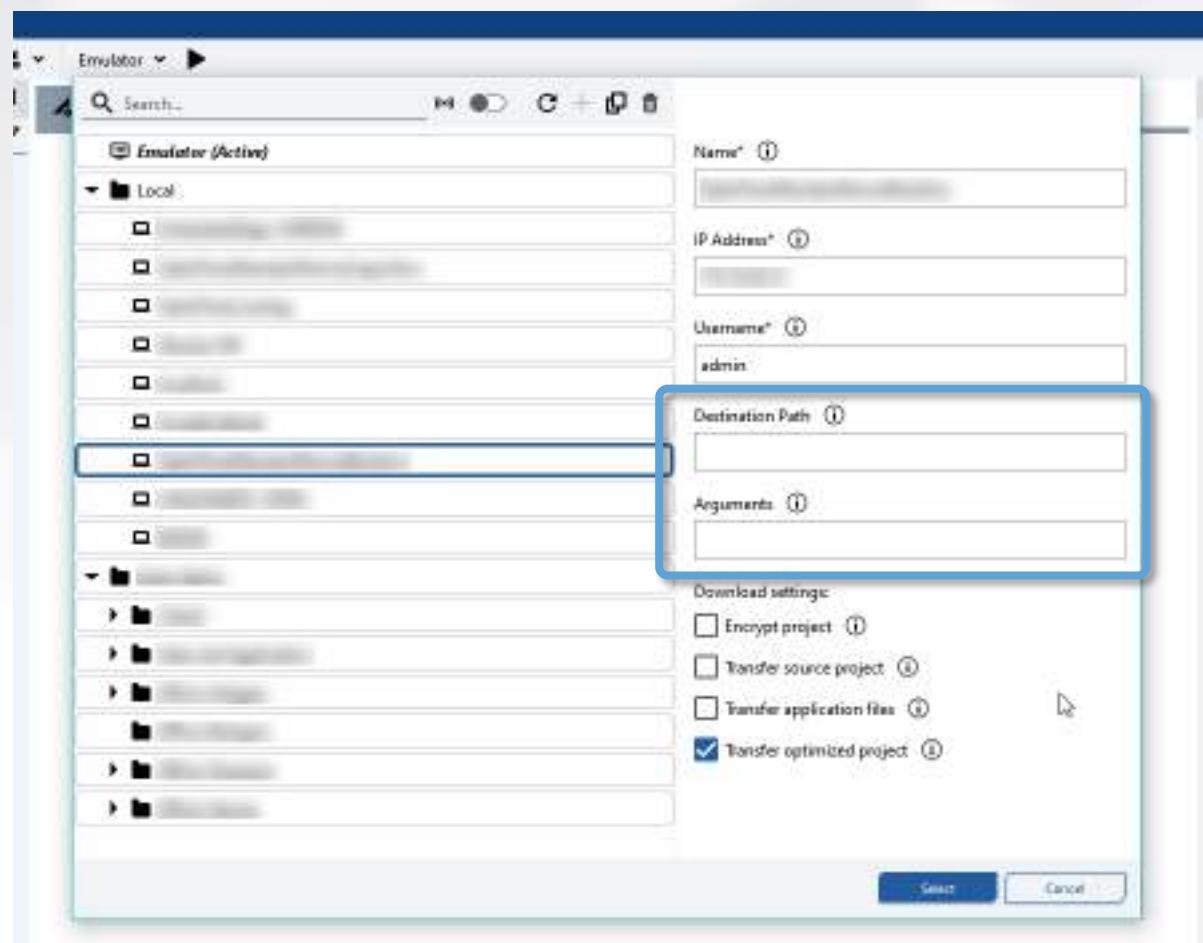
# Deployment options

- Optimize the project before deployment
  - In some cases, a lot of tags were imported in the application even if not used in the application
  - The optimization process will move all imported tags from the YAML file to a dedicated binary repository where tags are retrieved only if needed.
  - This can significantly improve startup time and memory usage
  - Cannot be used if the OPC/UA server is exposing the PLC tags



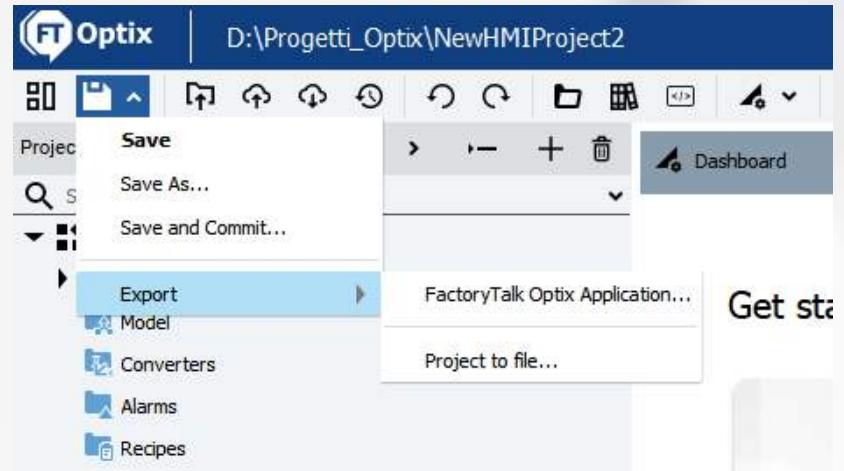
# Additional parameters

- A specific path on the remote device can be specified
  - Only for Windows-based devices
  - The user being selected for deployment must have write access to such path
- Additional startup arguments can be added
  - Only used for debugging if requested by SW support or R&D team, should be always blank

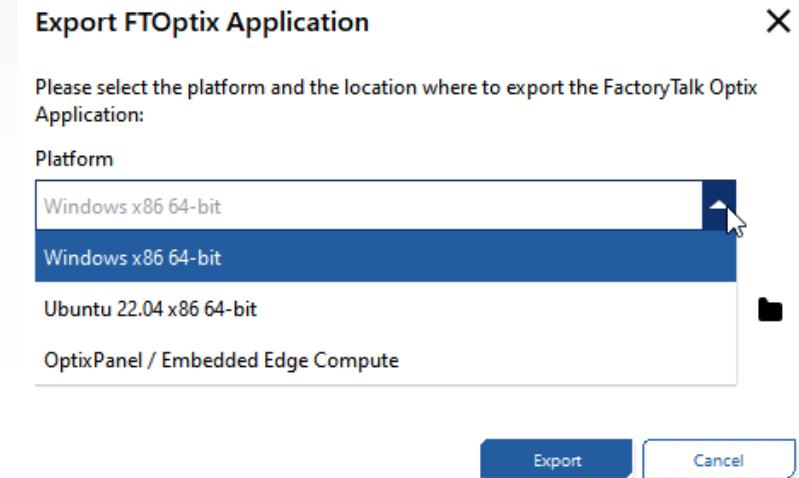


# Transfer project to target manually

- Application can be transferred manually
  - "**Export FactoryTalk Optix Application**" : exports the (semi) compiled Application into a folder that needs to be manually transferred and executed on Target
  - No encryption can be enabled if exporting the application from this menu



Platform	HMI Model
Optix Panel	Optix Panel Standard or Compact
Windows (on x64 CPU)	iPC Windows
Ubuntu (on x64 CPU)	iPC Linux with Ubuntu 22



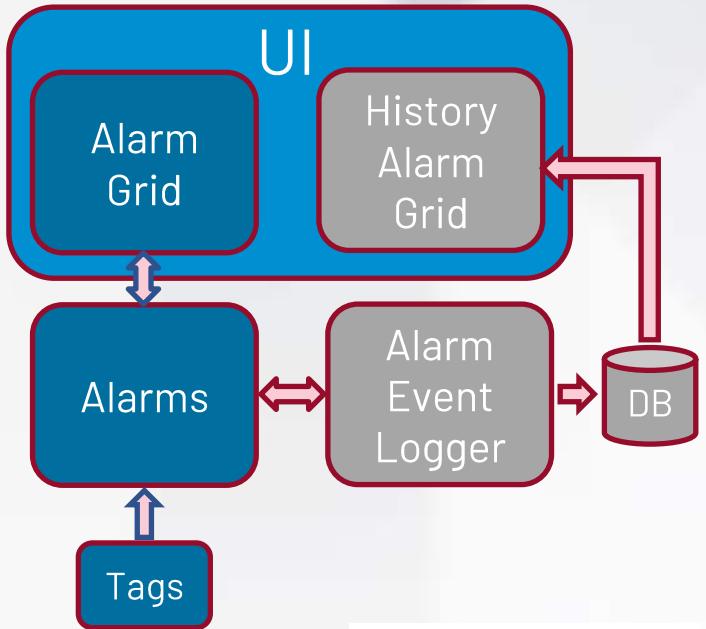


# Manage alarms



# Alarms

- Alarm functional module is used to define alarm events connected to Tags/Variables and manage them at runtime
- Active Alarms
  - just define alarms
  - drag&drop Alarm Grid widget from Template Library to UI
- History of Alarms
  - add a new datastore (Embedded database or ODBC)
  - drag&drop Alarm Event Logger object from Template Library, into Loggers node
  - drag&drop History Alarm Grid widget from Template Library to UI



OPC/UA Alarms  
specifications

# Alarm types

- Types of alarms:
  - *Digital*: "two-state" alarm, typically used to monitor Boolean tags
  - *Level*: an alarm that activates when the monitored variable exceed the range "Low – High"
  - *Deviation*: alarms that activate when the monitored variable exceed the range defined by a Setpoint and Low/High deviation limit
  - *Rate of Change*: like Level Alarm but evaluated with specific polling time (i.e. to filter spurious spikes)
- Exclusive/Non-Exclusive (when all four limits are used)
  - Exclusive: exceeding High and High-high limits, only High-high alarms will be shown
  - Non-Exclusive: exceeding High and High-high limits, both alarms will be shown

Properties	
↑ I/O	Name DigitalAlarm1
	Type Digital alarm
Input variable	0
Normal state value	0
Enabled	True
Auto acknowledge	False
Auto confirm	False
Message	
Severity	1

# Alarm types

- Types of alarms:
  - *Digital*: "two-state" alarm, typically used to monitor Boolean tags
  - *Level*: an alarm that activates when the monitored variable exceed the range "Low – High"
  - *Deviation*: alarms that activate when the monitored variable exceed the range defined by a Setpoint and Low/High deviation limit
  - *Rate of Change*: like Level Alarm but evaluated with specific polling time (i.e. to filter spurious spikes)
- Exclusive/Non-Exclusive (when all four limits are used)
  - Exclusive: exceeding High and High-high limits, only High-high level will be shown
  - Non-Exclusive: exceeding High and High-high limits, both levels will be shown

Properties	
Name	ExclusiveLevelAlarm1
Type	Exclusive level alarm
Input variable	0
High-high limit	0
High limit	0
Low limit	0
Low-low limit	0
Enabled	True
Auto acknowledge	False
Auto confirm	False
Message	
Severity	1

# Alarm types

- Types of alarms:
  - *Digital*: "two-state" alarm, typically used to monitor Boolean tags
  - *Level*: an alarm that activates when the monitored variable exceed the range "Low – High"
  - *Deviation*: alarms that activate when the monitored variable exceed the range defined by a Setpoint and Low/High deviation limit
  - *Rate of Change*: like Level Alarm but evaluated with specific polling time (i.e. to filter spurious spikes)
- Exclusive/Non-Exclusive (when all four limits are used)
  - Exclusive: exceeding High and High-high limits, only High-high level will be shown
  - Non-Exclusive: exceeding High and High-high limits, both levels will be shown

Properties	
Name	ExclusiveDeviationAlarm1
Type	Exclusive deviation alarm
Input variable	0
Set point value	0
High-high limit	0
High limit	0
Low limit	0
Low-low limit	0
Enabled	True
Auto acknowledge	False
Auto confirm	False
Message	
Severity	1

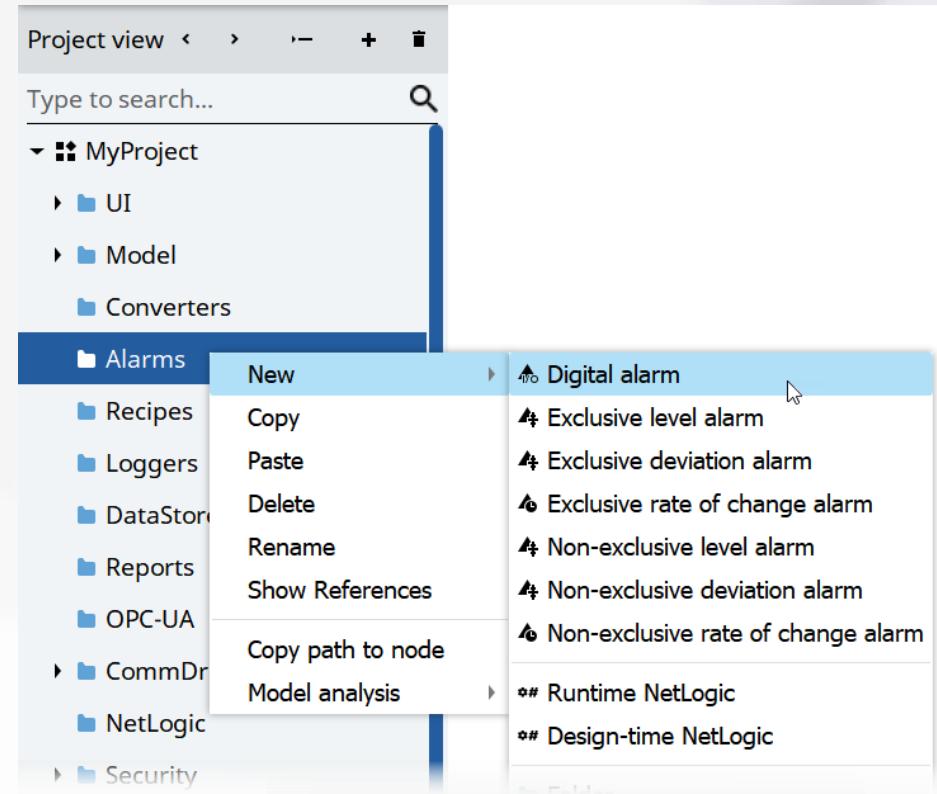
# Alarm types

- Types of alarms:
  - *Digital*: "two-state" alarm, typically used to monitor Boolean tags
  - *Level*: an alarm that activates when the monitored variable exceed the range "Low – High"
  - *Deviation*: alarms that activate when the monitored variable exceed the range defined by a Setpoint and Low/High deviation limit
  - *Rate of Change*: like Level Alarm but evaluated with specific polling time (i.e. to filter spurious spikes)
- Exclusive/Non-Exclusive (when all four limits are used)
  - Exclusive: exceeding High and High-high limits, only High-high level will be shown
  - Non-Exclusive: exceeding High and High-high limits, both levels will be shown

Properties	
Name	ExclusiveRateOfChangeAlarm1
Type	Exclusive rate of change alarm
Input variable	0
Polling time	0000:00:00.000
High-high limit	0
High limit	0
Low limit	0
Low-low limit	0
Enabled	True
Auto acknowledge	False
Auto confirm	False
Message	
Severity	1

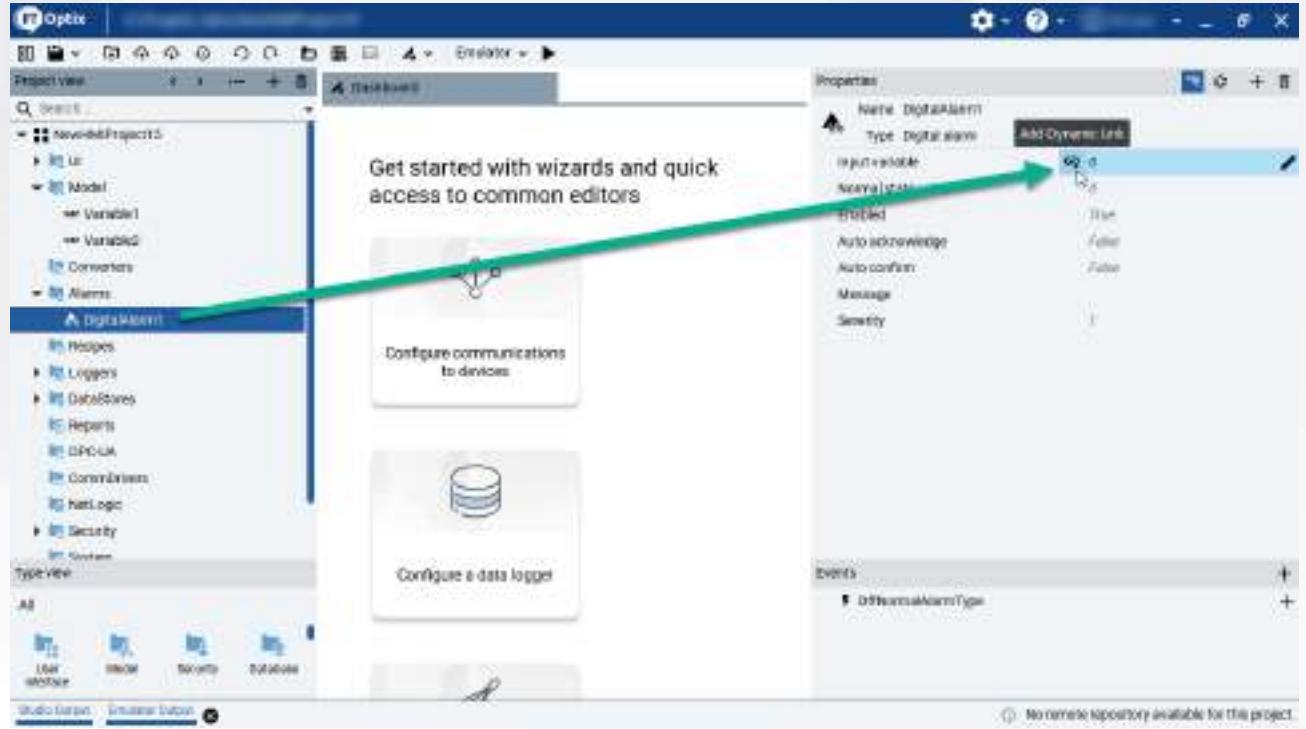
# Alarm definition

- How to define alarms
  1. Choose the alarm type
  2. Configure Input Variable and properties
- For Tags where every bit bring alarm information
  1. Select the single tag
  2. Select the bit index
- Import/Export
  - Alarms Definition can be imported/exported to CSV file using a Template Library script (customizable as needed)
  - The path of the CSV file can be set to the current project folder using the %PROJECTDIR% placeholder



# Alarm definition

- How to define alarms
  1. Choose the alarm type
  2. Configure Input Variable and properties
- For Tags where every bit bring alarm information
  1. Select the single tag
  2. Select the bit index
- Import/Export
  - Alarms Definition can be imported/exported to CSV file using a Template Library script (customizable as needed)
  - The path of the CSV file can be set to the current project folder using the %PROJECTDIR% placeholder

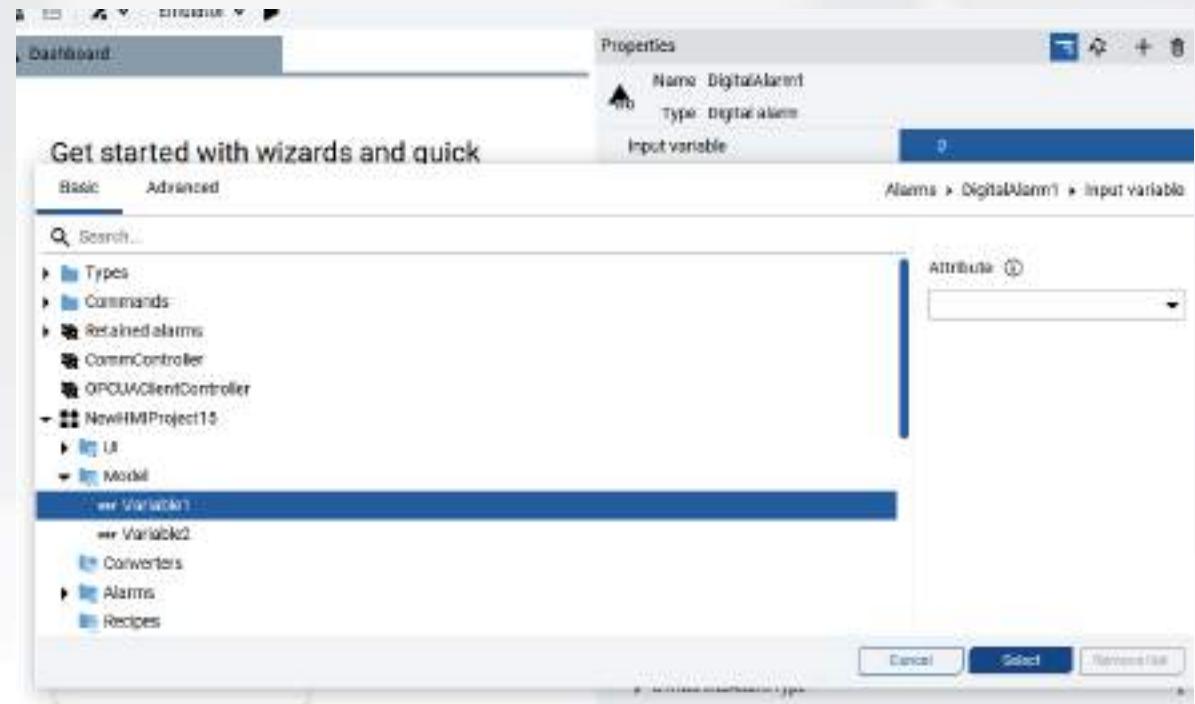


# Alarm definition

- How to define alarms
  1. Choose the alarm type
  2. Configure Input Variable and properties

- For Tags where every bit bring alarm information

1. Select the single tag
2. Select the bit index

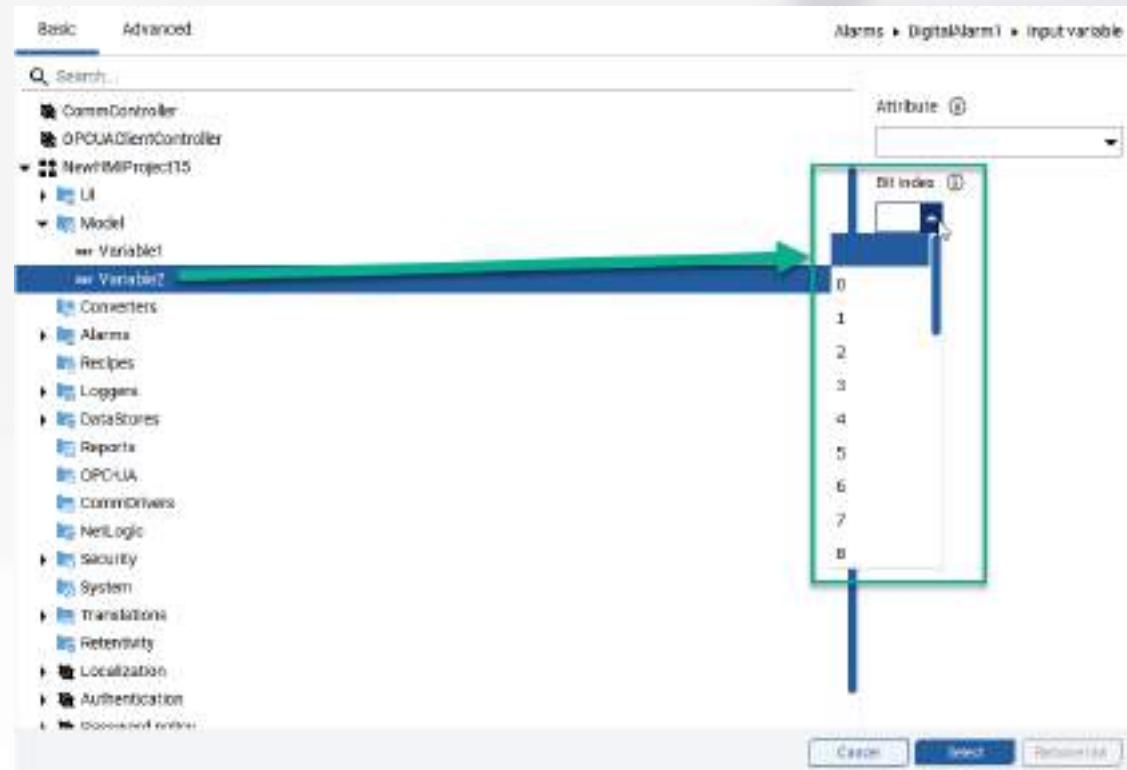


- Import/Export

- Alarms Definition can be imported/exported to CSV file using a Template Library script (customizable as needed)
- The path of the CSV file can be set to the current project folder using the %PROJECTDIR% placeholder

# Alarm definition

- How to define alarms
  1. Choose the alarm type
  2. Configure Input Variable and properties
- For Tags where every bit bring alarm information
  1. Select the single tag
  2. Select the bit index
- Import/Export
  - Alarms Definition can be imported/exported to CSV file using a Template Library script (customizable as needed)
  - The path of the CSV file can be set to the current project folder using the %PROJECTDIR% placeholder



# Show alarms

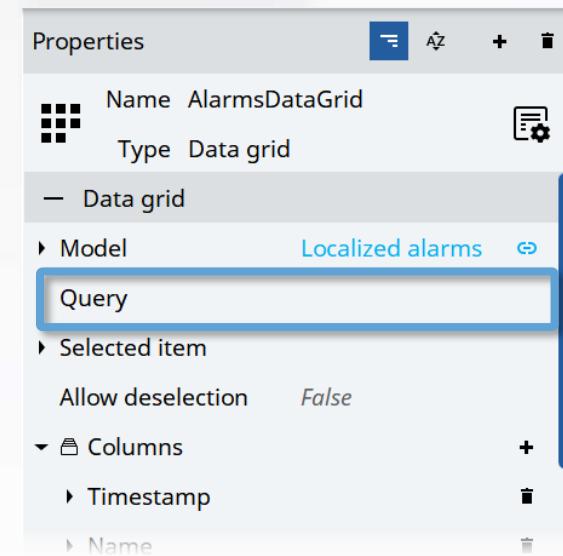
- Available Widgets

- into Template Library
  - Alarm Grid,
  - Alarm Banner,
  - Alarm Icon

Timestamp	Name	Source Variable	Message	Active	Acked	Confirmed	Severity
Feb 10, 2022, 3:44:05 PM	DigitalAlarm1.1	ModbusTag1	Alarm on Tag1 bit 1	●	✗	✓	1
Feb 10, 2022, 3:44:04 PM	DigitalAlarm1.0	ModbusTag1	Alarm on Tag1 bit 0	●	✗	✓	1

**Acknowledge** **Acknowledge All** **Confirm** **Confirm All**

- customizable
  - columns,
  - buttons,
  - the "Query" parameter allow to filter alarms of Alarm Grid
- Support Messages in different languages
- Supports different messages per each threshold of analog alarms



# Show alarms history

- To record alarms history some additional resources are needed

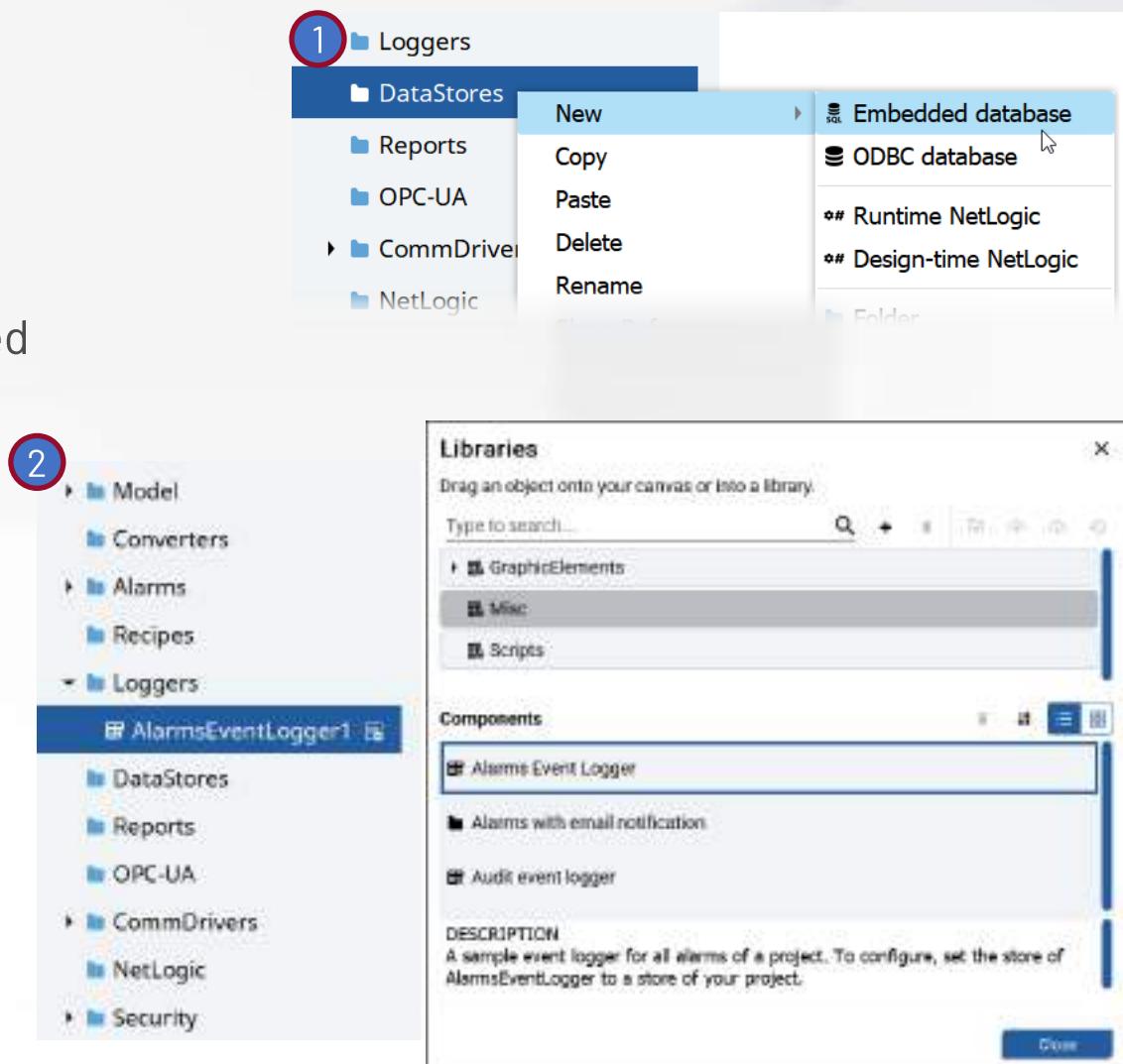
1. Datastore resource (if not already present)
2. Alarm event logger (from Template Library) just drag&drop and configure datastore to be used

- Available Widgets

- Alarm History grid
- Alarm History grid with filter

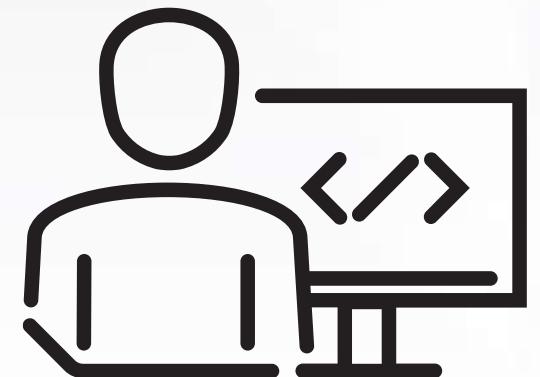
- Export

- Alarm History can be exported with script "Alarms History Exporter" available in Template Library



# Hands-on session

- Define some digital alarms configuring the input variables
- Add the Alarm Grid widget to see alarms at runtime with the emulator
- Add the Alarm Event Logger to log historical alarm events
- Add the Alarm History Grid widget to see historical

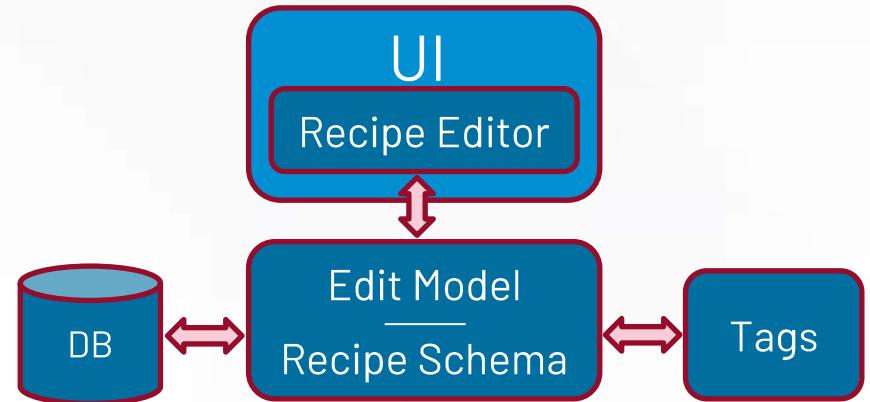
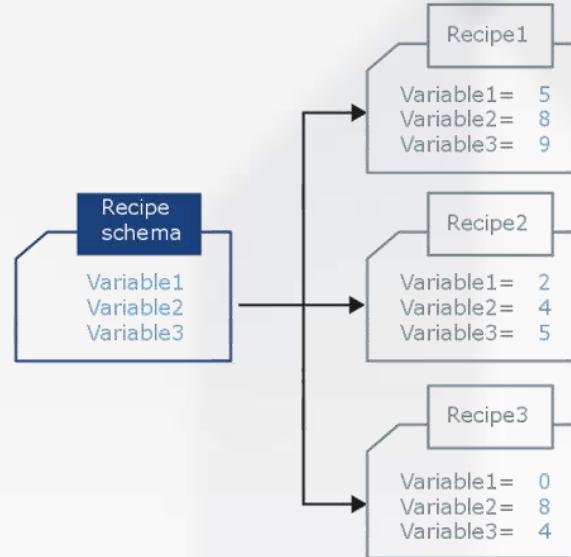


# Recipes



# Recipes

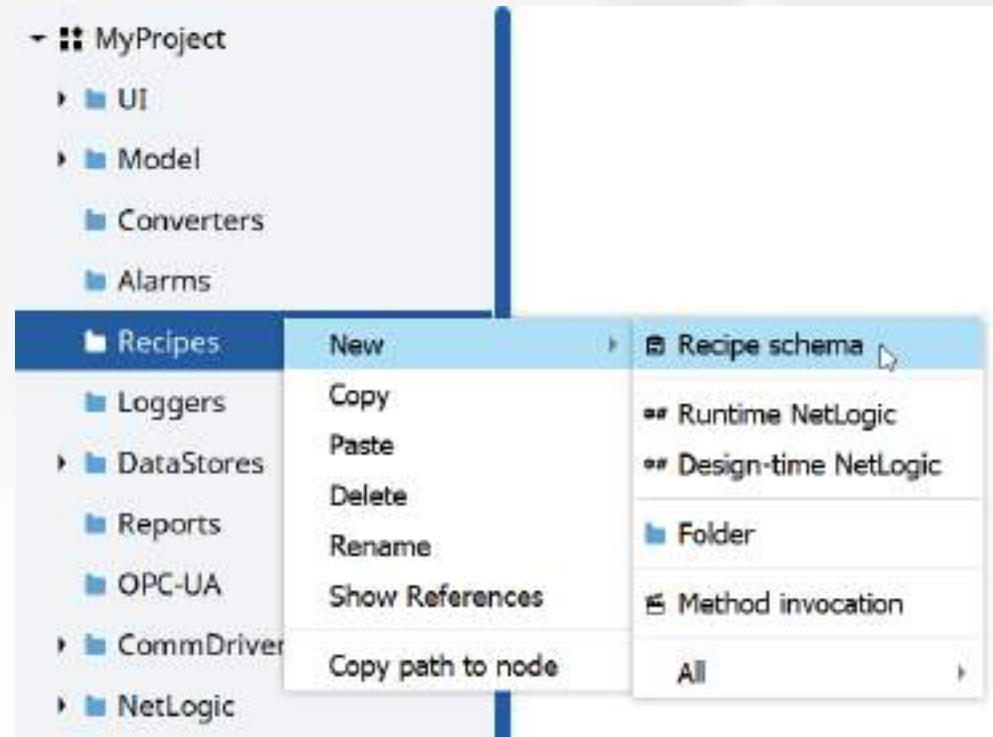
- Allows defining a group of values/set points stored in database archives (embedded or ODBC)
- in Studio
  - configure the Recipe Schema by defining datastore and selecting Tags
- in Runtime
  - User can create several recipes, by defining values manually or by reading values directly from PLC
  - User can store the recipe in the database for future use
  - User can retrieve the recipe from the database and transfer to the PLC



# Configure a recipe schema

- Steps to configure Recipes

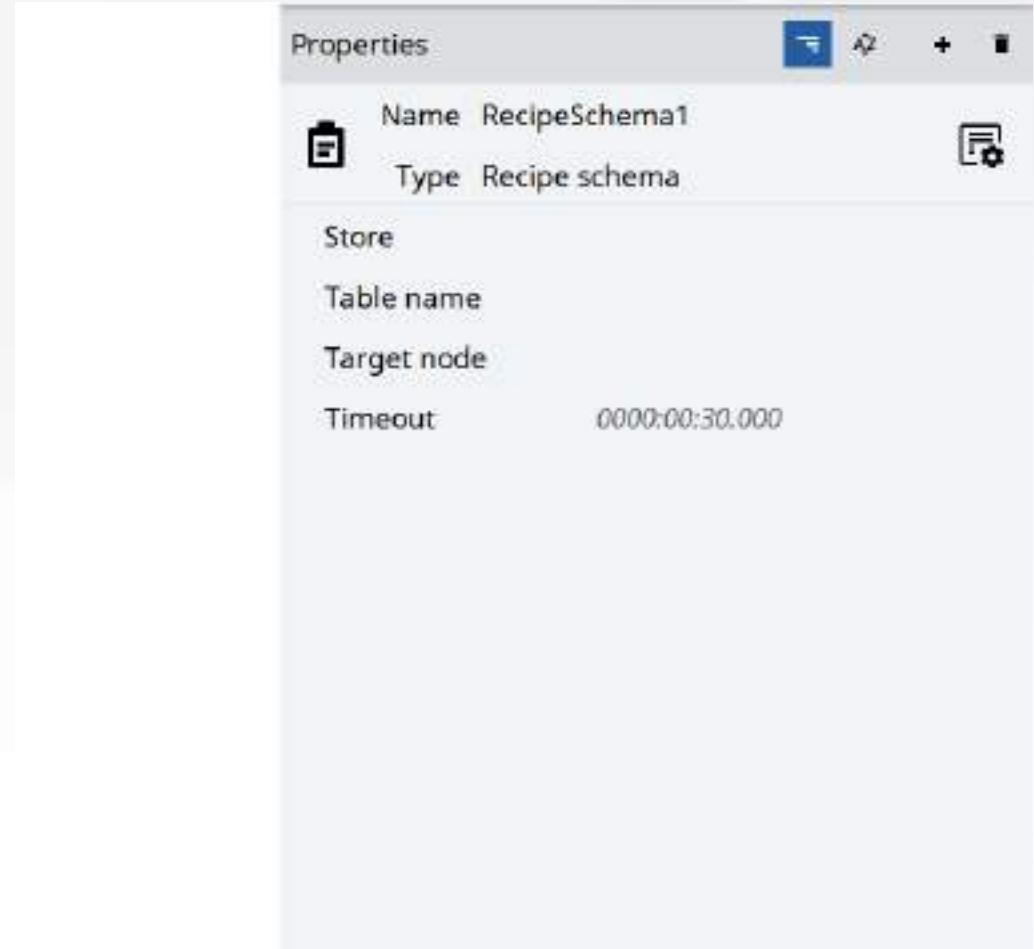
1. Add a Recipe Schema by  
Recipes > New > Recipe Schema
2. Select the Datastore  
(if not available add a new one)
3. Configure the Target Node  
This is the project folder containing Variables  
that will be used in the Recipe Schema  
Example: Tags, Models...
4. Select which Variables add to the Schema  
then confirm with "Apply"



# Configure a recipe schema

- Steps to configure Recipes

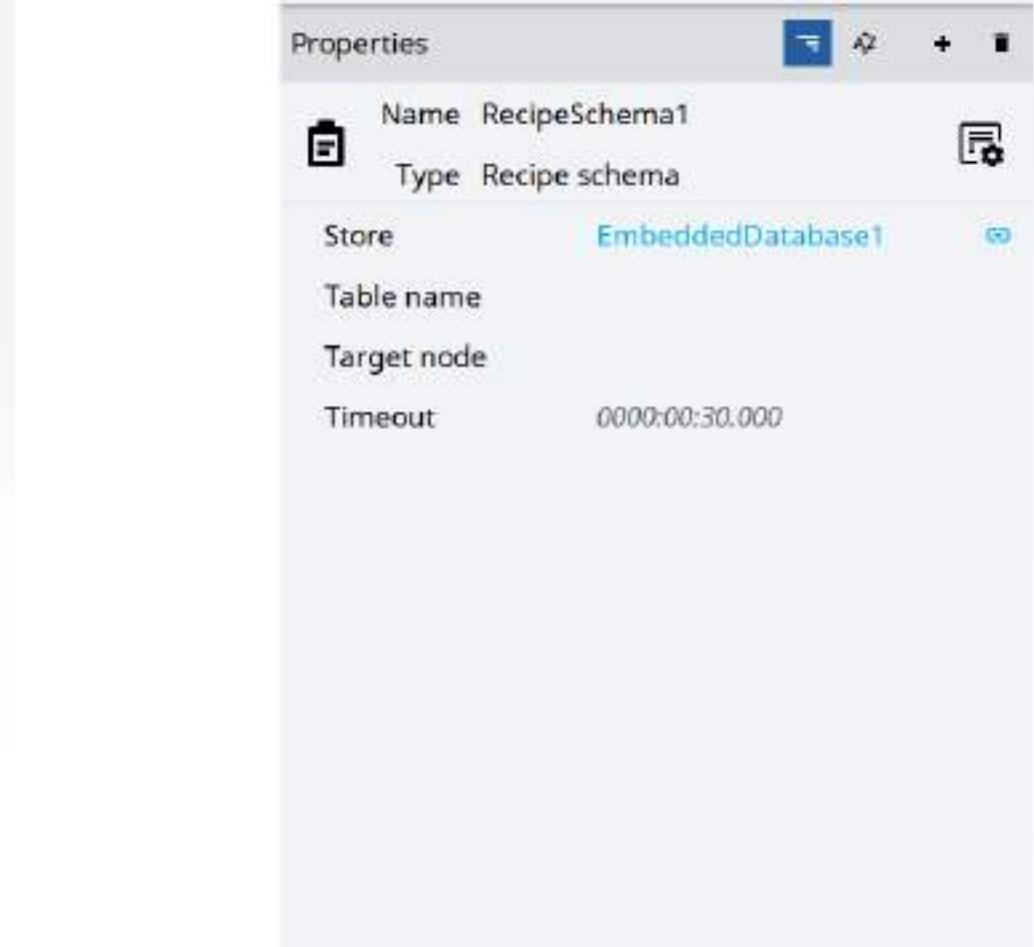
1. Add a Recipe Schema by  
Recipes > New > Recipe Schema
2. Select the Datastore  
(if not available add a new one)
3. Configure the Target Node  
This is the project folder containing Variables  
that will be used in the Recipe Schema  
Example: Tags, Models...
4. Select which Variables add to the Schema  
then confirm with "Apply"



# Configure a recipe schema

- Steps to configure Recipes

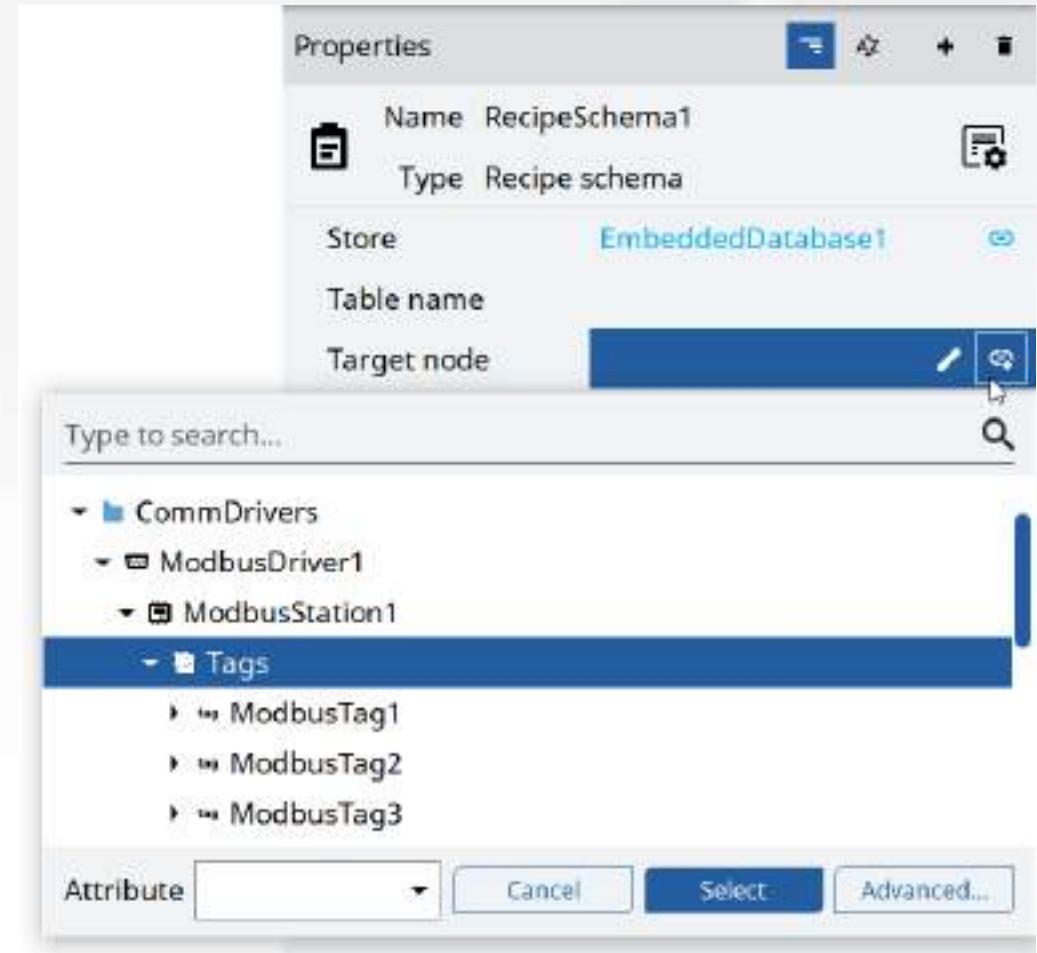
1. Add a Recipe Schema by  
Recipes > New > Recipe Schema
2. Select the Datastore  
(if not available add a new one)
3. Configure the Target Node  
This is the project folder containing Variables  
that will be used in the Recipe Schema  
Example: Tags, Models...
4. Select which Variables add to the Schema  
then confirm with "Apply"



# Configure a recipe schema

- Steps to configure Recipes

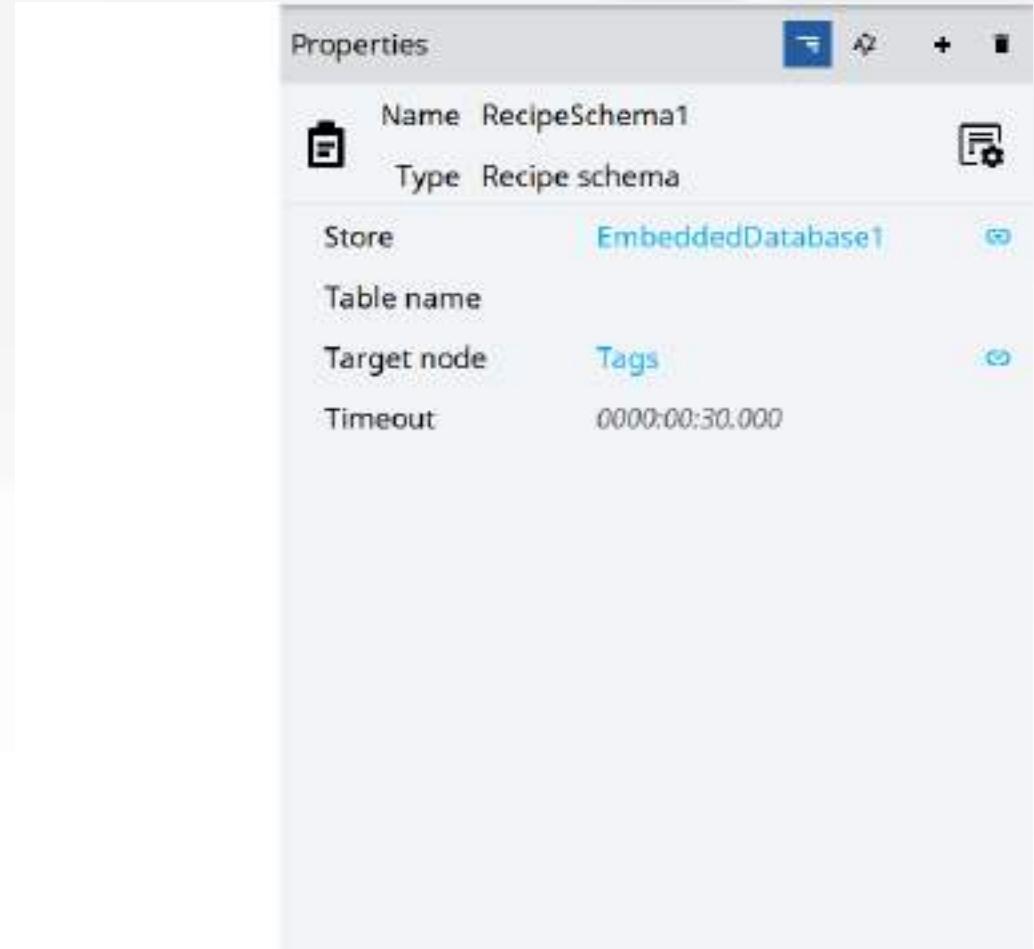
1. Add a Recipe Schema by  
Recipes > New > Recipe Schema
2. Select the Datastore  
(if not available add a new one)
3. Configure the Target Node  
This is the project folder containing Variables  
that will be used in the Recipe Schema  
Example: Tags, Models...
4. Select which Variables add to the Schema  
then confirm with "Apply"



# Configure a recipe schema

- Steps to configure Recipes

1. Add a Recipe Schema by  
Recipes > New > Recipe Schema
2. Select the Datastore  
(if not available add a new one)
3. Configure the Target Node  
This is the project folder containing Variables  
that will be used in the Recipe Schema  
Example: Tags, Models...
4. Select which Variables add to the Schema  
then confirm with "Apply"



# Configure a recipe schema

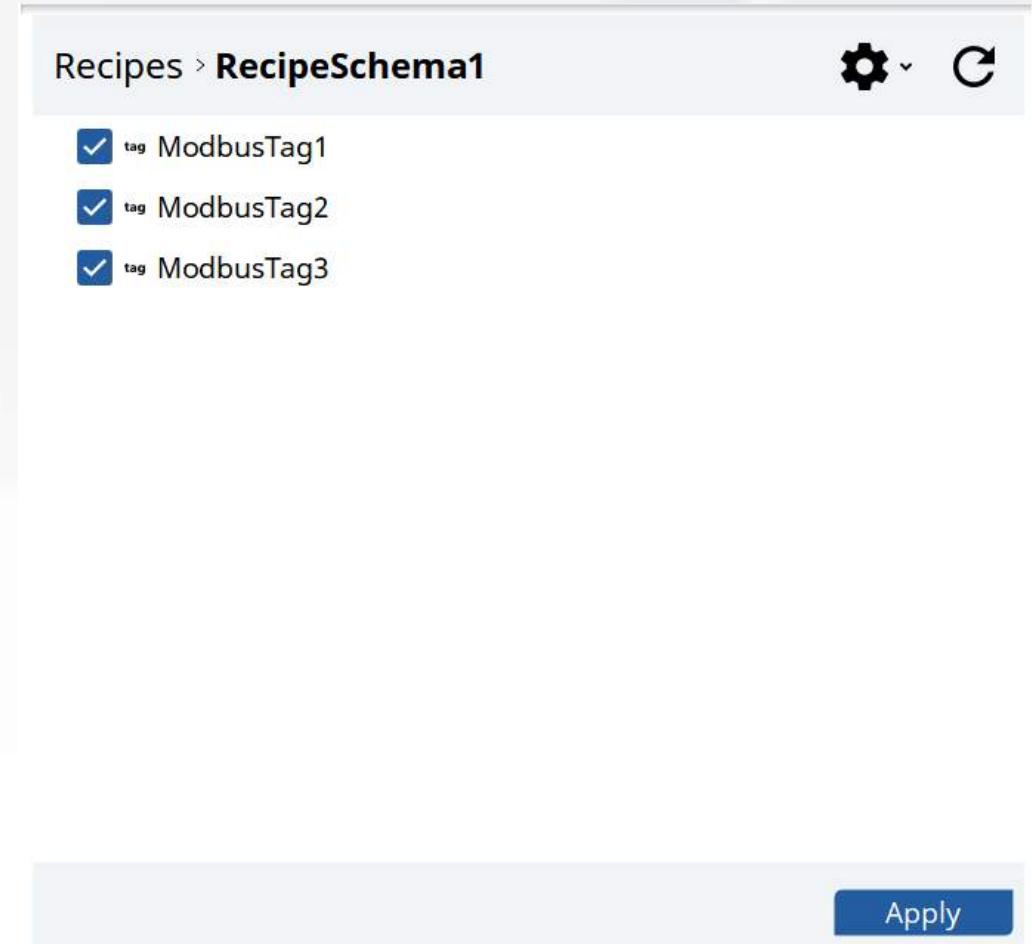
- Steps to configure Recipes

1. Add a Recipe Schema by  
Recipes > New > Recipe Schema

2. Select the Datastore  
(if not available add a new one)

3. Configure the Target Node  
This is the project folder containing Variables  
that will be used in the Recipe Schema  
Example: Tags, Models...

4. Select which Variables add to the Schema  
then confirm with "Apply"



# Recipe editor widget

- Steps to configure the Recipe Editor widget

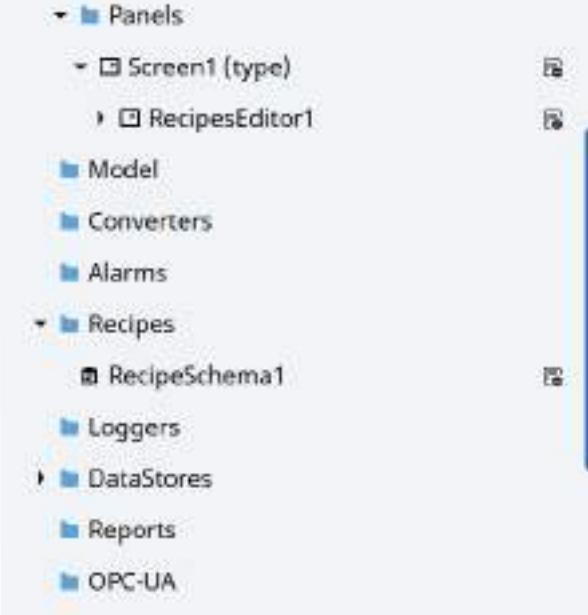
1. Drag&drop the widget from Template Library
2. Into Property pane,  
select the recipe schema
3. Right-click on the "RecipeEditorUISetup"  
and select "Execute Setup"

- Widget:

- Save: save/create the recipe into the database
- Delete: delete the recipe from the database
- Load: read Tag values and load them into recipe values
- Apply: write recipe values into Tags

- Import/Export

- Recipes can be exported/imported  
through script available into Template Library



# Recipe editor widget

- Steps to configure the Recipe Editor widget

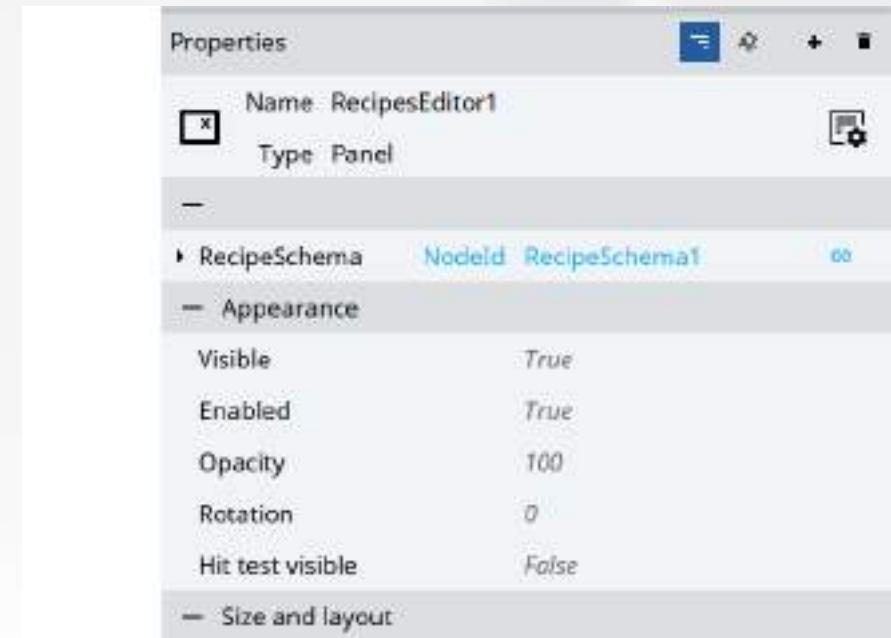
1. Drag&drop the widget from Template Library
2. Into Property pane,  
select the recipe schema
3. Right-click on the "RecipeEditorUISetup"  
and select "Execute Setup"

- Widget:

- Save: save/create the recipe into the database
- Delete: delete the recipe from the database
- Load: read Tag values and load them into recipe values
- Apply: write recipe values into Tags

- Import/Export

- Recipes can be exported/imported  
through script available into Template Library



# Recipe editor widget

- Steps to configure the Recipe Editor widget

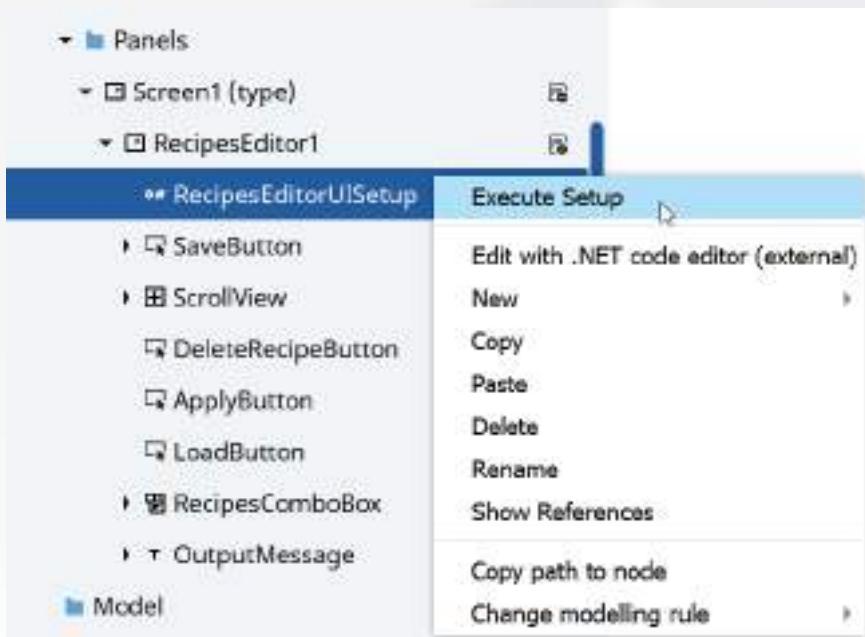
1. Drag&drop the widget from Template Library
2. Into Property pane,  
select the recipe schema
3. Right-click on the "RecipeEditorUISetup"  
and select "Execute Setup"

- Widget:

- Save: save/create the recipe into the database
- Delete: delete the recipe from the database
- Load: read Tag values and load them into recipe values
- Apply: write recipe values into Tags

- Import/Export

- Recipes can be exported/imported  
through script available into Template Library



# Recipe editor widget

- Steps to configure the Recipe Editor widget

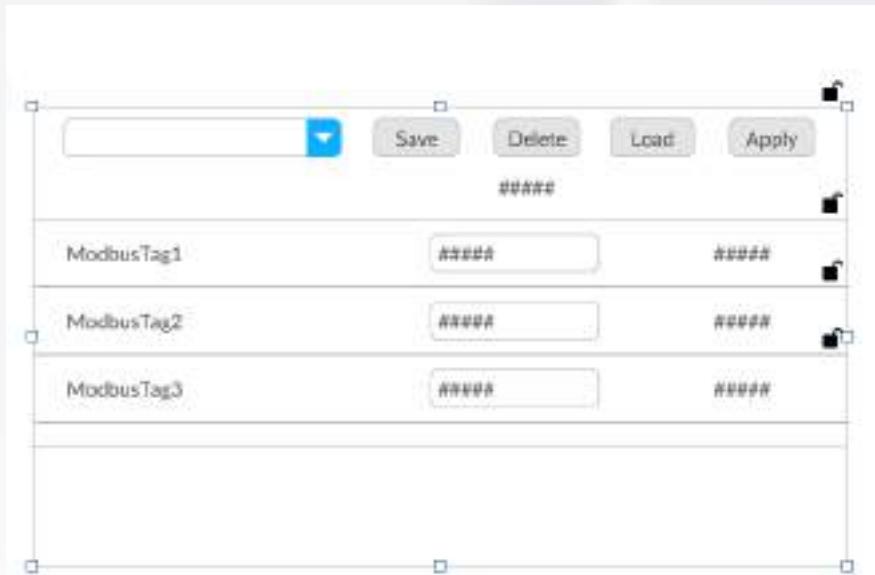
1. Drag&drop the widget from Template Library
2. Into Property pane,  
select the recipe schema
3. Right-click on the "RecipeEditorUISetup"  
and select "Execute Setup"

- Widget:

- Save: save/create the recipe into the database
- Delete: delete the recipe from the database
- Load: read Tag values and load them into recipe values
- Apply: write recipe values into Tags

- Import/Export

- Recipes can be exported/imported  
through script available into Template Library



# Recipe editor widget

- Steps to configure the Recipe Editor widget

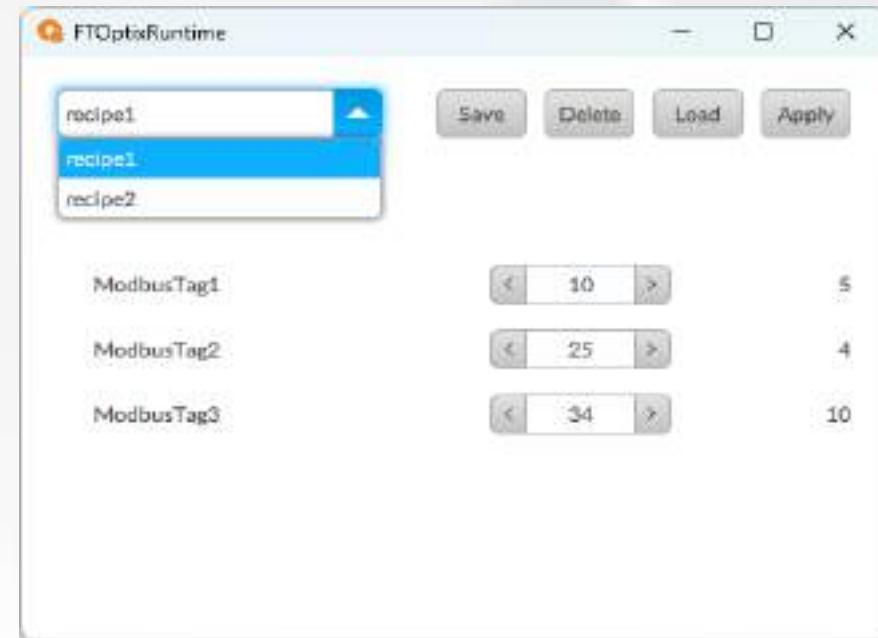
1. Drag&drop the widget from Template Library
2. Into Property pane,  
select the recipe schema
3. Right-click on the "RecipeEditorUISetup"  
and select "Execute Setup"

- Widget:

- Save: save/create the recipe into the database
- Delete: delete the recipe from the database
- Load: read Tag values and load them into recipe values
- Apply: write recipe values into Tags

- Import/Export

- Recipes can be exported/imported  
through script available into Template Library



# Recipe editor widget

- Steps to configure the Recipe Editor widget

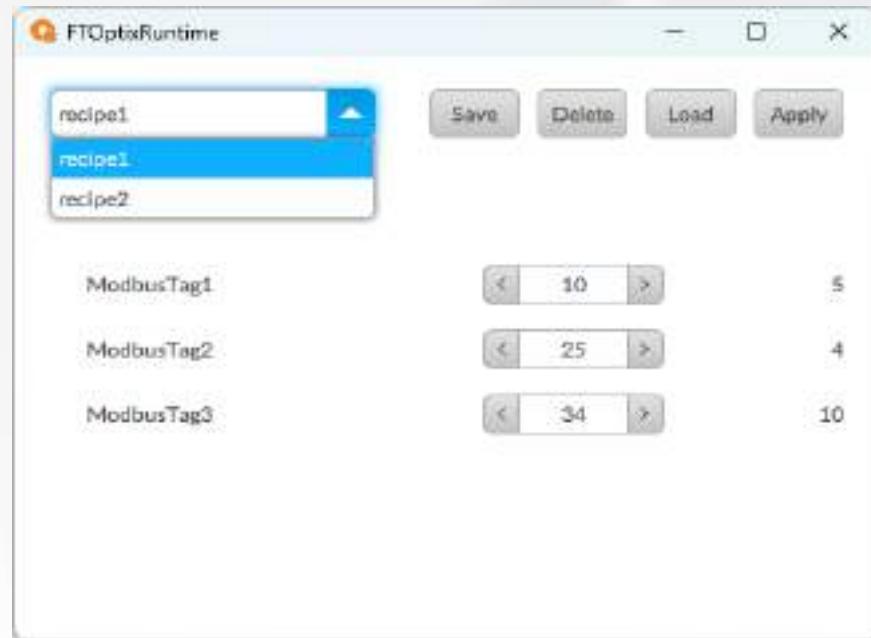
1. Drag&drop the widget from Template Library
2. Into Property pane,  
select the recipe schema
3. Right-click on the "RecipeEditorUISetup"  
and select "Execute Setup"

- Widget:

- Save: save/create the recipe into the database
- Delete: delete the recipe from the database
- Load: read Tag values and load them into recipe values
- Apply: write recipe values into Tags

- Import/Export

- Recipes can be exported/imported  
through script available into Template Library



# Hands-on session

- Create a recipe schema using tags coming from the communication driver
- Add the recipe widget to the project
- Execute the Recipe widget "setup"
- Play with the widget at runtime with Emulator

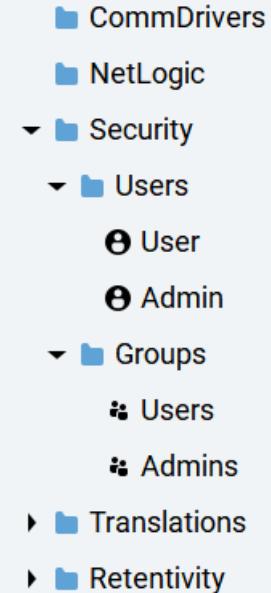


# Configure users and groups



# Users and groups

- Allow defining a level of Security for commands/UI objects
- Project Properties
  - Authentication Mode: Model = Project's users, Local = iPC Local users, Domain = Active Directory users
  - Default User Folder
  - Password Policy
- Presentation Engine properties
  - Login Window: on Web Presentation Engine only, allow to specify a Window that should contain a Login Form to be prompted at startup.
  - Starting user: "Anonymous" it's the user logged in at the startup (it's not visible in the Users folder)



# Users and groups

- Allow defining a level of Security for commands/UI objects

- Project Properties

- Authentication Mode: Model = Project's users,  
Local = iPC Local users,  
Domain = Active Directory users
- Default User Folder
- Password Policy

▼ Authentication	
Authentication mode	<i>Model only</i>
Default user folder	<a href="#">Browse</a>
Default domain name	<a href="#">Browse</a>
Domain server address	
CA certificate file	
▼ Password policy	
Maximum password age	0
Enforce password history	1
Minimum password age	0
Minimum password length	8

- Presentation Engine properties

- Login Window: on Web Presentation Engine only, allow to specify a Window that should contain a Login Form to be prompted at startup.
- Starting user: "Anonymous" it's the user logged in at the startup (it's not visible in the Users folder)

# Users and groups

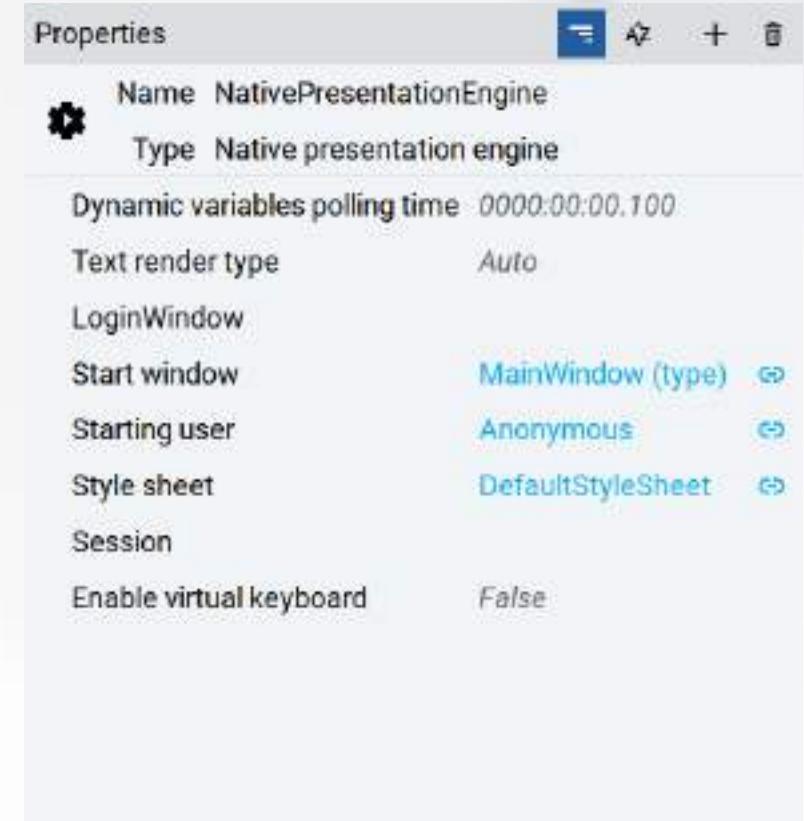
- Allow defining a level of Security for commands/UI objects

- Project Properties

- Authentication Mode:
  - Model = Project's users,
  - Local = iPC Local users,
  - Domain = Active Directory users
- Default User Folder
- Password Policy

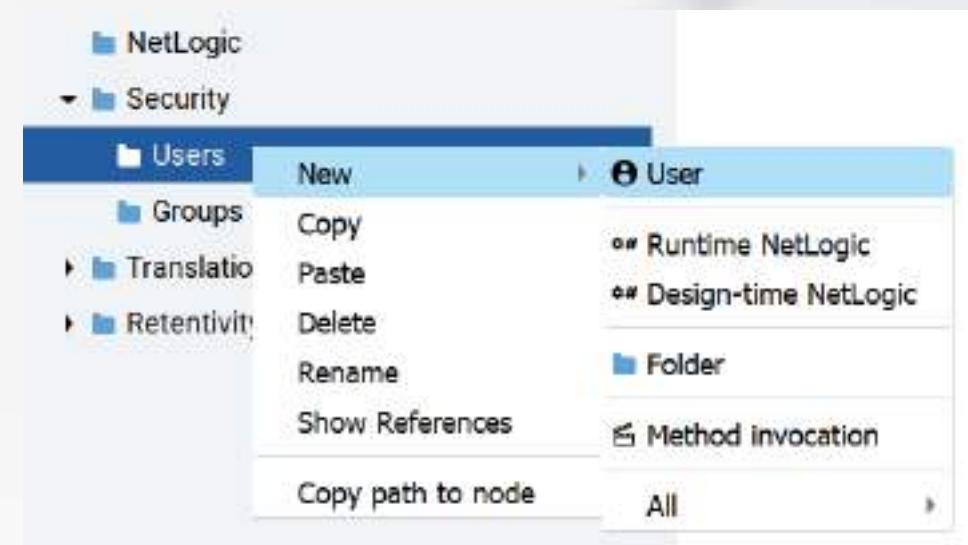
- Presentation Engine properties

- Login Window: on Web Presentation Engine only, allow to specify a Window that should contain a Login Form to be prompted at startup.
- Starting user: "Anonymous" it's the user logged in at the startup (it's not visible in the Users folder)



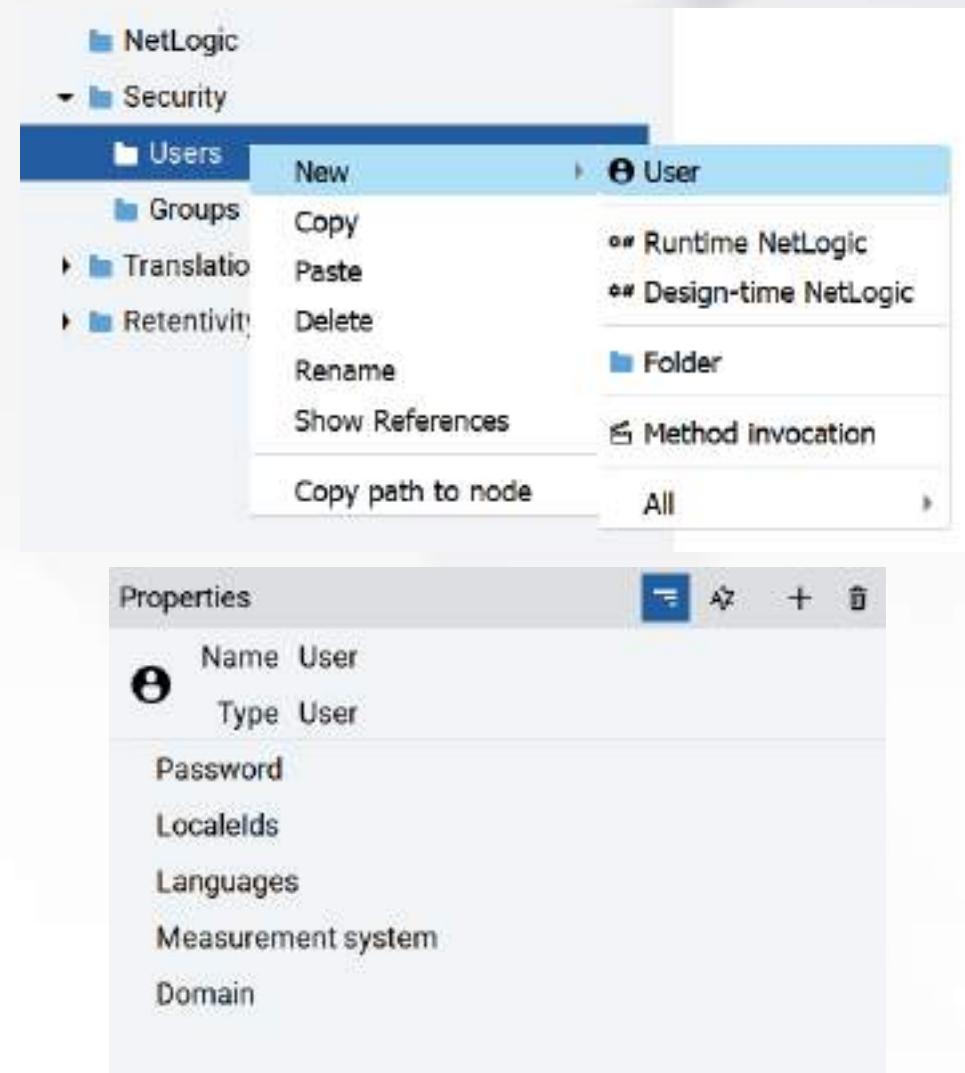
# Create users and groups

- User/Group Creation:
  - Users and Groups can be created manually, by the right click on Users or Groups folders (except for Local/Domain users)
- User properties:
  - Password
  - LocaleId: a combination of Regional Settings, Language and Measurement System
  - Languages: Translation table column to be used (overrides the default language of the LocaleId)
  - Measurement system: International System, British imperial or US Customary (overrides the default Measurement system of the LocaleId)
  - Domain: Domain name used in case of authentication mode: Domain only, Domain and Local, or Any (it is automatically populated after a successful login) – blank by default



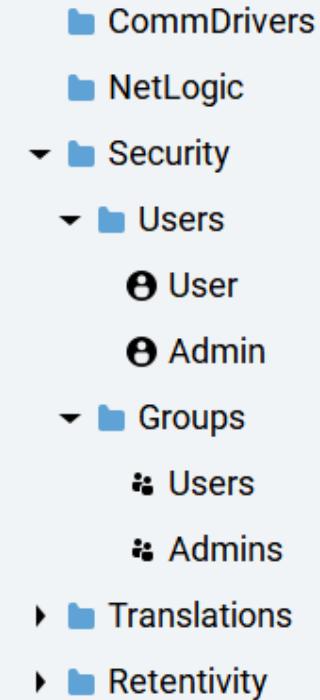
# Create users and groups

- User/Group Creation:
  - Users and Groups can be created manually, by the right click on Users or Groups folders (except for Local/Domain users)
- User properties:
  - Password
  - Localelds: a combination of Regional Settings, Language and Measurement System
  - Languages: Translation table column to be used (overrides the default language of the Localeld)
  - Measurement system: International System, British imperial or US Customary (overrides the default Measurement system of the Localeld)
  - Domain: Domain name used in case of authentication mode: Domain only, Domain and Local, or Any (it is automatically populated after a successful login) – blank by default



# Assign users to groups (and viceversa)

- Assignment must be done through the wizard "manage users and groups"
- The wizard allows assigning users to groups but also groups to users



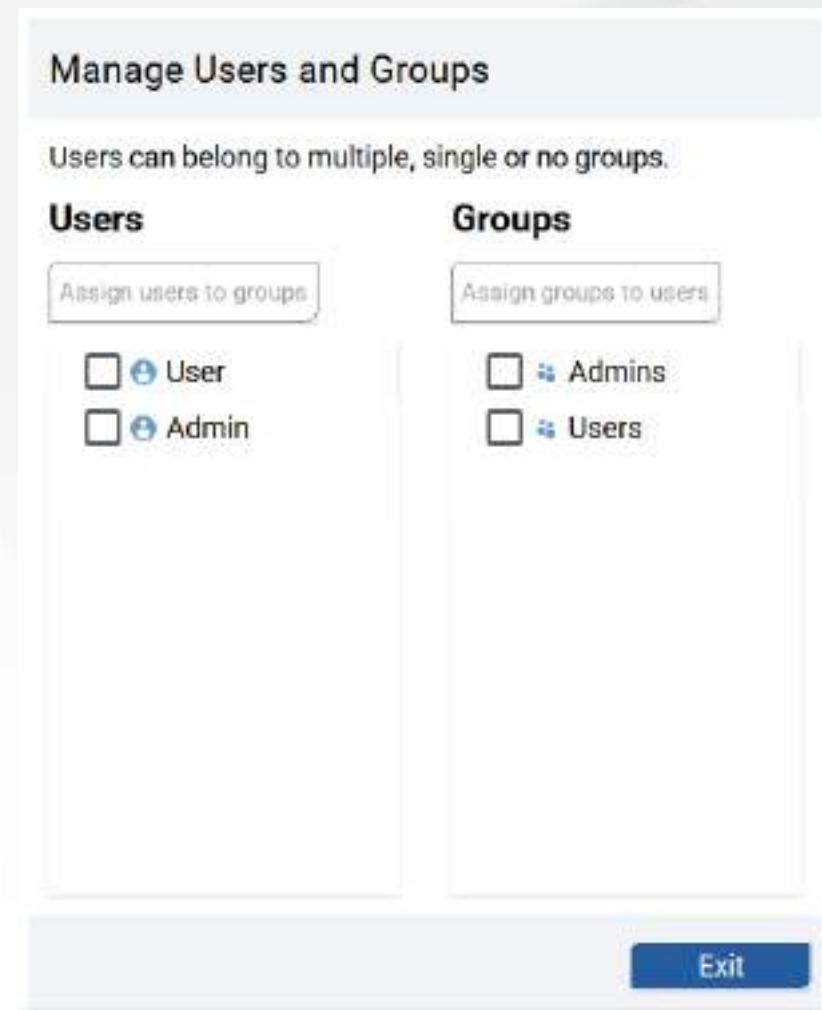
# Assign users to groups (and viceversa)

- Assignment must be done through the wizard "manage users and groups"
- The wizard allows assigning users to groups but also groups to users



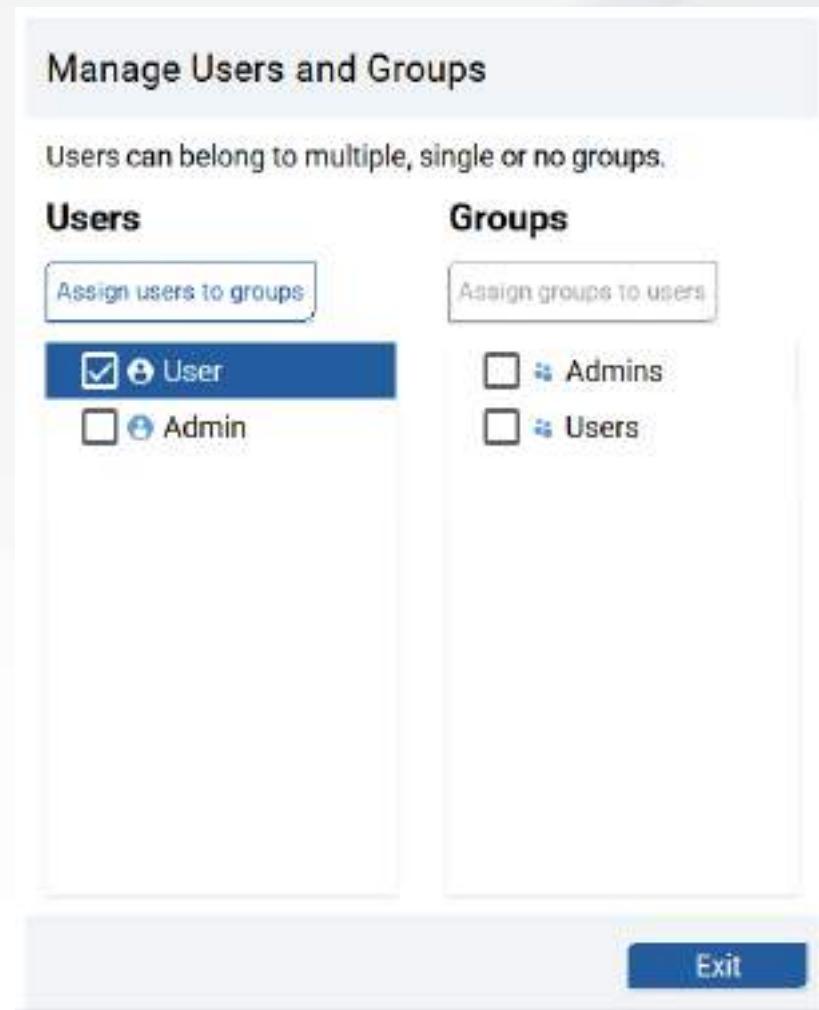
# Assign users to groups (and viceversa)

- Assignment must be done through the wizard "manage users and groups"
- The wizard allows assigning users to groups but also groups to users



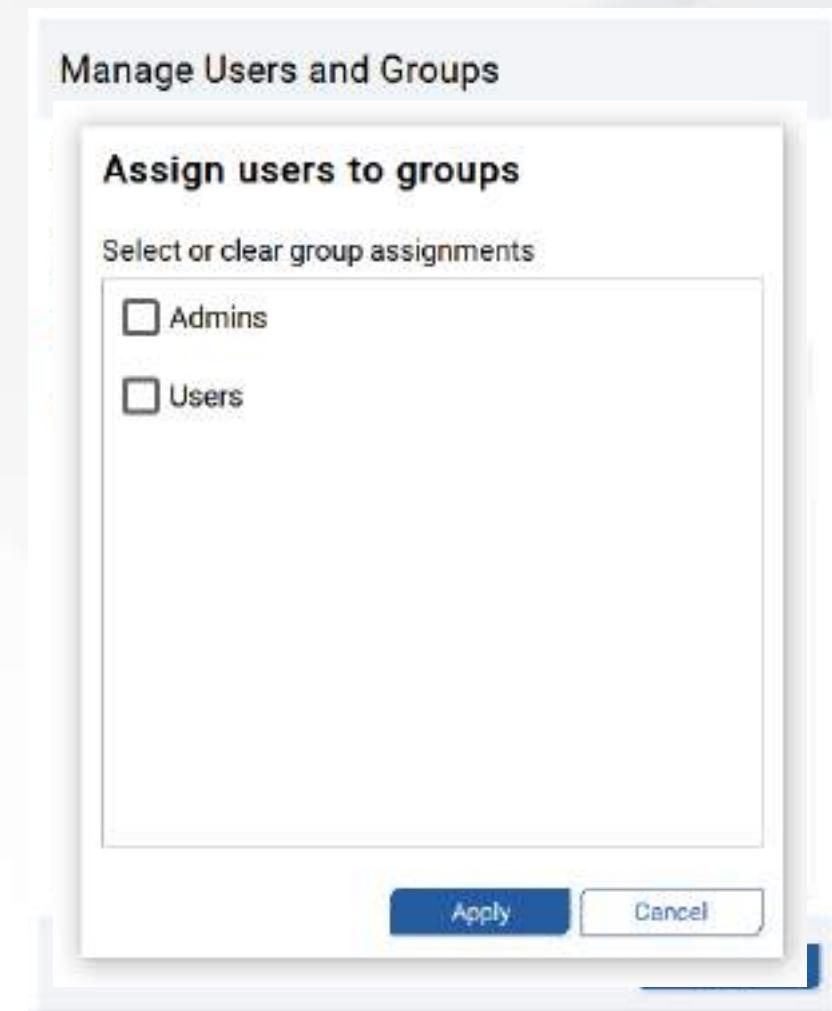
# Assign users to groups (and viceversa)

- Assignment must be done through the wizard "manage users and groups"
- The wizard allows assigning users to groups but also groups to users



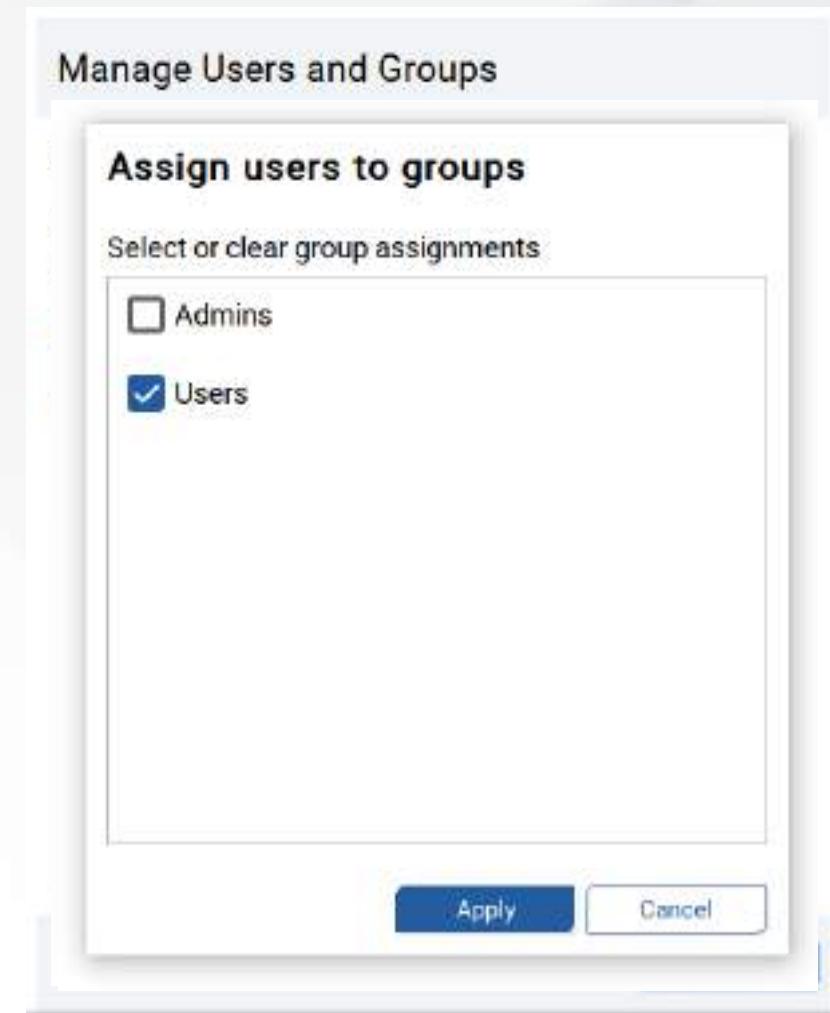
# Assign users to groups (and viceversa)

- Assignment must be done through the wizard "manage users and groups"
- The wizard allows assigning users to groups but also groups to users



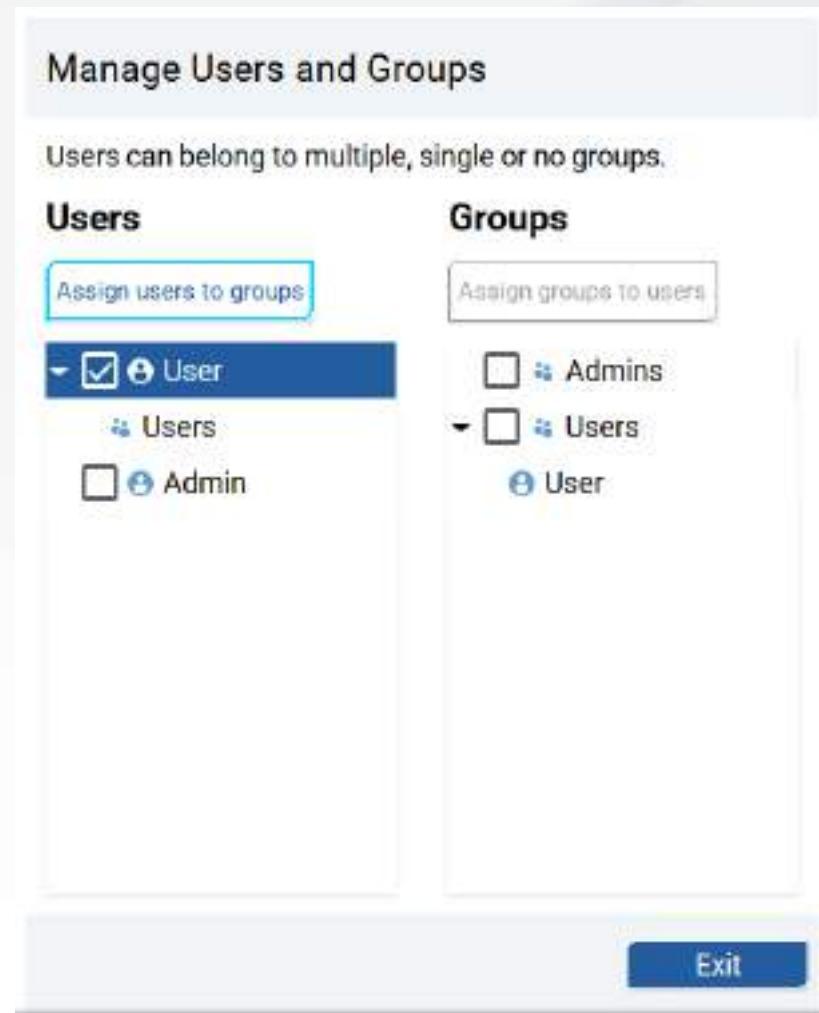
# Assign users to groups (and viceversa)

- Assignment must be done through the wizard "manage users and groups"
- The wizard allows assigning users to groups but also groups to users



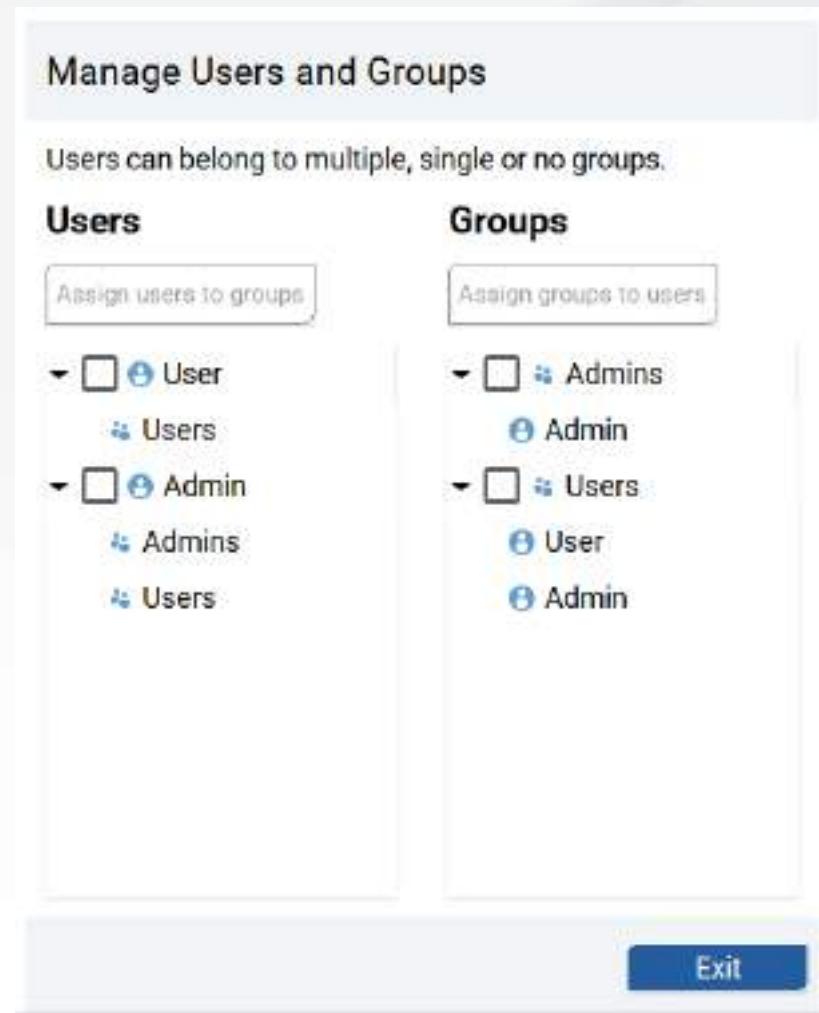
# Assign users to groups (and viceversa)

- Assignment must be done through the wizard "manage users and groups"
- The wizard allows assigning users to groups but also groups to users



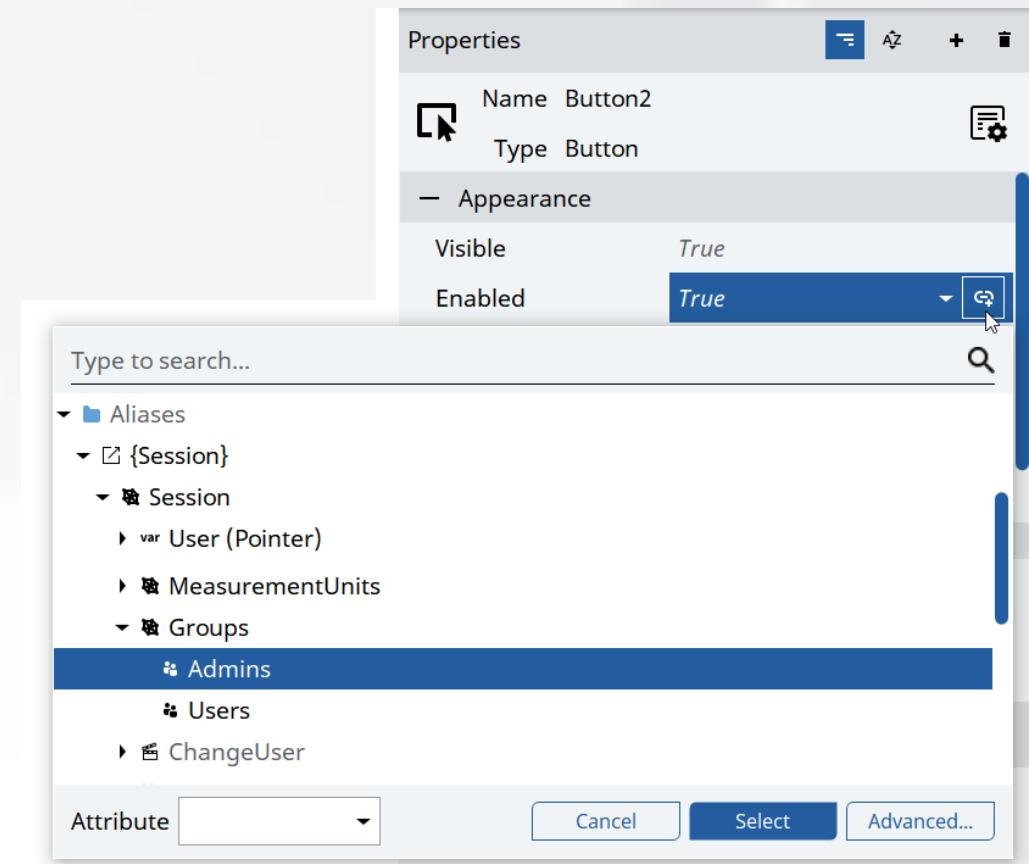
# Assign users to groups (and viceversa)

- Assignment must be done through the wizard "manage users and groups"
- The wizard allows assigning users to groups but also groups to users



# Define security into UI

- It's possible to make an object
  - "Visible/Invisible"
  - "Enabled/Disabled"
- just linking a Group to the needed property  
{Session}>Session>Groups> ...
- In this example "Button2" will be "Enabled" only for users of group "Admins"



# Manage user access at runtime

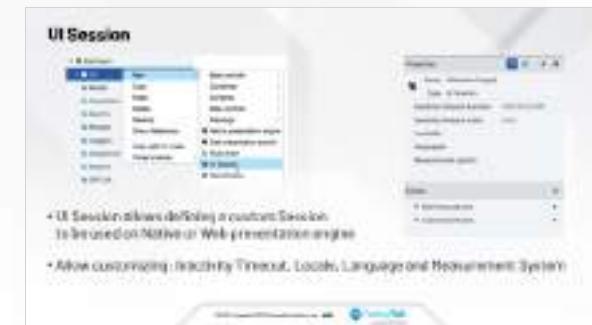
- Login options

- using the **widget "LoginForm"** of Template Library.  
Widget comes as a folder that contains the "LoginForm" to be placed in a Panel or called by a Dropdown button
- using the **method "Aliases > {Session} > Session > Login"**  
or "Aliases > {Session} > Session > Change User"
- using the **method "Commands > Core Commands > Change User"**



- Logout options

- using the **method "Aliases > {Session} > Session > Logout"**
- using an **UISession object** and use the IdleTimeoutEvent to call the method "UI Session > Logout" or "UI Session > Change User"
- using the **script "Idle Timeout"** available in Template Library, place into the MainWindow and associate a "Logout" command



# Manage user access at runtime

- Login options

- using the **widget "LoginForm"** of Template Library.  
Widget comes as a folder that contains the "LoginForm" to be placed in a Panel or called by a Dropdown button
- using the **method "Aliases > {Session} > Session > Login"** or "Aliases > {Session} > Session > Change User"
- using the **method "Commands > Core Commands > Change User"**

- Logout options

- using the **method "Aliases > {Session} > Session > Logout"**
- using an **UISession object** and use the IdleTimeoutEvent to call the method "UI Session > Logout" or "UI Session > Change User"
- using the **script "Idle Timeout"** available in Template Library, place into the MainWindow and associate a "Logout" command



# Manage user access at runtime

- Login options

- using the **widget "LoginForm"** of Template Library.  
Widget comes as a folder that contains the "LoginForm" to be placed in a Panel or called by a Dropdown button
  - using the **method "Aliases > {Session} > Session > Login"**  
or "Aliases > {Session} > Session > Change User"
  - using the **method "Commands > Core Commands > Change User"**

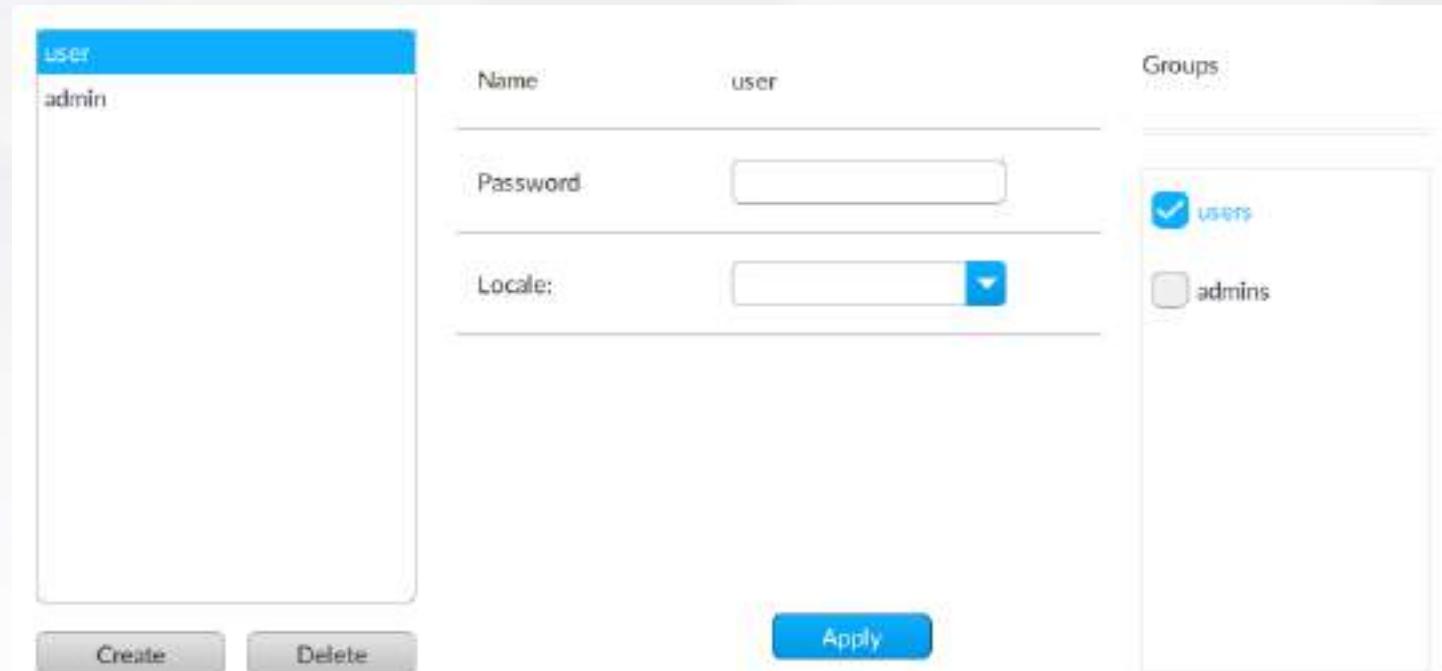
- Logout options

- using the **method "Aliases > {Session} > Session > Logout"**
  - using an **UI Session object** and use the IdleTimeoutEvent to call the method "UI Session > Logout" or "UI Session > Change User"
  - using the **script "Idle Timeout"** available in Template Library, place into the MainWindow and associate a "Logout" command



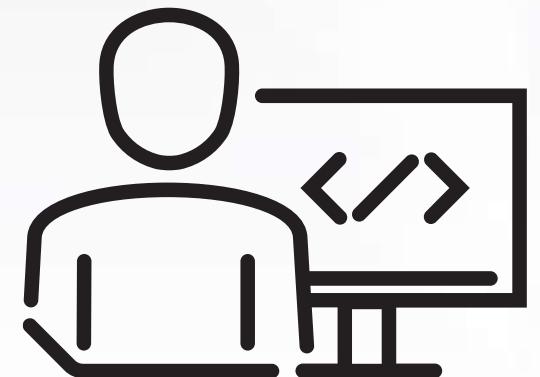
# Edit users at runtime

- using the **widget "User Editor"** of Template Library
- Widget, comes as a folder that contains the "UserEditorOverview" to be placed in a Panel
- It is required to link Users and Groups folder to the widget



# Hands-on session

- Create a couple of different users, and groups and make "user - group" association
- Define security at some UI objects (Visibility/Enabling)
- Add the Login form to a Panel
- Optional: call the Login form from a dropdown button  
paying attention to the background...
- Try at runtime with different users

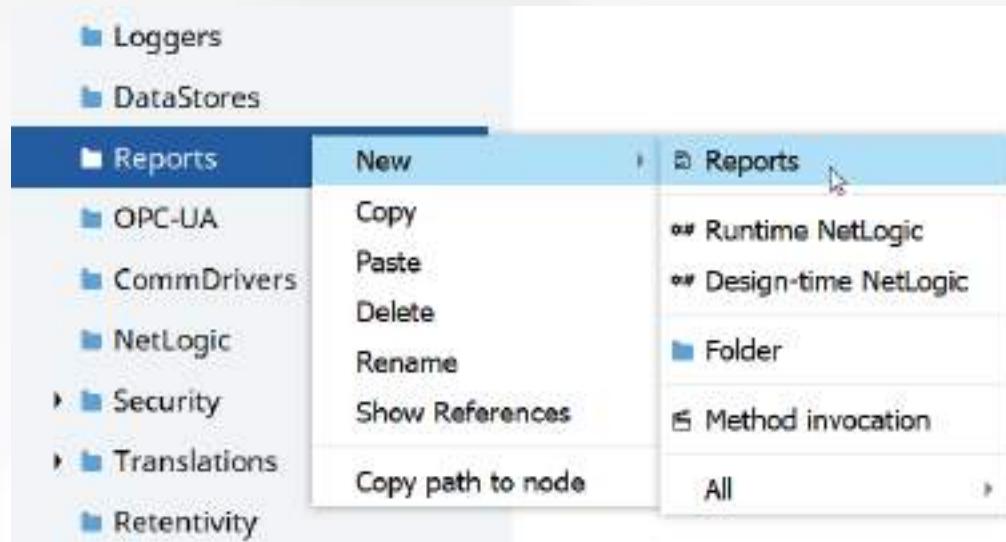


# Configure Reports

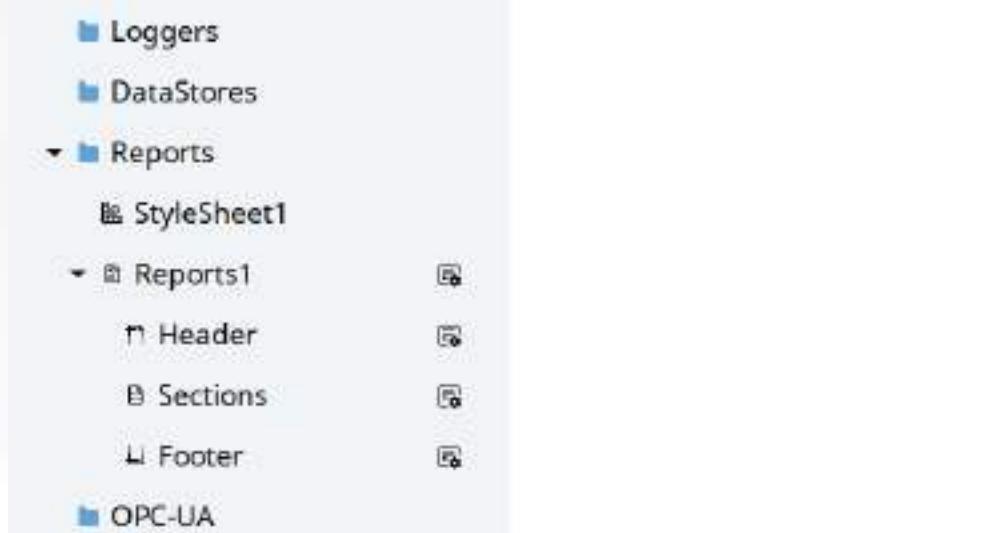


# Reports

- Design and generate reports in PDF format
- Data coming from FTOptixApplication and/or from a database
- Components:
  - StyleSheet
  - Header
  - Sections
  - Footer



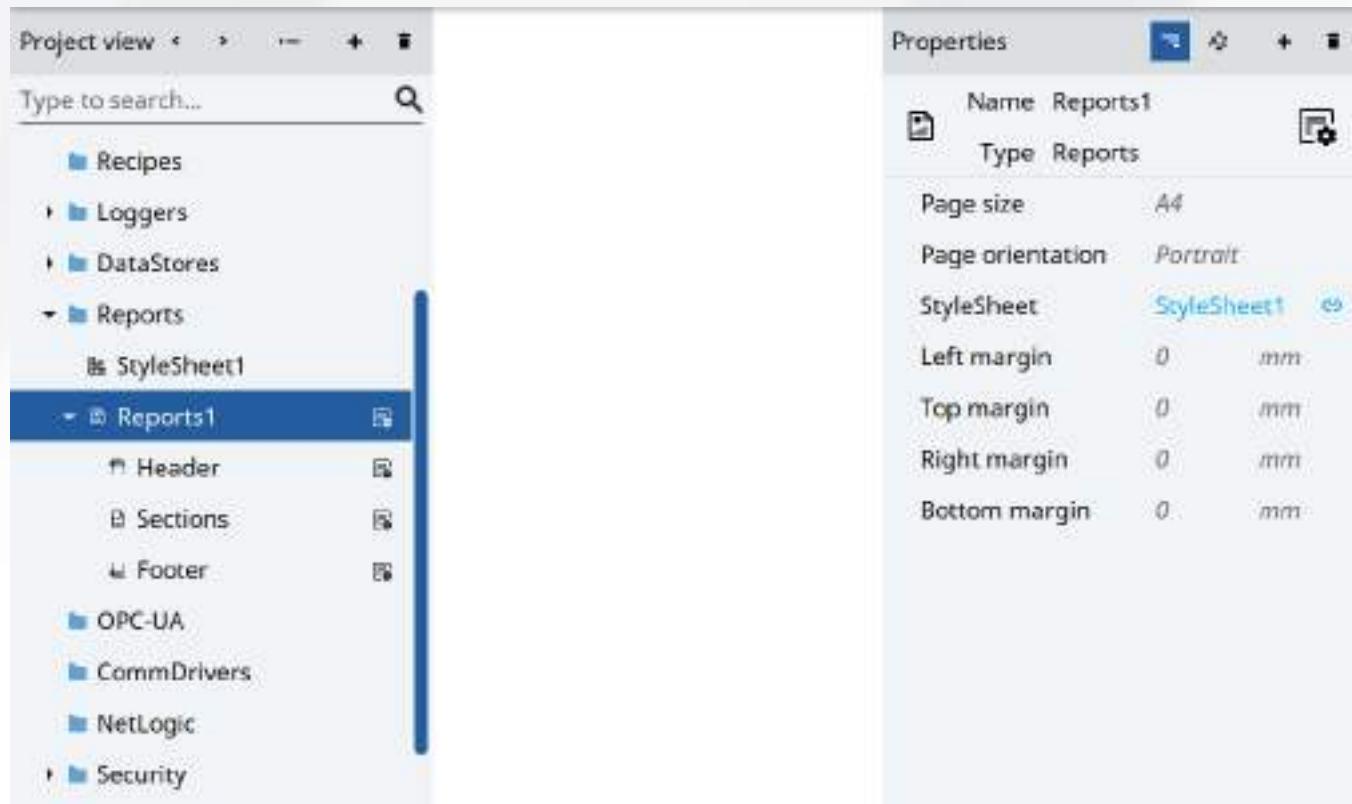
# Reports

- Design and generate reports in PDF format
  - Data coming from FTOptixApplication and/or from a database
  - Components:
    - StyleSheet
    - Header
    - Sections
    - Footer
- 
- The screenshot shows a hierarchical tree structure of report components:
- Loggers
  - DataStores
  - Reports
    - StyleSheet1
    - Reports1
      - Header
      - Sections
      - Footer
    - OPC-UA

# Report properties

- Allow setting the report's global layout properties, which apply to all pages of the final PDF document

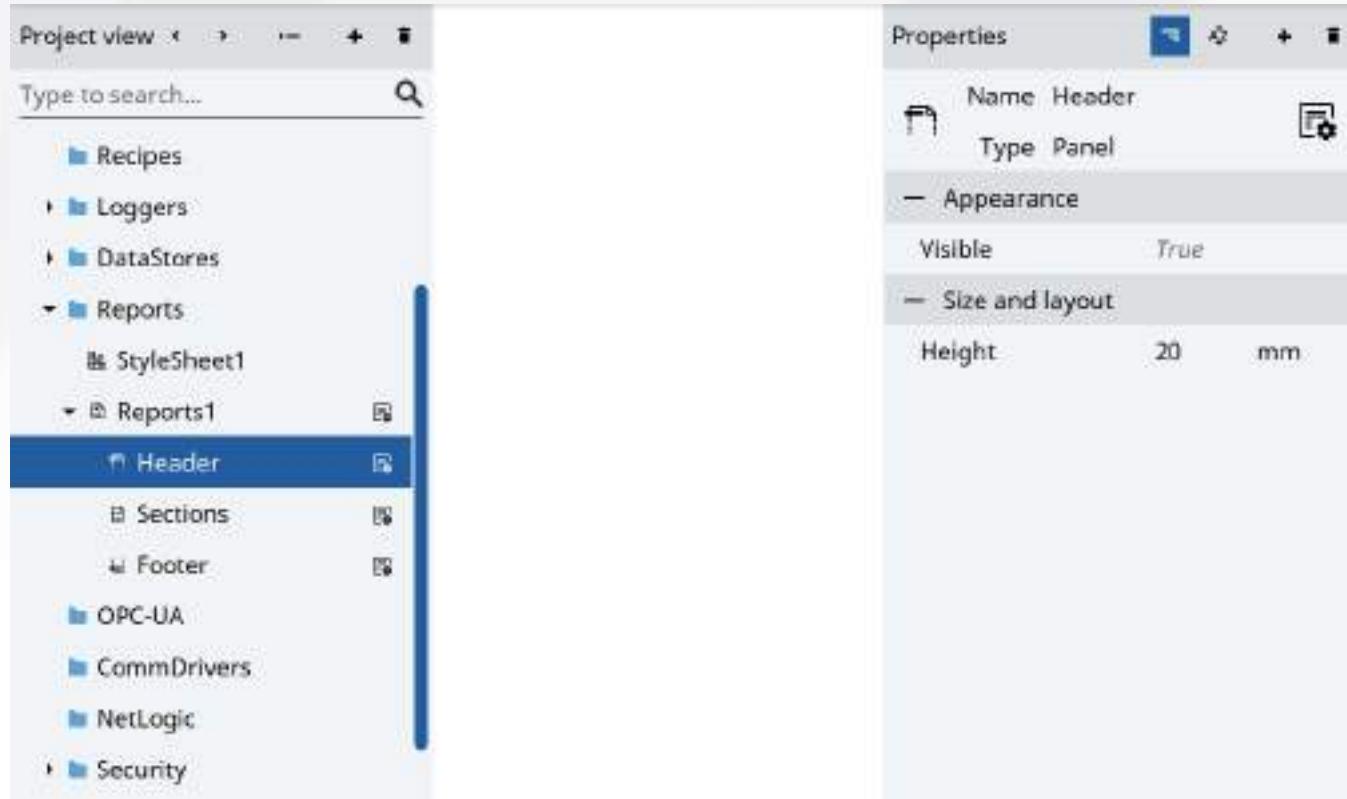
- Page size,
- Page orientation,
- Style sheet used,
- Page margins



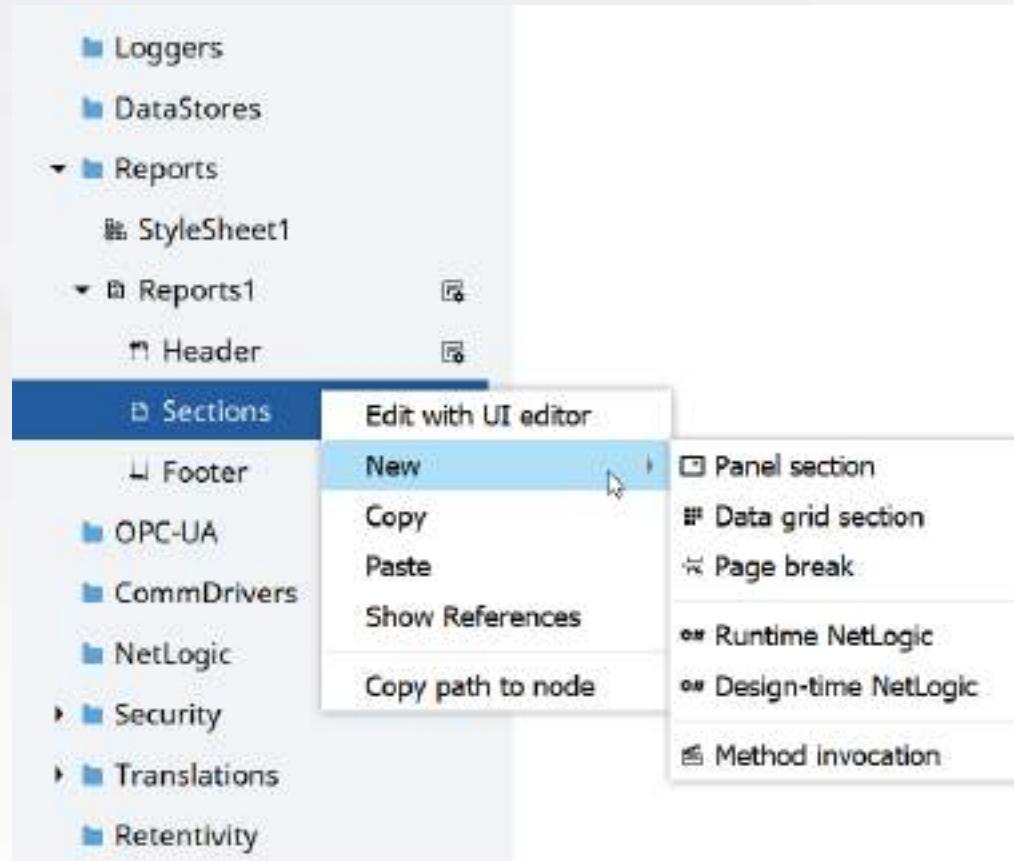
# Report header and footer

- Header/Footer is content that appears at the top/bottom of a report page
- Properties:

- Visible
- Height

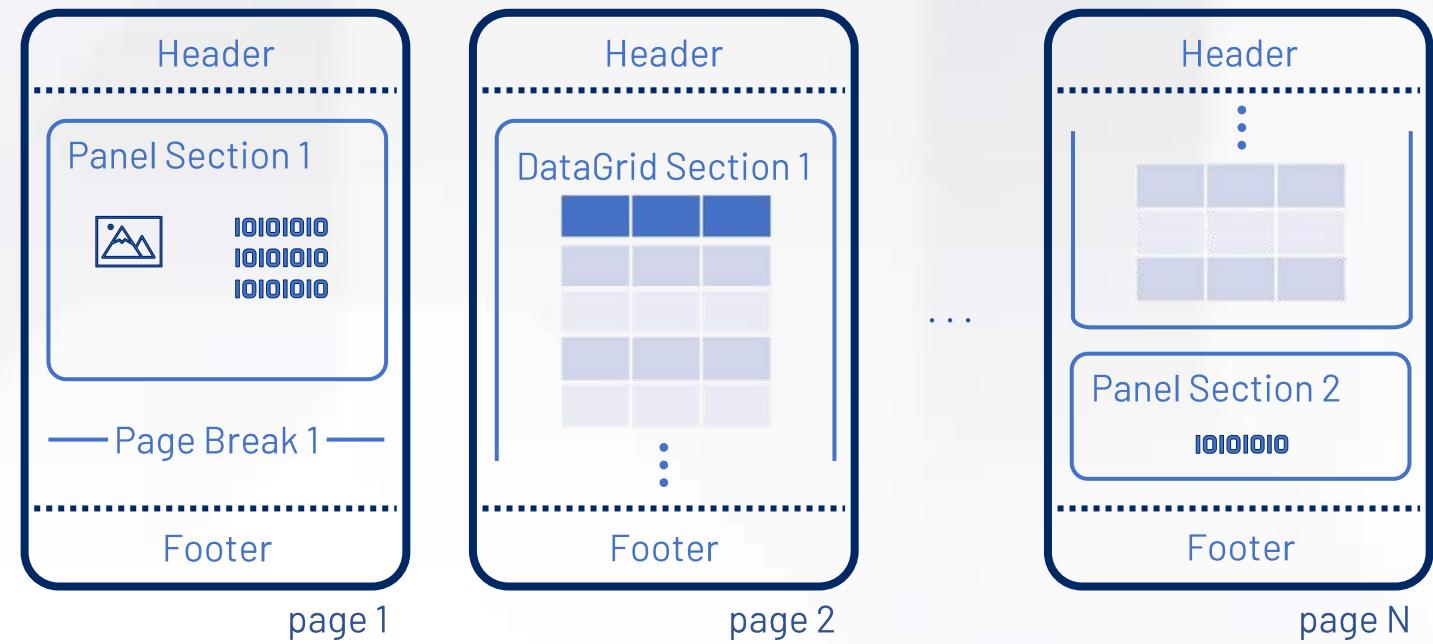
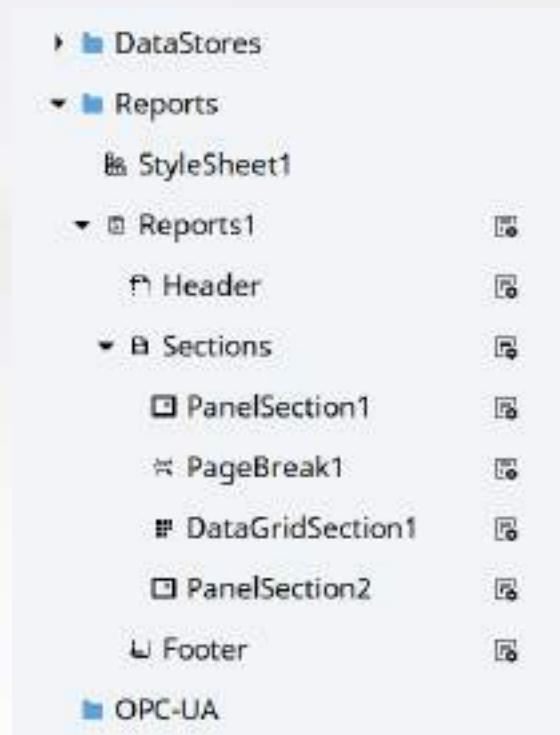


# Report sections



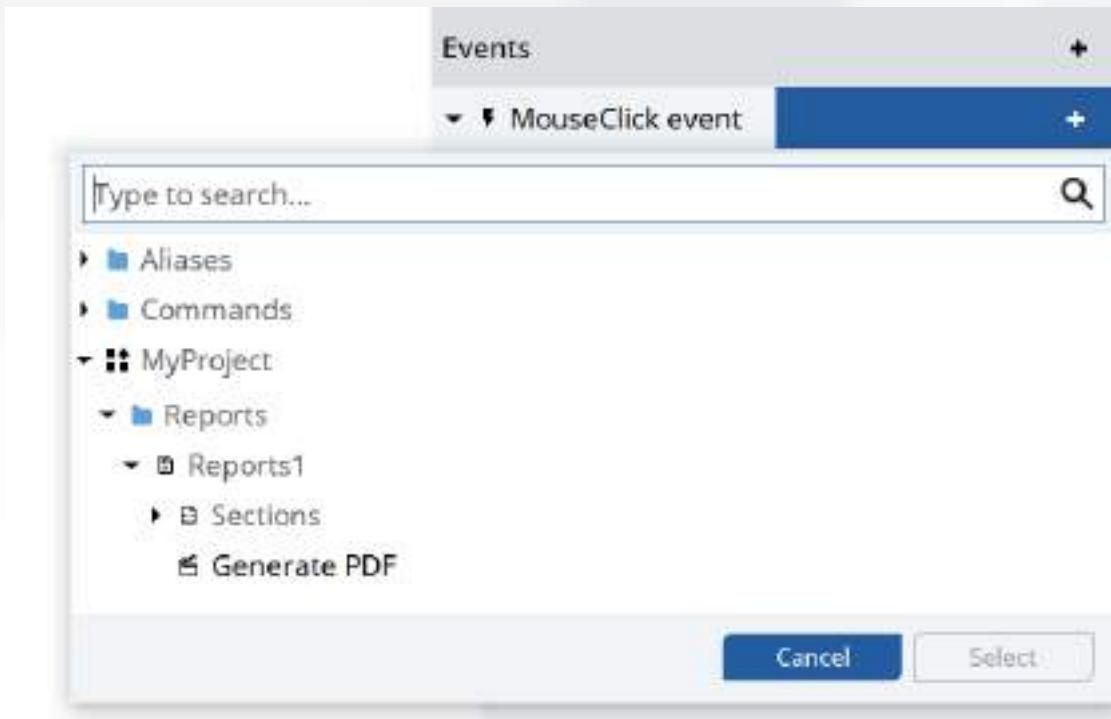
- **Panel:** container of graphical objects as Label, Rectangle, Panel, Image. Can be used to show variable values.
- **DataGrid:** container of data read from a database in table format. Can be used to show historical data collected by a Datalogger or Alarms.
- **Page break:** placeholder that forces the report to continue to another page.

# Report sections example



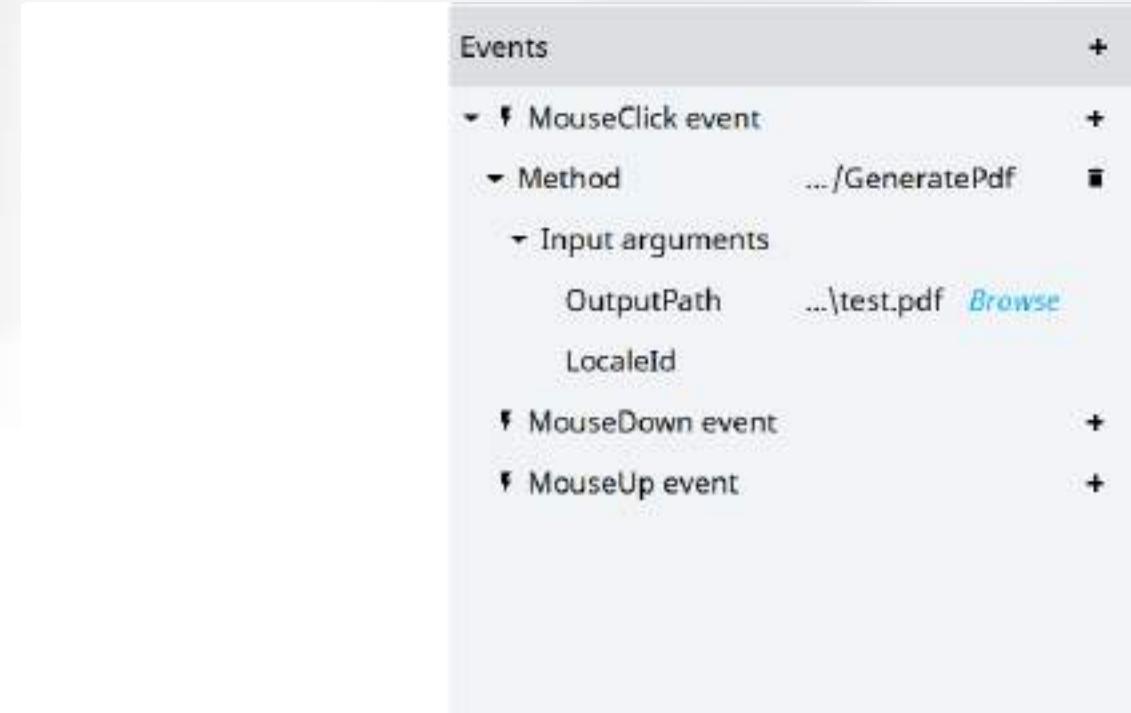
# Generate report at runtime

- Report resource expose method "Generate PDF"
- Can be called for example on Mouse Click event
- Properties:
  - Output Path: the path of the PDF file
  - Locale Id (optional): language to be used inside report



# Generate report at runtime

- Report resource expose method "Generate PDF"
- Can be called for example on Mouse Click event
- Properties:
  - Output Path: the path of the PDF file
  - Locale Id (optional): language to be used inside report

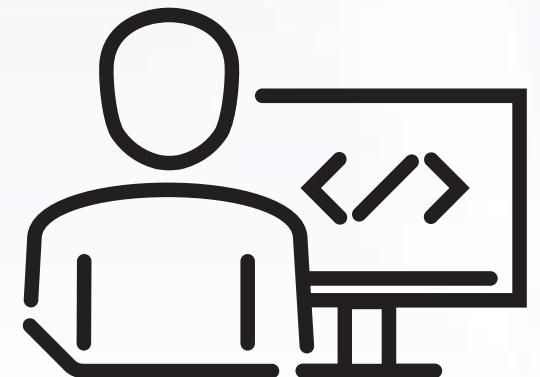


# Output Path

Shortcut	Description
%PROJECTDIR%	<p>A directory that contains files transferred at runtime (images, documents, and so on).</p> <p>Example: <code>C:\Users\UserName\Documents\Rockwell Automation\FactoryTalk Optix \ProjectName</code></p>
%APPLICATIONDIR%	<p>A directory that contains objects used at runtime, including these objects:</p> <ul style="list-style-type: none"><li>• Embedded database</li><li>• Retentivity storage</li><li>• Exported files</li></ul>
%USBN% , where N is a number associated with the USB device.	A path to a specific USB device connected to the target device.

# Hands-on session

- Define a report that includes a Datagrid to show values from a datalogger table
- Add a button that calls the "Generate PDF" method



# Set Locale, Languages and Measurement System



# Localization

- a Locale is **a set of User Interface settings** based on country

Locale	en-US	it-IT
Regional Settings	mm/dd/yyyy ... 10.5	dd/mm/yyyy ... 10,5
Language/Translations	Hello	Ciao
Measurement System	°F	°C

- Locale, Language and Measurement System are **related to the Session**
  - A session is a Runtime context in which a user executes operations
  - Examples: a session of Native Presentation Engine can look different from a session of Web Presentation Engine.  
A session of User1 can look different from a session of User2

# Locale

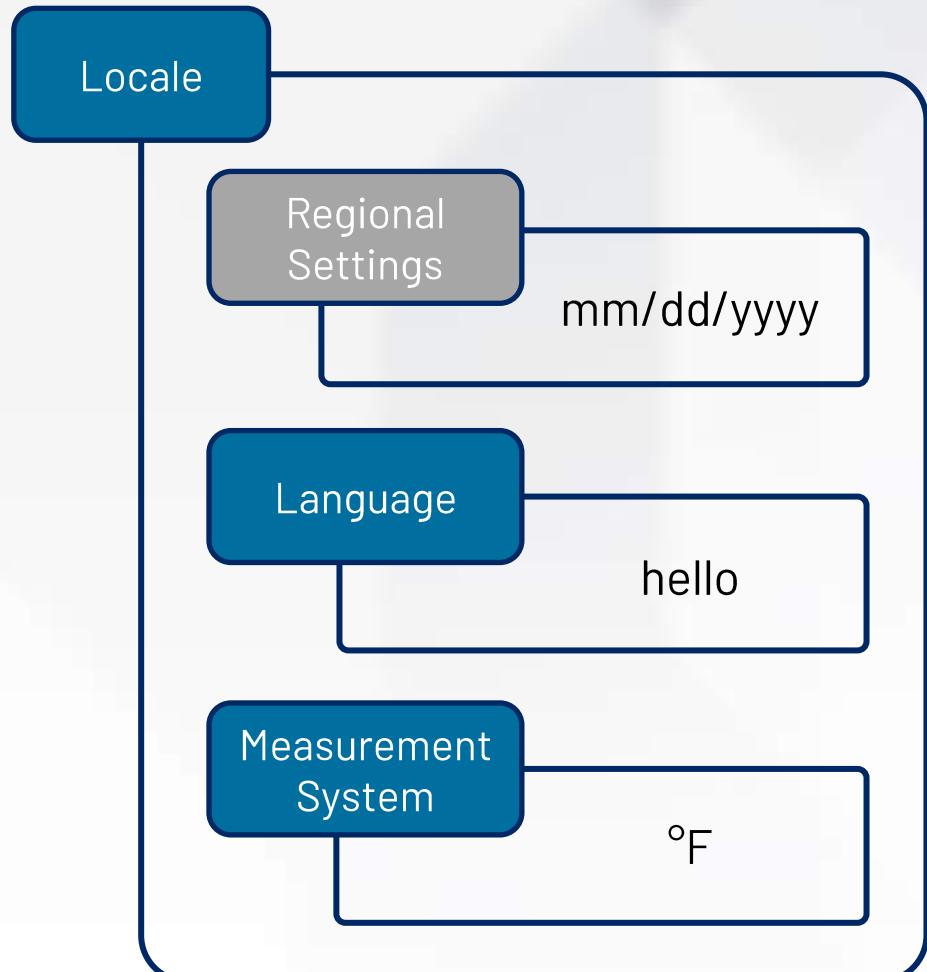
- Changing Locale will set altogether:

- Regional Settings
  - Language
  - Measurement System

- How to change the Locale at Runtime

- Method
    - Command > Variable commands > Set variable value
  - Variable to Modify
    - Aliases > {Session} > Session > ActualLocaleId
  - Value
    - Select the Locale ISO code

- Language and Measurement System can be set independently from Locale



# Locale

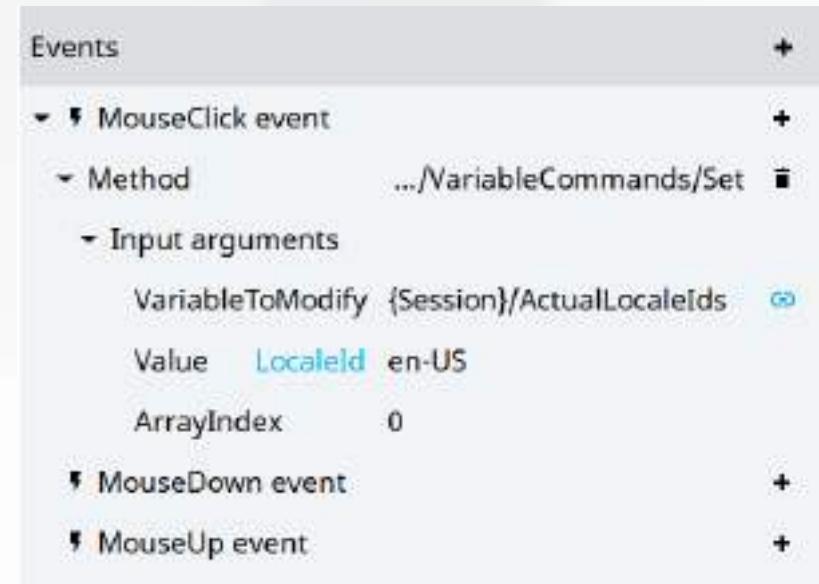
- Changing Locale will set altogether:

- Regional Settings
- Language
- Measurement System

- How to change the Locale at Runtime

- Method
  - Command > Variable commands > Set variable value
- Variable to Modify
  - Aliases > {Session} > Session > ActualLocaleIds
- Value
  - Select the Locale ISO code

- Language and Measurement System can be set independently from Locale



# Languages - Translations

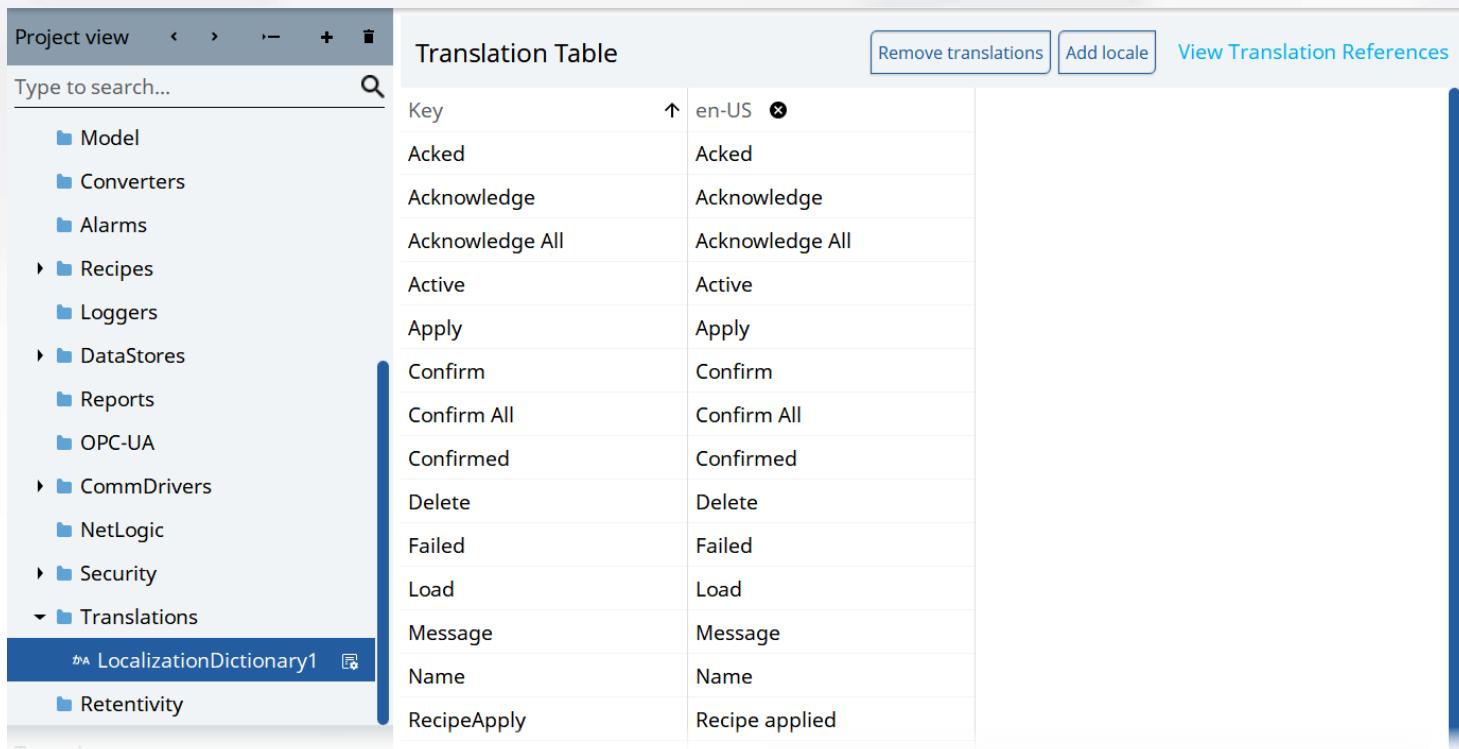
- The translation editor makes it possible to translate strings used in multilingual projects

- Translation Table**

- shows the translation of strings (Keys) in different languages (Locales)

- Translation Key References**

- shows all strings of the project that can be synched with the Translation Table (to become a key)



The screenshot shows the FactoryTalk Translation Table interface. On the left is a navigation tree under 'Project view' with categories like Model, Converters, Alarms, Recipes, Loggers, DataStores, Reports, OPC-UA, CommDrivers, NetLogic, Security, and Translations. Under Translations, 'LocalizationDictionary1' is selected. At the bottom of the tree, 'Retentivity' is listed. The main area is titled 'Translation Table' and contains a table with two columns: 'Key' and 'en-US'. The table lists various terms and their corresponding translations:

Key	en-US
Acked	Acked
Acknowledge	Acknowledge
Acknowledge All	Acknowledge All
Active	Active
Apply	Apply
Confirm	Confirm
Confirm All	Confirm All
Confirmed	Confirmed
Delete	Delete
Failed	Failed
Load	Load
Message	Message
Name	Name
RecipeApply	Recipe applied

# Languages - Translations

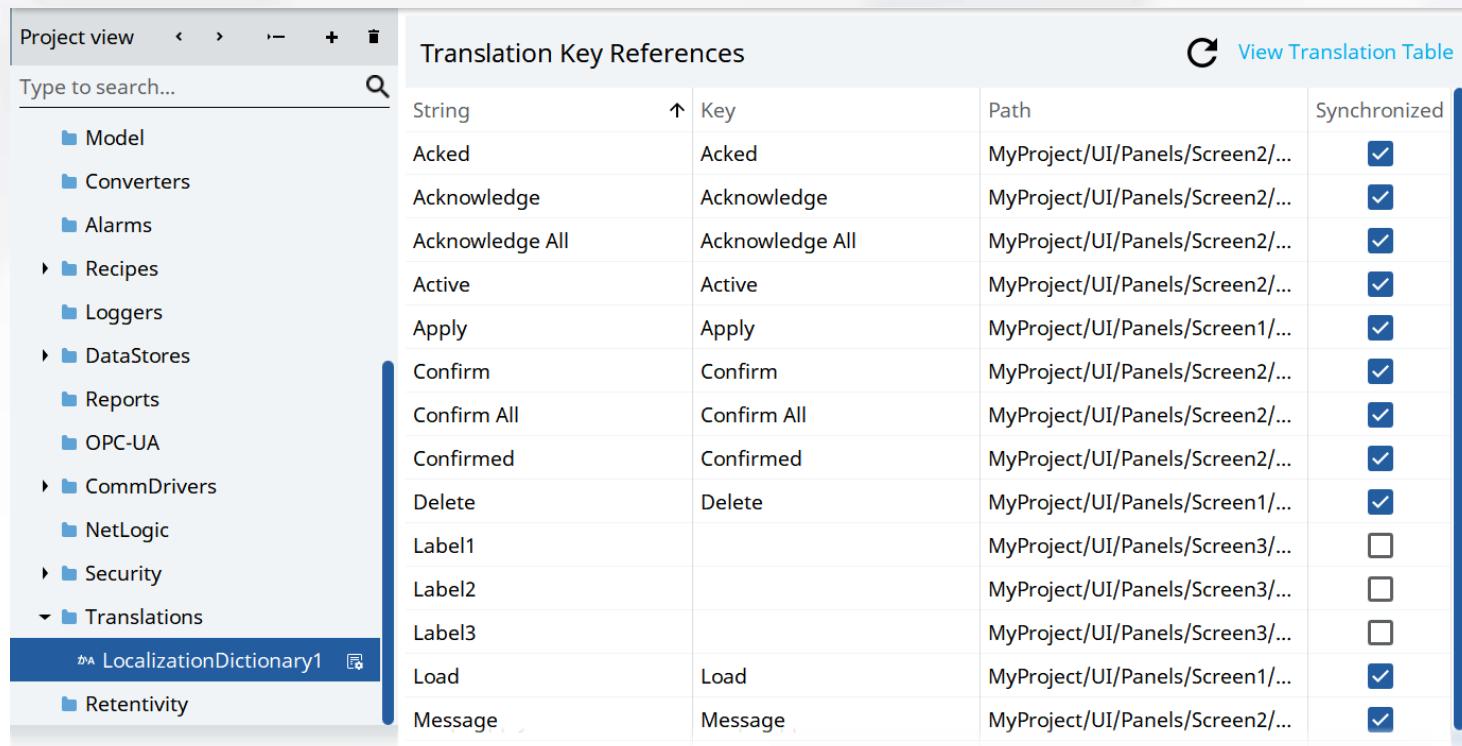
- The translation editor makes it possible to translate strings used in multilingual projects

## • Translation Table

- shows the translation of strings (Keys) in different languages (Locales)

## • Translation Key References

- shows all strings of the project that can be synched with the Translation Table (to become a key)



The screenshot shows the FactoryTalk Translation Key References interface. On the left is a tree view of project components: Model, Converters, Alarms, Recipes, Loggers, DataStores, Reports, OPC-UA, CommDrivers, NetLogic, Security, and Translations. Under Translations, there are two items: LocalizationDictionary1 and Retentivity. The main area is titled "Translation Key References" and contains a table with columns: String, Key, Path, and Synchronized. The table lists various strings and their corresponding keys, along with their paths in the project and synchronization status.

String	Key	Path	Synchronized
Acked	Acked	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Acknowledge	Acknowledge	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Acknowledge All	Acknowledge All	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Active	Active	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Apply	Apply	MyProject/UI/Panels/Screen1/...	<input checked="" type="checkbox"/>
Confirm	Confirm	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Confirm All	Confirm All	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Confirmed	Confirmed	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Delete	Delete	MyProject/UI/Panels/Screen1/...	<input checked="" type="checkbox"/>
Label1		MyProject/UI/Panels/Screen3/...	<input type="checkbox"/>
Label2		MyProject/UI/Panels/Screen3/...	<input type="checkbox"/>
Label3		MyProject/UI/Panels/Screen3/...	<input type="checkbox"/>
Load	Load	MyProject/UI/Panels/Screen1/...	<input checked="" type="checkbox"/>
Message	Message	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>

# Translation table

- Remove Translations

- to remove one or more keys from the table

- Add locale

1. Click to add a new language
2. Select the language in a standard Language-Country ISO code
3. Confirm adding the new language to the Project properties

- View Translation Reference

- switch to Key Reference table

Translation Table		<a href="#">Remove translations</a>	<a href="#">Add locale</a>	<a href="#">View Translation References</a>
Key	↑ en-US			
Acked	Acked			
Acknowledge	Acknowledge			
Acknowledge All	Acknowledge All			
Active	Active			
Apply	Apply			
Confirm	Confirm			
Confirm All	Confirm All			
Confirmed	Confirmed			
Delete	Delete			

- Import/Export

- Translation Table can be exported and imported back using the "Translation Importer Exporter" script available in Template Library

# Translation table

- Remove Translations

- to remove one or more keys from the table

- Add locale

1. Click to add a new language
2. Select the language in a standard Language-Country ISO code
3. Confirm adding the new language to the Project properties

- View Translation Reference

- switch to Key Reference table

Translation Table		Remove translations	Add locale	View Translation References
Key	en-US		Select the new locale...	
Acked	Acked		it-JT	
Acknowledge	Acknowledge		it-CH	
Acknowledge All	Acknowledge All		it-IT	
Active	Active		iu-CA	
Apply	Apply			
Confirm	Confirm			
Confirm All	Confirm All			
Confirmed	Confirmed			
Delete	Delete			

- Import/Export

- Translation Table can be exported and imported back using the "Translation Importer Exporter" script available in Template Library

# Translation table

- Remove Translations

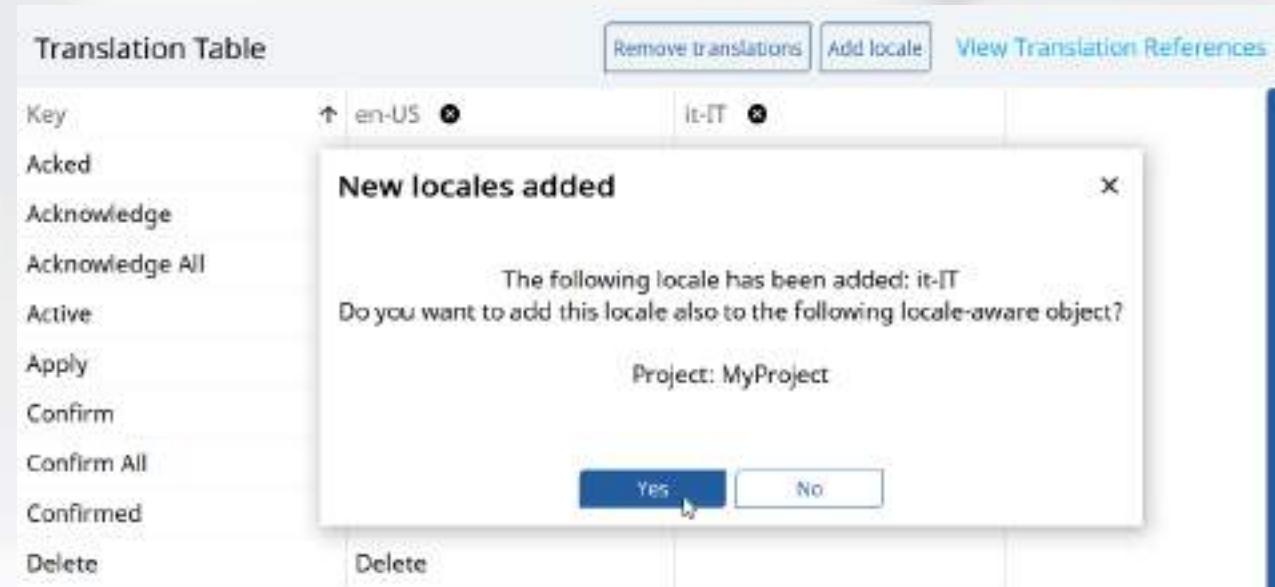
- to remove one or more keys from the table

- Add locale

1. Click to add a new language
2. Select the language in a standard Language-Country ISO code
3. Confirm adding the new language to the Project properties

- View Translation Reference

- switch to Key Reference table



- Import/Export

- Translation Table can be exported and imported back using the "Translation Importer Exporter" script available in Template Library

# Translation key references

- Key Reference table

- shows all strings of the project

- Setting "Synchronized" to true

- creates a key into Translation Table with the same text

- copy the key text into the default language (en-US)

Translation Key References		View Translation Table	
String	Key	Path	Synchronized
Acked	Acked	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Acknowledge	Acknowledge	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Acknowledge All	Acknowledge All	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Active	Active	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Apply	Apply	MyProject/UI/Panels/Screen1/...	<input checked="" type="checkbox"/>
Confirm	Confirm	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Confirm All	Confirm All	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Confirmed	Confirmed	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Delete	Delete	MyProject/UI/Panels/Screen1/...	<input checked="" type="checkbox"/>
Label1		MyProject/UI/Panels/Screen3/...	<input type="checkbox"/>
Label2		MyProject/UI/Panels/Screen3/...	<input type="checkbox"/>
Label3		MyProject/UI/Panels/Screen3/...	<input type="checkbox"/>
Load	Load	MyProject/UI/Panels/Screen1/...	<input checked="" type="checkbox"/>
Message	Message	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>

# Translation key references

- Key Reference table

- shows all strings of the project

- Setting "Synchronized" to true

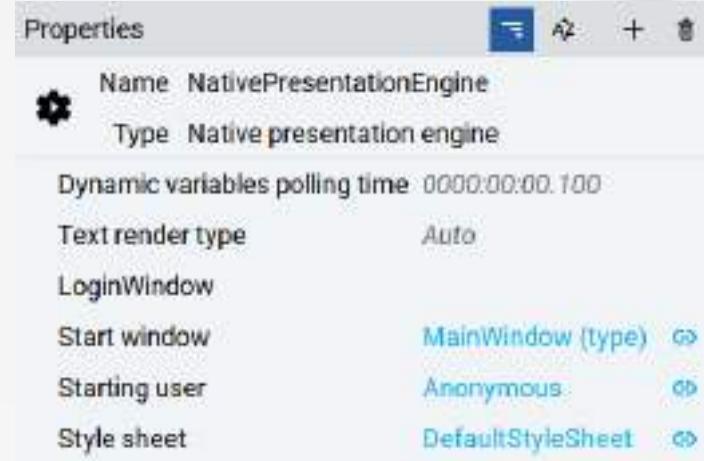
- creates a key into Translation Table with the same text

- copy the key text into the default language (en-US)

Translation Key References			
String	↑ Key	Path	Synchronized
Acked	Acked	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Acknowledge	Acknowledge	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Acknowledge All	Acknowledge All	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Active	Active	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Apply	Apply	MyProject/UI/Panels/Screen1/...	<input checked="" type="checkbox"/>
Confirm	Confirm	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Confirm All	Confirm All	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Confirmed	Confirmed	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>
Delete	Delete	MyProject/UI/Panels/Screen1/...	<input checked="" type="checkbox"/>
Label1	Label1	MyProject/UI/Panels/Screen3/...	<input checked="" type="checkbox"/>
Label2		MyProject/UI/Panels/Screen3/...	<input type="checkbox"/>
Label3		MyProject/UI/Panels/Screen3/...	<input type="checkbox"/>
Load	Load	MyProject/UI/Panels/Screen1/...	<input checked="" type="checkbox"/>
Message	Message	MyProject/UI/Panels/Screen2/...	<input checked="" type="checkbox"/>

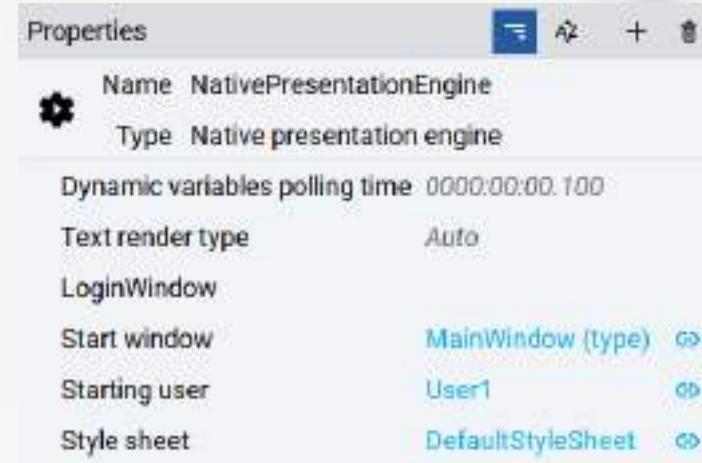
# Translation usage

- Two ways to set the "starting" language:
  1. Language or Locale Id of the "Starting user"
  2. First language listed in the "Locales" project property
- How to change the language at Runtime
  - Method
    - Command > Variable commands > Set variable value
  - Variable to Modify
    - Aliases > {Session} > Session > ActualLanguages
  - Value
    - Select the language ISO code



# Translation usage

- Two ways to set the "starting" language:
  1. Language or Locale Id of the "Starting user"
  2. First language listed in the "Locales" project property
- How to change the language at Runtime
  - Method
    - Command > Variable commands > Set variable value
  - Variable to Modify
    - Aliases > {Session} > Session > ActualLanguages
  - Value
    - Select the language ISO code



# Translation usage

- Two ways to set the "starting" language:
  1. Language or Locale Id of the "Starting user"
  2. First language listed in the "Locales" project property



- How to change the language at Runtime
  - Method
    - Command > Variable commands > Set variable value
    - Variable to Modify
      - Aliases > {Session} > Session > ActualLanguages
  - Value
    - Select the language ISO code

# Translation usage

- Two ways to set the "starting" language:
  1. Language or Locale Id of the "Starting user"
  2. First language listed in the "Locales" project property
- How to change the language at Runtime
  - Method
    - Command > Variable commands > Set variable value
    - Variable to Modify
      - Aliases > {Session} > Session > ActualLanguages
  - Value
    - Select the language ISO code

Properties	
Name	MyProject
Type	Project folder
Locales	it-IT;en-US
Translation fallback locales	en-US
Branching enabled	False
Measurement systems map	Default mapping
Authentication mode	Model only
Default user folder	Users

# Translation usage

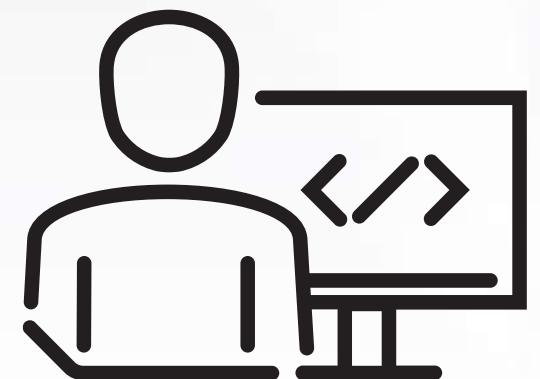
- Two ways to set the "starting" language:
  1. Language or Locale Id of the "Starting user"
  2. First language listed in the "Locales" project property
- How to change the language at Runtime
  - Method
    - Command > Variable commands > Set variable value
  - Variable to Modify
    - Aliases > {Session} > Session > ActualLanguages
  - Value
    - Select the language ISO code

Properties	
Name	MyProject
Type	Project folder
Locales	it-IT;en-US
Translation fallback locales	en-US
Branching enabled	False
Measurement systems map	Default mapping
Authentication mode	Model only
Default user folder	Users

Events	
MouseClick event	+
Method	.../VariableCommands/Set
Input arguments	
VariableToModify	{Session}/ActualLanguages
Value	LocaleId: en-US
ArrayIndex	0
MouseDown event	+
MouseUp event	+

# Hands-on session

- Use some strings to label objects around the project
- Import strings into the Translation Table using the Key Reference Table then edit the translation manually
- Change the language at Runtime with the Emulator
- Optionally
  - try to export the Translation table,
  - modify with an external editor,
  - import back in Optix Studio



# Measurement system

- Choose between:
  - International System of Units (SI)
  - US Customary measurement system
  - British Imperial units
- is enough to set the Unit of Measure on the Variable, then at Runtime, the conversion will be automatically applied depending on the Measurement System chosen



liter / quart



°C / °F

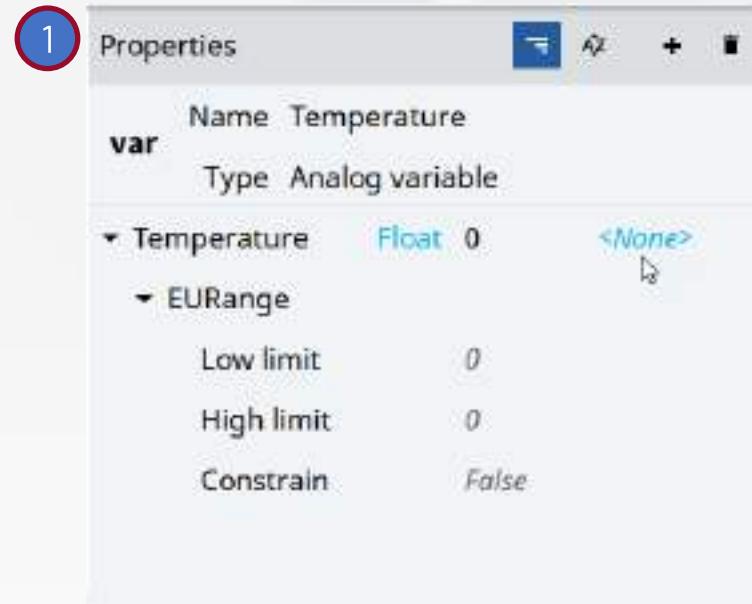


meter / yard

# Localize an Analog Variable

1. Define a new Analog Variable with Engineering Unit into the Model folder
2. Add an Analog Variable to the object
3. Into the object's Analog Variable set an Advanced Dynamic Link using "Binding mode between Source and Parent engineering unit"
4. Link the object's value to the object's Analog Variable

- Unit of measure can be defined directly within a Comm.Driver Tag as well



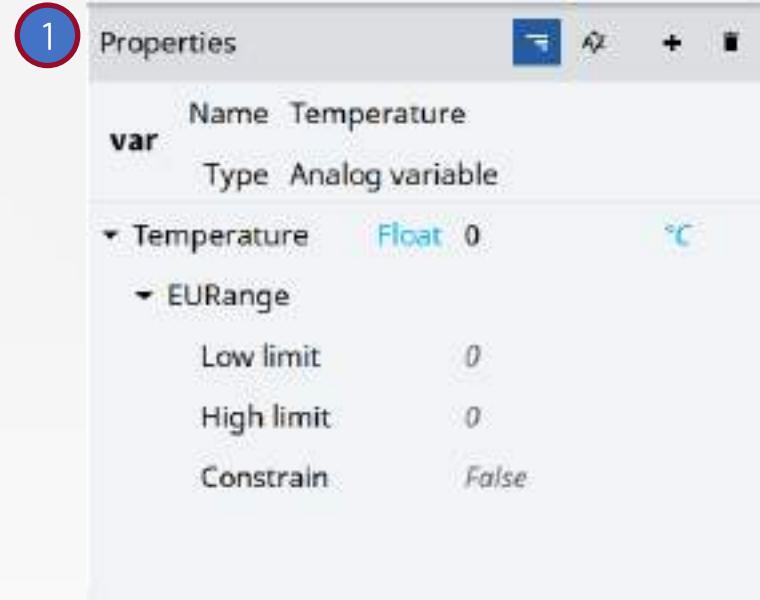
# Localize an Analog Variable

1. Define a new Analog Variable with Engineering Unit into the Model folder
  2. Add an Analog Variable to the object
  3. Into the object's Analog Variable set an Advanced Dynamic Link using "Binding mode between Source and Parent engineering unit"
  4. Link the object's value to the object's Analog Variable
- Unit of measure can be defined directly within a Comm.Driver Tag as well



# Localize an Analog Variable

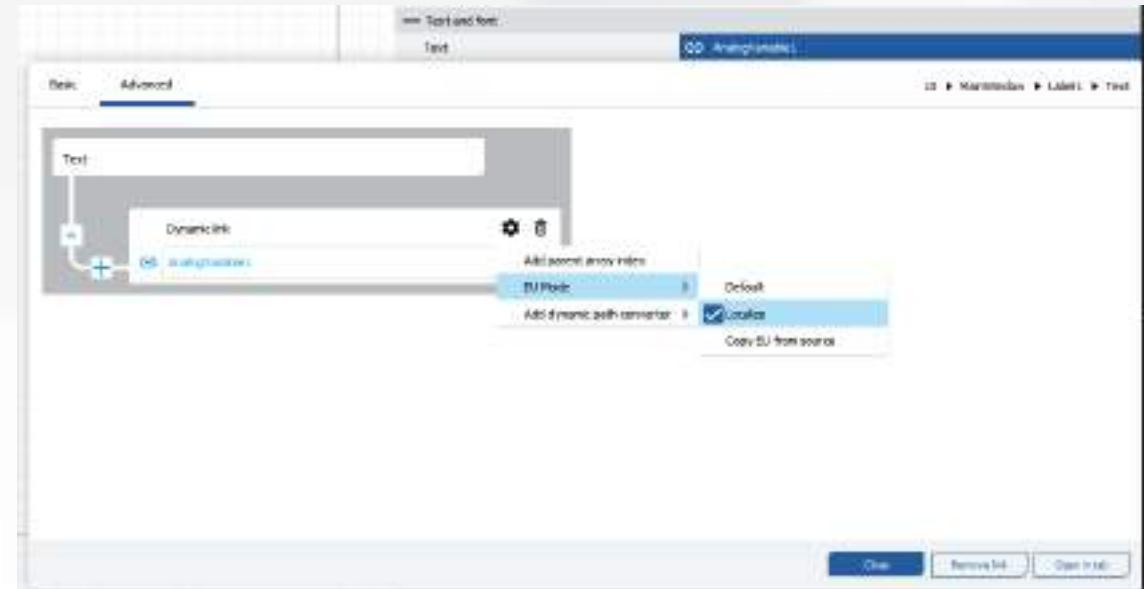
1. Define a new Analog Variable with Engineering Unit into the Model folder
  2. Add an Analog Variable to the object
  3. Into the object's Analog Variable set an Advanced Dynamic Link using "Binding mode between Source and Parent engineering unit"
  4. Link the object's value to the object's Analog Variable
- Unit of measure can be defined directly within a Comm.Driver Tag as well



# Localize an Analog Variable

1. Define a new Analog Variable with Engineering Unit into the Model folder
  2. Add an Analog Variable to the object
  3. Into the object's Analog Variable set an Advanced Dynamic Link using "Binding mode between Source and Parent engineering unit"
  4. Link the object's value to the object's Analog Variable
- Unit of measure can be defined directly within a Comm.Driver Tag as well

2



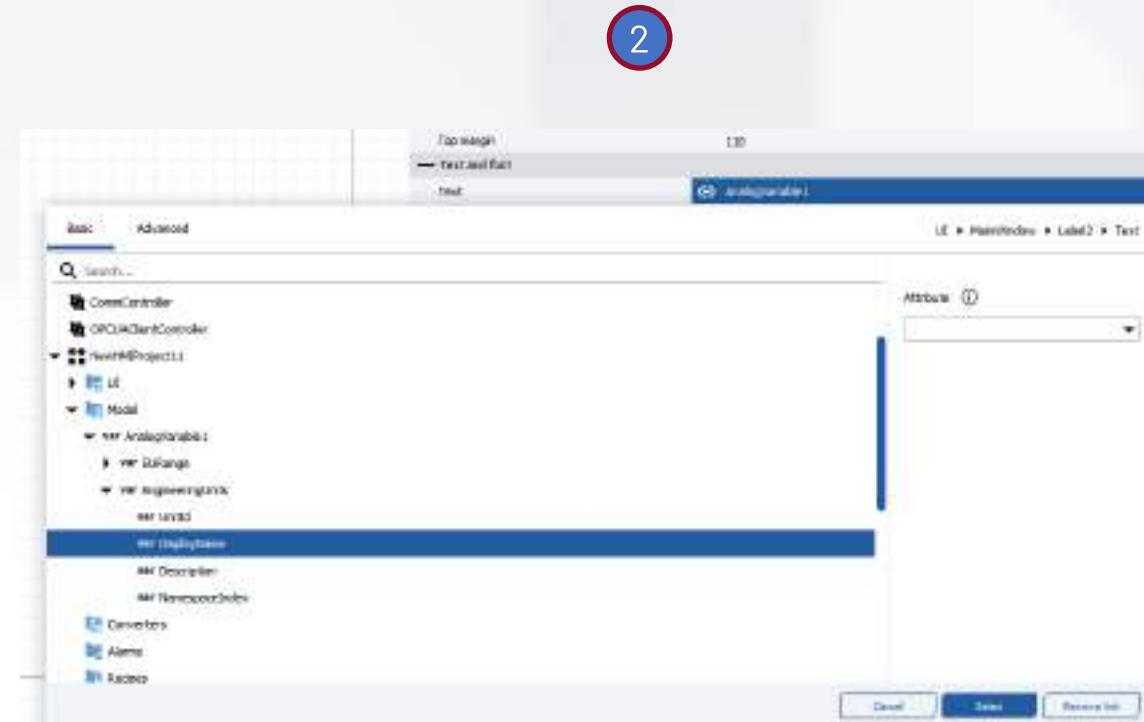
# Display and localize the Engineering Unit

1. Add a label to the screen and create a DynamicLink from the Analog Variable to the Text property of the label
2. Open the DynamicLink popup, expand the Analog Variable and select Engineering Unit > Display Name
3. Reopen the DynamicLink popup, move to the Advanced tab and localize the Engineering Unit



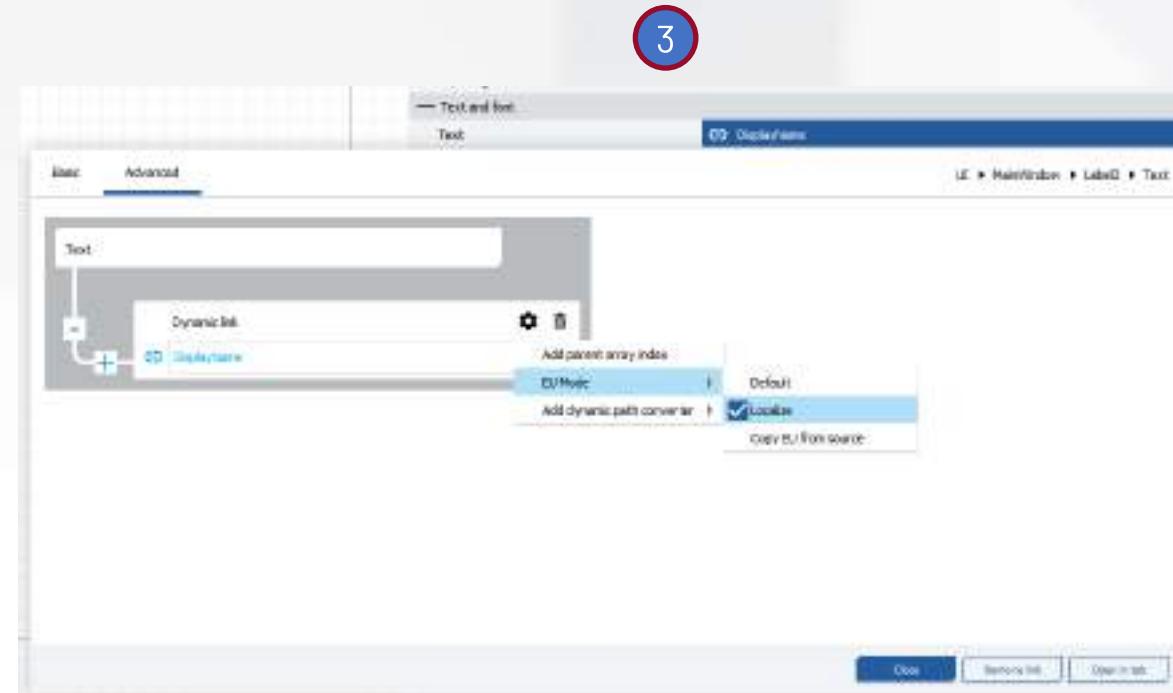
# Display and localize the Engineering Unit

1. Add a label to the screen and create a DynamicLink from the Analog Variable to the Text property of the label
2. Open the DynamicLink popup, expand the Analog Variable and select Engineering Unit > Display Name
3. Reopen the DynamicLink popup, move to the Advanced tab and localize the Engineering Unit



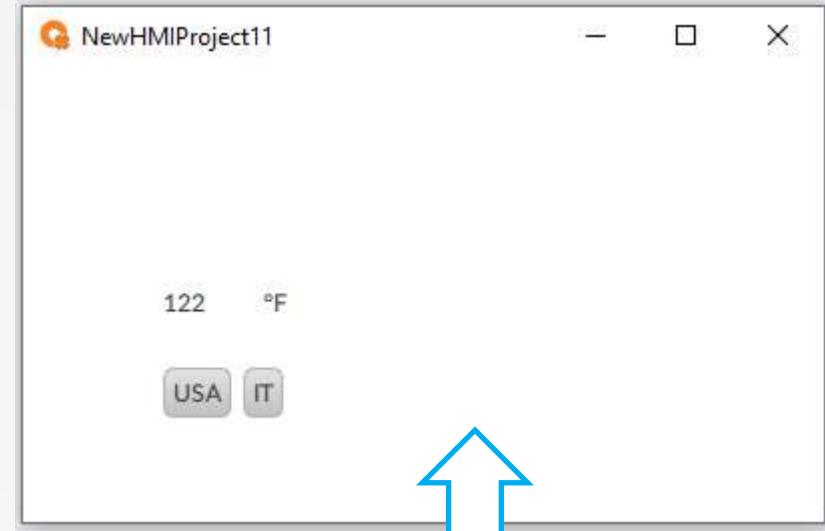
# Display and localize the Engineering Unit

1. Add a label to the screen and create a DynamicLink from the Analog Variable to the Text property of the label
2. Open the DynamicLink popup, expand the Analog Variable and select Engineering Unit > Display Name
3. Reopen the DynamicLink popup, move to the Advanced tab and localize the Engineering Unit



# Test the conversion

1. Add two buttons to change the Measurement System
2. Execute the runtime and check the conversion



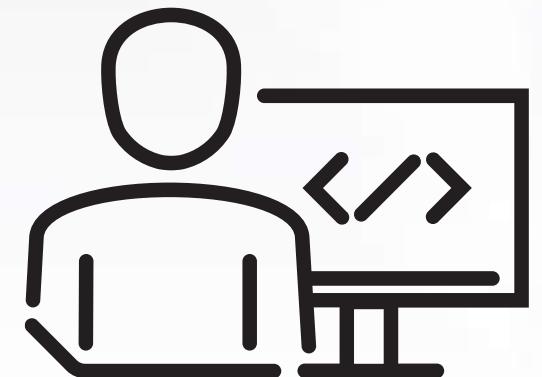
# Set a measurement system

- How to change the Measurement System at Runtime
  - Method
    - Command > Variable commands > Set variable value
  - Variable To Modify
    - Aliases > {Session} > Session > ActualMeasurementSystem
  - Value
    - Select one of the available Measurement Systems



# Hands-on session

- Set one or more Units of Measure to one or more variables
- Add a button to change the Measurement System (or the Locale)
- Check the conversion is applied





Develop using version  
control and collaboration



# Version control and Collaboration

## • Version Control

- used for keeping track of incrementally-different versions
- work also locally (offline)

## • Collaboration

- distributed version control
- form of version control in which the full history, is mirrored on every developer's computer
- enables automatic management of merges,
- does not rely on a single location for backups
- Git is a distributed version control system
- GitHub is an Internet hosting service using Git



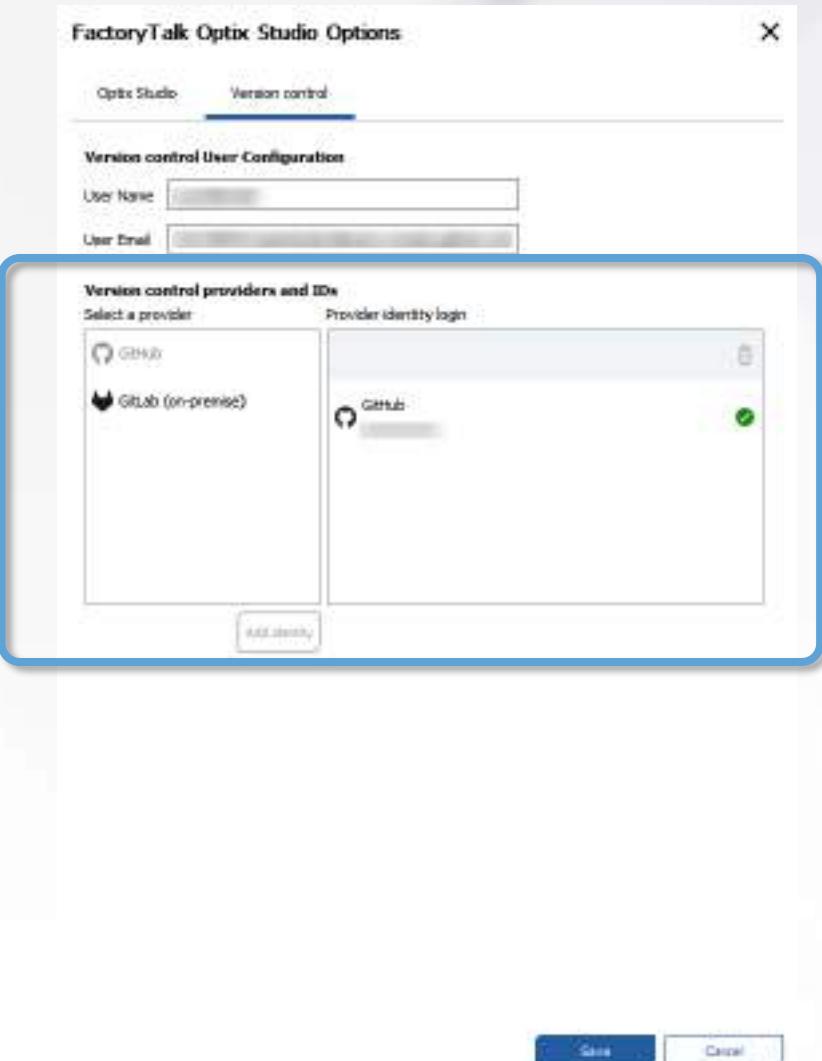
# Version control and Collaboration

## • Version Control

- used for keeping track of incrementally-different versions
- work also locally (offline)

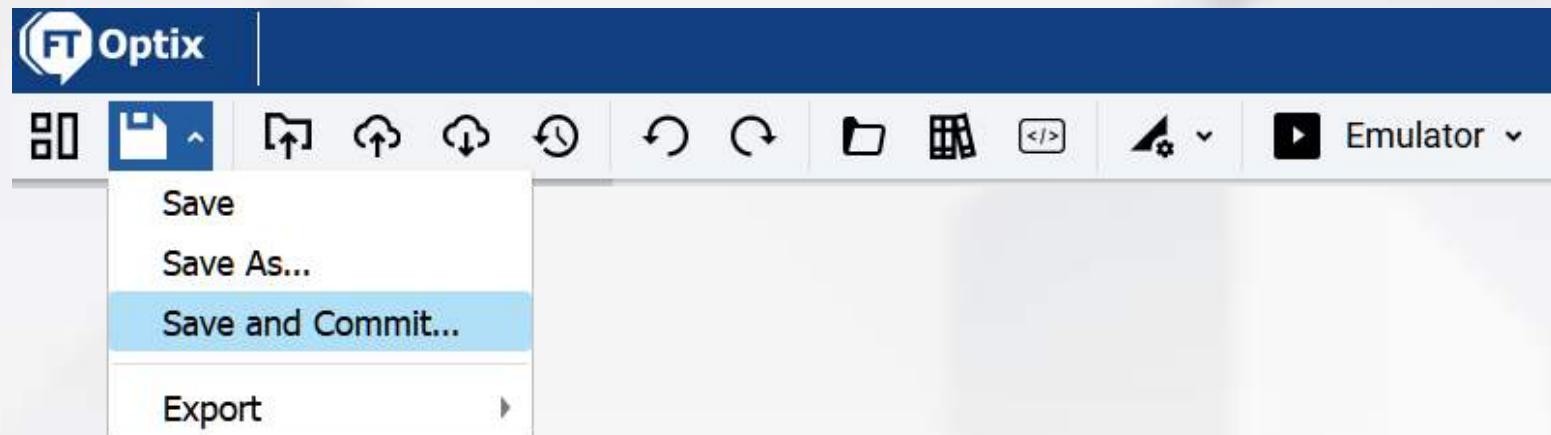
## • Collaboration

- distributed version control
- form of version control in which the full history, is mirrored on every developer's computer
- enables automatic management of merges,
- does not rely on a single location for backups
- Git is a distributed version control system
- GitHub is an Internet hosting service using Git



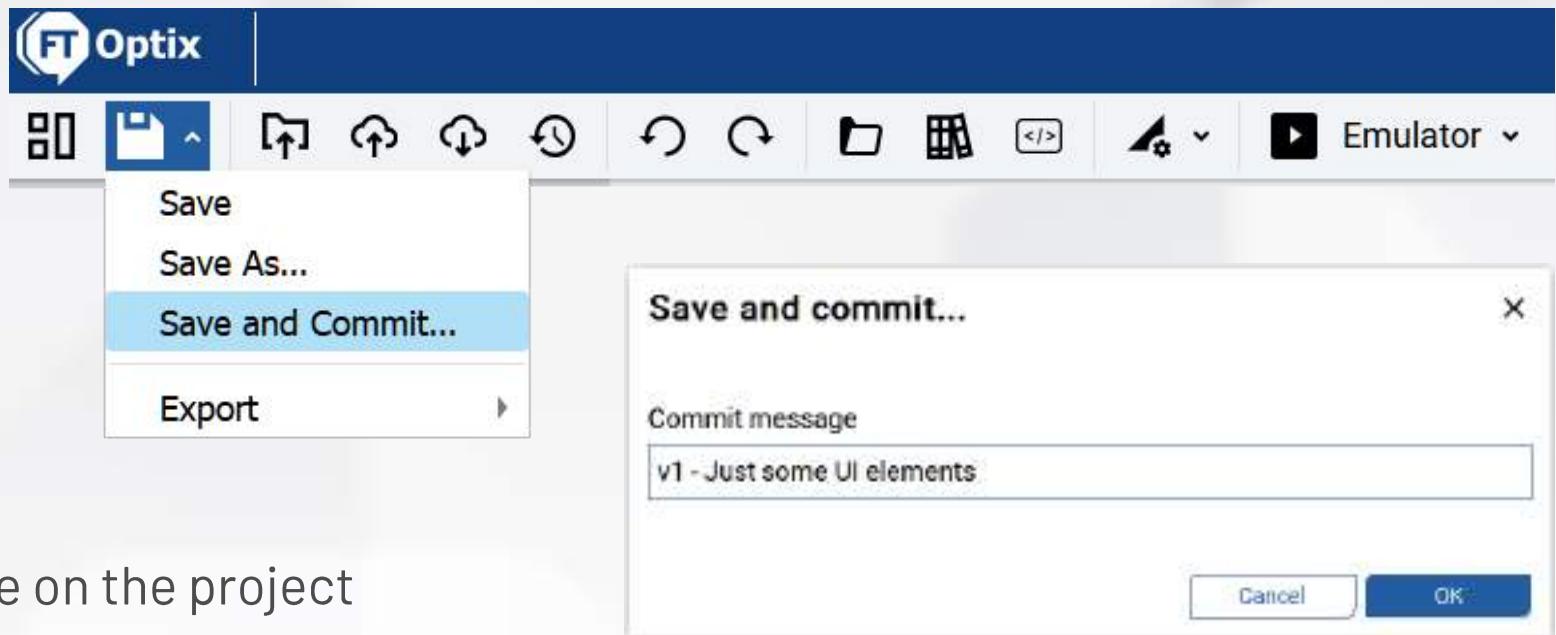
# Version control: save and commit locally

- Save and Commit,  
allow saving the project  
keeping track of changes
- Works "Local"
- Ideal for those who work alone  
with no colleagues that collaborate on the project
- No more needs to make zip files named v1, v2...
- The history button show differences,  
and allow restoring a previous version



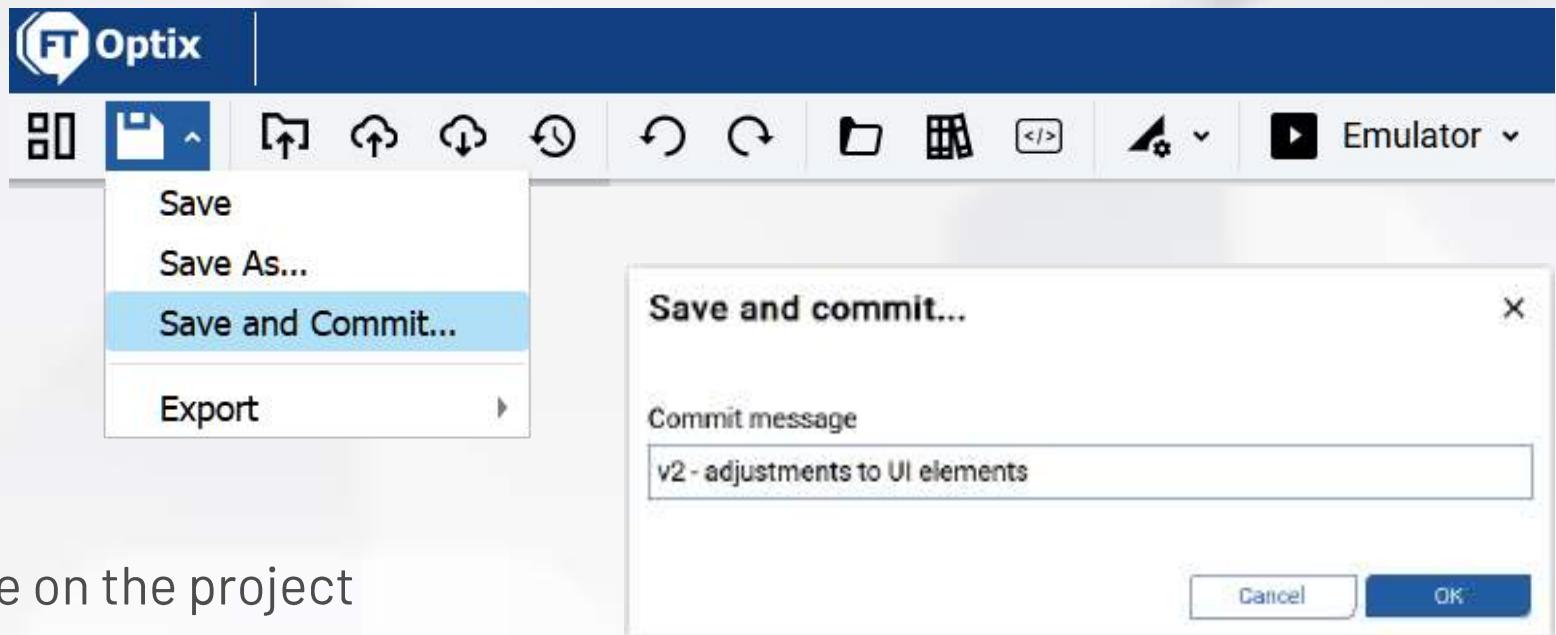
# Version control: save and commit locally

- Save and Commit,  
allow saving the project  
keeping track of changes
- Works "Local"
- Ideal for those who work alone  
with no colleagues that collaborate on the project
- No more needs to make zip files named v1, v2...
- The history button show differences,  
and allow restoring a previous version



# Version control: save and commit locally

- Save and Commit,  
allow saving the project  
keeping track of changes
- Works "Local"
- Ideal for those who work alone  
with no colleagues that collaborate on the project
- No more needs to make zip files named v1, v2...
- The history button show differences,  
and allow restoring a previous version



# Version control: save and commit locally

- Save and Commit,  
allow saving the project  
keeping track of changes
- Works "Local"
- Ideal for those who work alone  
with no colleagues that collaborate on the project
- No more needs to make zip files named v1, v2...
- The history button show differences,  
and allow restoring a previous version

The screenshot displays a software interface for managing project versions and configurations. On the left, a 'View history' panel shows two commits: 'v2 - adjustments to UI elements' (checked) and 'v1 - Just some UI elements' (checked). Both were made on 'Thu Sep 22 09:33:47 2022 +0200 by LucBeg'. On the right, a tree view shows the project structure: 'MyProject' contains 'UI', which further contains 'MainWindow (type)'. Under 'MainWindow (type)', there are three spin boxes: 'SpinBox1' (var LeftMargin: 130 ▶ 260), 'SpinBox2' (var TopMargin: 78 ▶ 80), and 'SpinBox3' (var Value: DynamicLink {NodId:ns=81;g=ee385914...}). A 'Restore' button is located at the bottom right of the configuration pane.

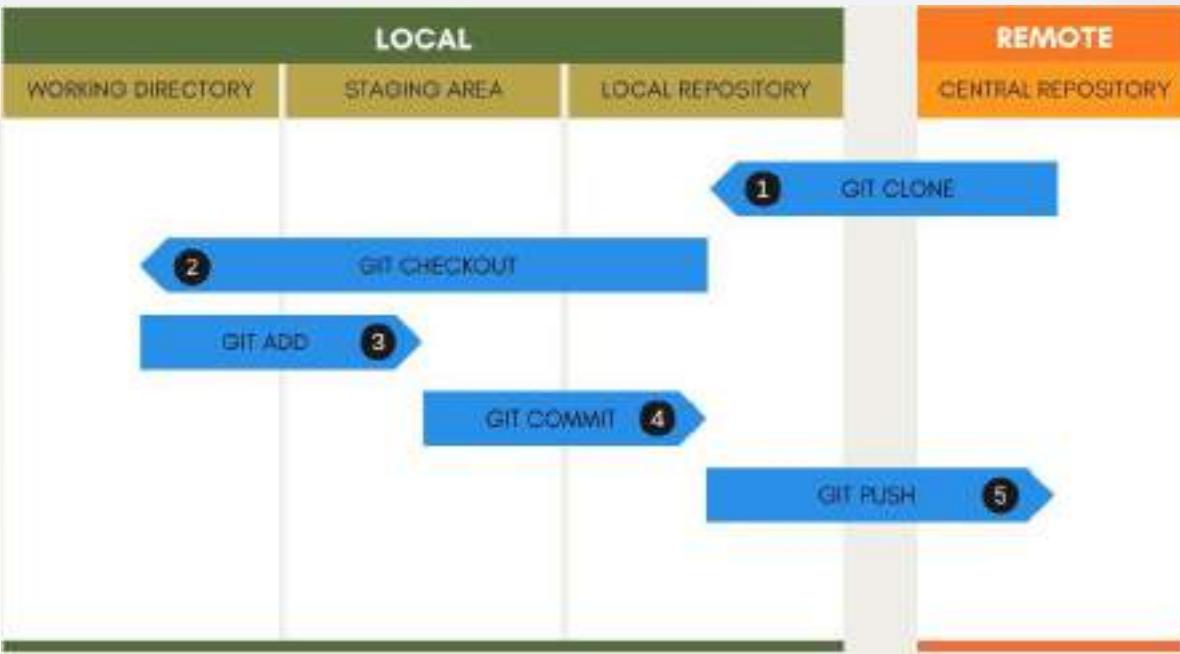
# Version control: save and commit locally

- Save and Commit,  
allow saving the project  
keeping track of changes
- Works "Local"
- Ideal for those who work alone  
with no colleagues that collaborate on the project
- No more needs to make zip files named v1, v2...
- The history button show differences,  
and allow restoring a previous version

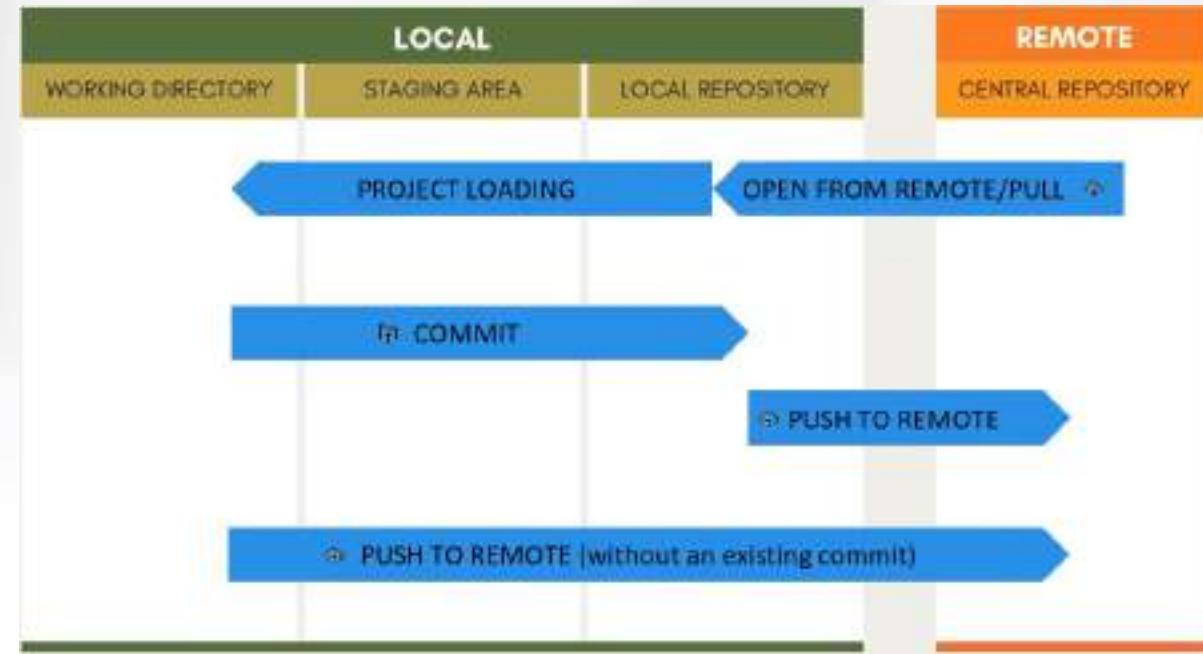
The screenshot shows a software interface with two main panels. On the left, a 'View history' panel displays a timeline of commits. The top commit is 'v2 - adjustments to UI elements' (checkbox checked) and the bottom commit is 'v1 - Just some UI elements' (checkbox checked). Both commits were made on 'Thu Sep 22 09:33:47 2022 +0200 by LucBeg'. On the right, a 'MyProject' tree view shows various components: BranchingEnabled (0), PasswordPolicy, Locales (en-US), LocaleFallbackList (en-US), UI, Model, Converters, Alarms, Recipes, Loggers, DataStores, Reports, OPC-UA, and CommDrivers. A warning message at the bottom states: '⚠️ Commits that occurred after a restored version will be removed permanently.' with a 'Restore' button.

# GIT workflow vs. Optix workflow

GIT Workflow



Optix Workflow



- Optix «hides» the GIT commands to improve the user experience
- User does not need to know anything about GIT commands and workflow

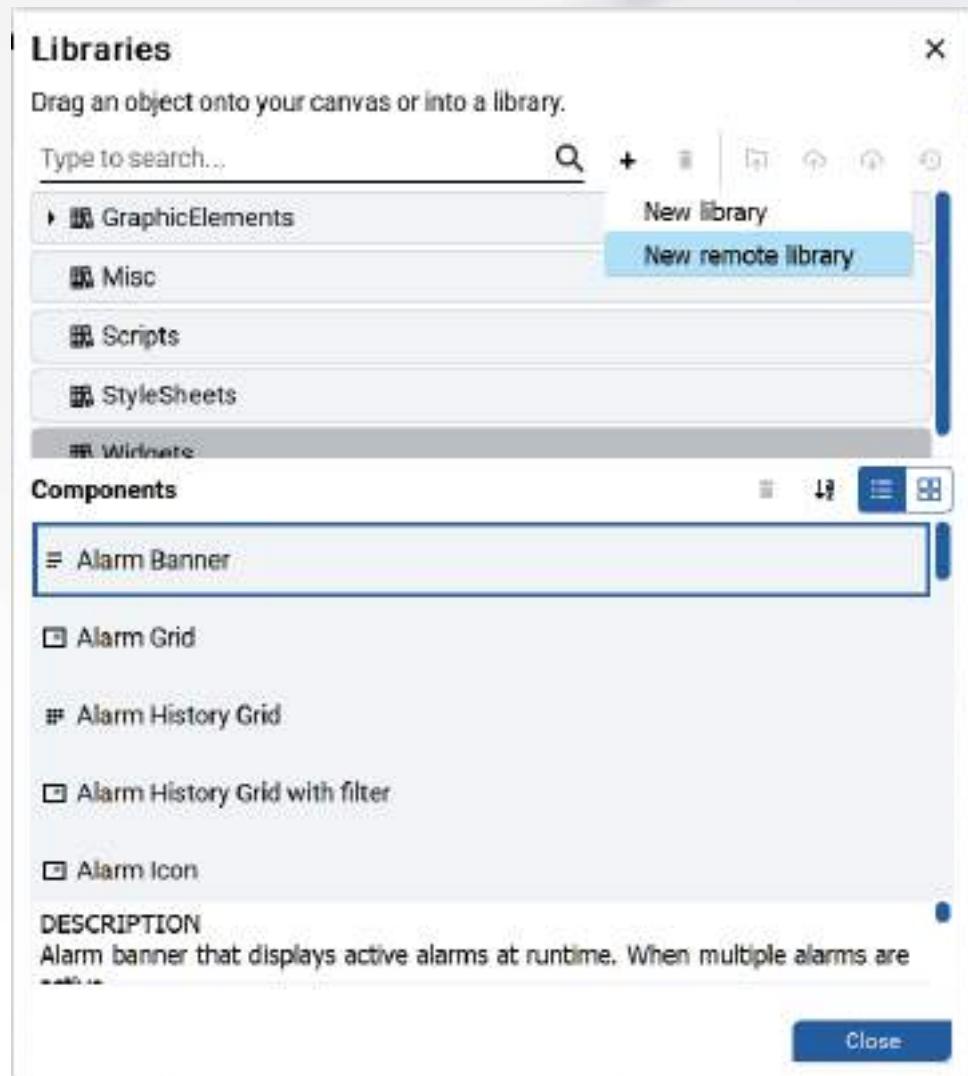
# Collaboration tips

- Always configure/verify the GitHub access from Studio options
- General Git-related tips
  - Repository name **MUST** have the same name of Optix project
  - Some git providers does not allow non-alphanumerical characters
  - You can set the repository to be Private, then share with colleagues



# Sharing libraries using Version Control

- a Custom Library can be shared exactly the same as the projects
- a Shared Library is useful when the company has built his own graphical or widget library in the way all projects reuse the same basic components
- Libraries does not offer back compatibility, once somebody from the collaborators upgrades a library, everybody must upgrade the IDE to use it (same as projects)



# Hands-on session

- Execute some Save and commit
- Explore the version history
- Try to restore to a previous state



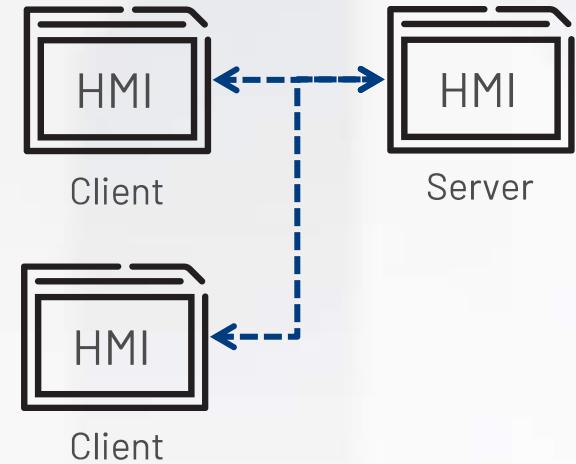
A large, abstract graphic consisting of a grid of blue dots arranged in a wave-like, undulating pattern across the entire slide.

# Define client-server architectures via OPC-UA



# Define client-server architectures via OPC-UA

- Thanks to the Full-featured OPC-UA architecture of FactoryTalk Optix  
It's possible to define client-server architectures
- Clients will have access to:
  - Variables
  - Methods
  - Alarms and Historical Alarms
  - Datalogger and Eventlogger
  - Recipes
- Client: HMI project with OPC-UA Client (Full OPC-UA Client = 3 tokens)
- Server: HMI project with OPC-UA Server (1 client = 1 token, 2 or more clients = 3 tokens)



# Configure the server

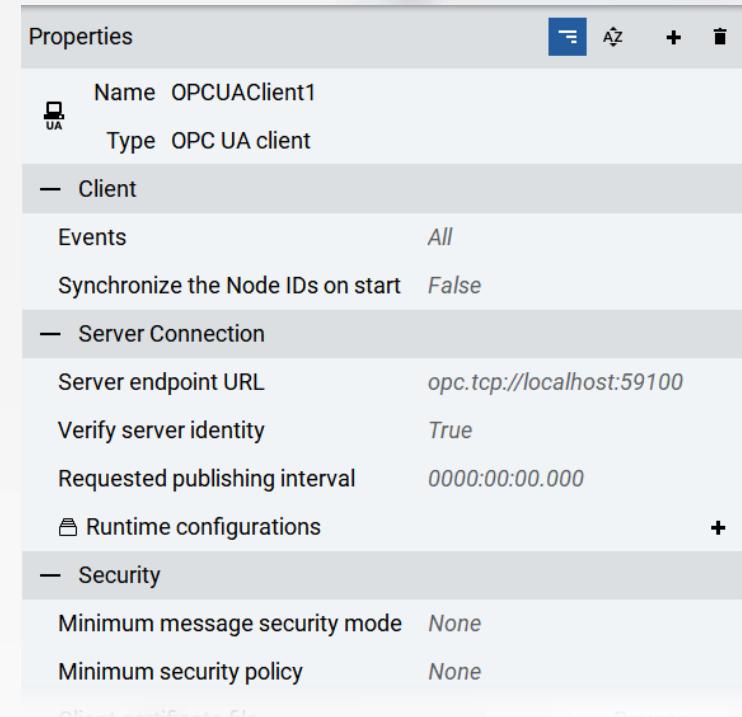
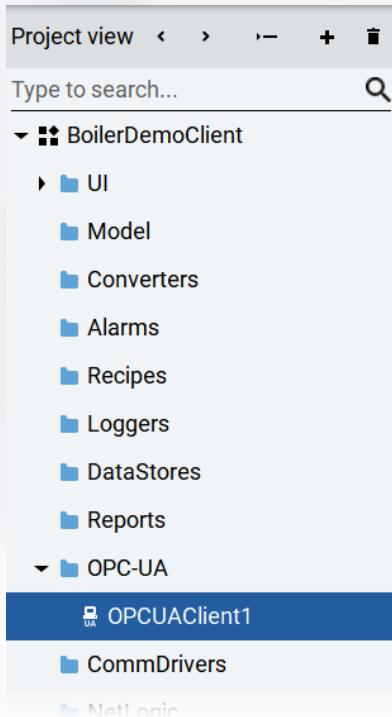
- On the HMI project, that is going to act as a Server, add the OPCUAServer node
- Endpoint can be manually changed
- Also Nodes to publish can be manually defined

The image shows two side-by-side windows. The left window is titled 'Project view' and displays a tree structure of a project named 'BoilerDemo'. The 'OPC-UA' folder is expanded, showing the 'OPCUAServer' node, which is highlighted with a blue selection bar. Other nodes visible include 'UI', 'Model', 'Converters', 'Alarms', 'Recipes', 'Loggers', 'DataStores', 'Reports', 'CommDrivers', and 'NetLogic'. A search bar at the top of the project view window contains the text 'Type to search...'. The right window is titled 'Properties' and shows configuration settings for the selected 'OPCUAServer' node. The 'Name' is set to 'OPCUAServer' and the 'Type' is 'OPC UA server'. Under the 'Server' section, the 'Endpoint URL' is set to 'opc.tcp://localhost:59100' with a 'Browse' link. The 'Nodes to publish' section lists several items, including 'Sampling interval' (set to '0000:00:00.100') and 'Multiple connection' (set to 'False'). The 'Security' section includes fields for 'Minimum message sec...' (set to 'None') and 'Minimum security policy' (set to 'None'). The 'Information' section shows the 'Product URI' as 'FactoryTalkOptivHMI-FTOptivApplication'. The entire properties window has a light gray header bar with icons for sorting, filtering, and adding.

Properties	
Name	OPCUAServer
Type	OPC UA server
— Server	
Endpoint URL	opc.tcp://localhost:59100 <a href="#">Browse</a>
Nodes to publish	
Sampling interval	0000:00:00.100
Multiple connection	False
— Security	
Minimum message sec...	<a href="#">None</a>
Minimum security policy	<a href="#">None</a>
Server certificate file	<a href="#">Browse</a>
Server private key file	<a href="#">Browse</a>
— Information	
Product URI	FactoryTalkOptivHMI-FTOptivApplication

# Configure the client

- On the HMI project, that is going to act as a Client, add the OPCUAClient node
- Server endpoint must be the one defined into the Server project
- Events property should be set to "All" to be able to attach to Methods, Alarms, Loggers...



# Configure the client

- Using the OPCUA Tag importer, it's possible to import not only variables but also objects!
- For example, select the entire folders Model, Alarms, Loggers and DataStores
- Elements like Alarms, Datalogger and AlarmsEventLogger can be used as they were part of the Client project

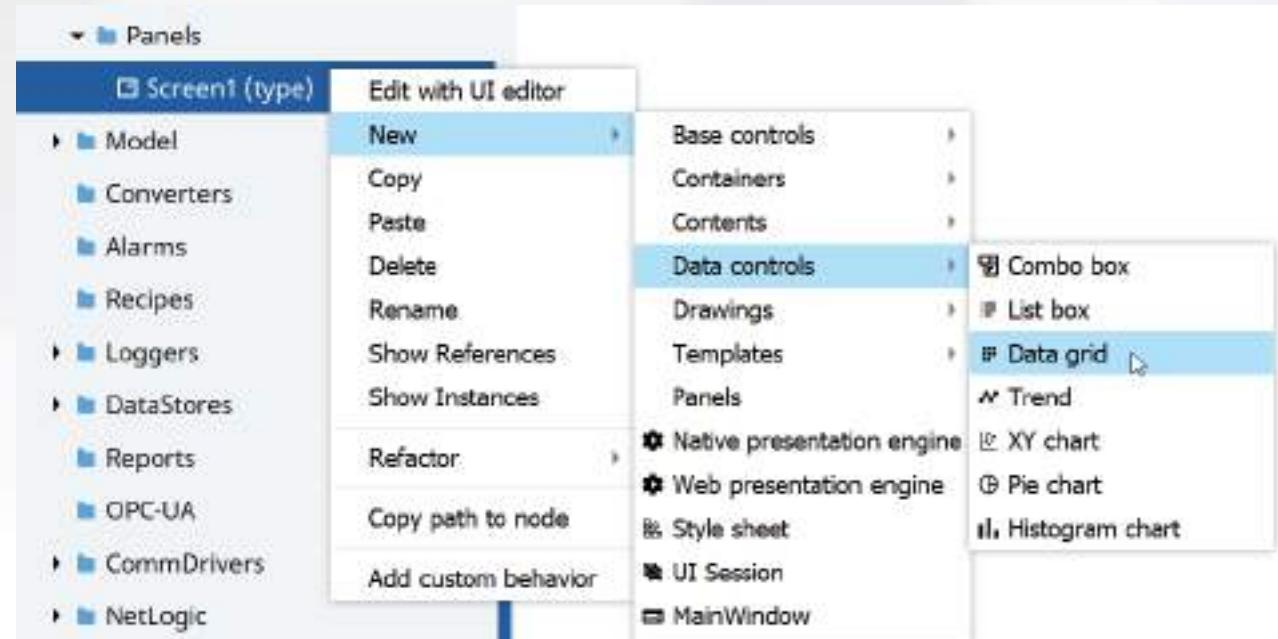
OPC-UA > OPCUAClient1 > **OPCUATagImporter**

Online Type to search...

<input checked="" type="checkbox"/>	BoilerDemo	Project folder
<input type="checkbox"/>	var BranchingEnabled	Variable Boolean
<input type="checkbox"/>	var Locales	Variable String[0]
<input type="checkbox"/>	var LocaleFallbackList	Variable String[0]
> <input type="checkbox"/>	UI	Folder
> <input checked="" type="checkbox"/>	Model	Folder
<input type="checkbox"/>	Converters	Folder
> <input checked="" type="checkbox"/>	Alarms	Folder
> <input type="checkbox"/>	Recipes	Folder
> <input checked="" type="checkbox"/>	Loggers	Folder
> <input checked="" type="checkbox"/>	DataStores	Folder
<input type="checkbox"/>	Reports	Folder
> <input type="checkbox"/>	OPC-UA	Folder
<input type="checkbox"/>	var MeasurementSystemMap	Variable MeasurementSyste...
> <input type="checkbox"/>	PasswordPolicy	Password policy

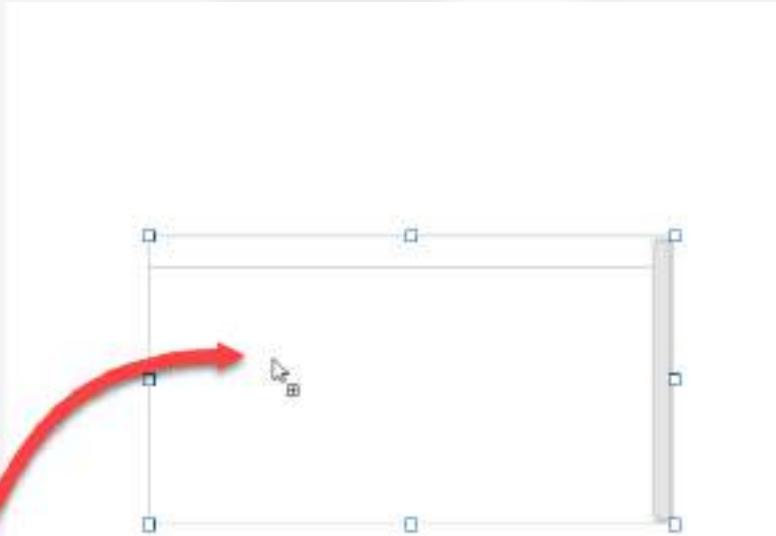
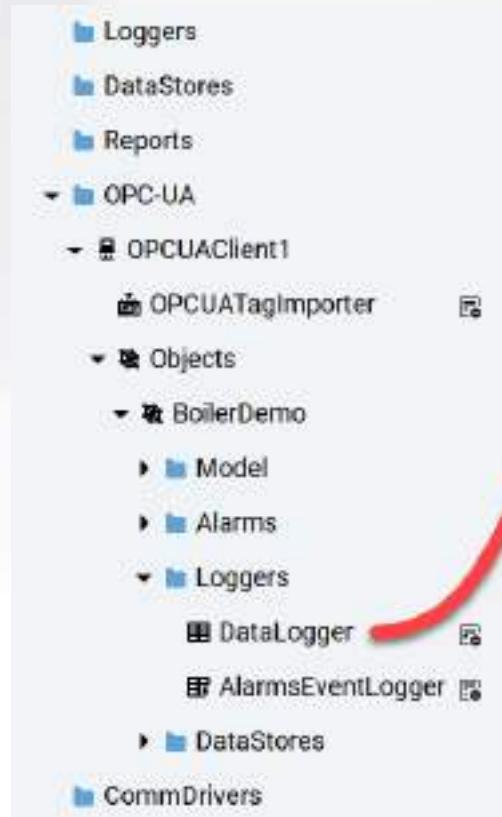
# Configure the client

- Using the OPCUA Tag importer, it's possible to import not only variables but also objects!
- For example, select the entire folders Model, Alarms, Loggers and DataStores
- Elements like Alarms, Datalogger and AlarmsEventLogger can be used as they were part of the Client project



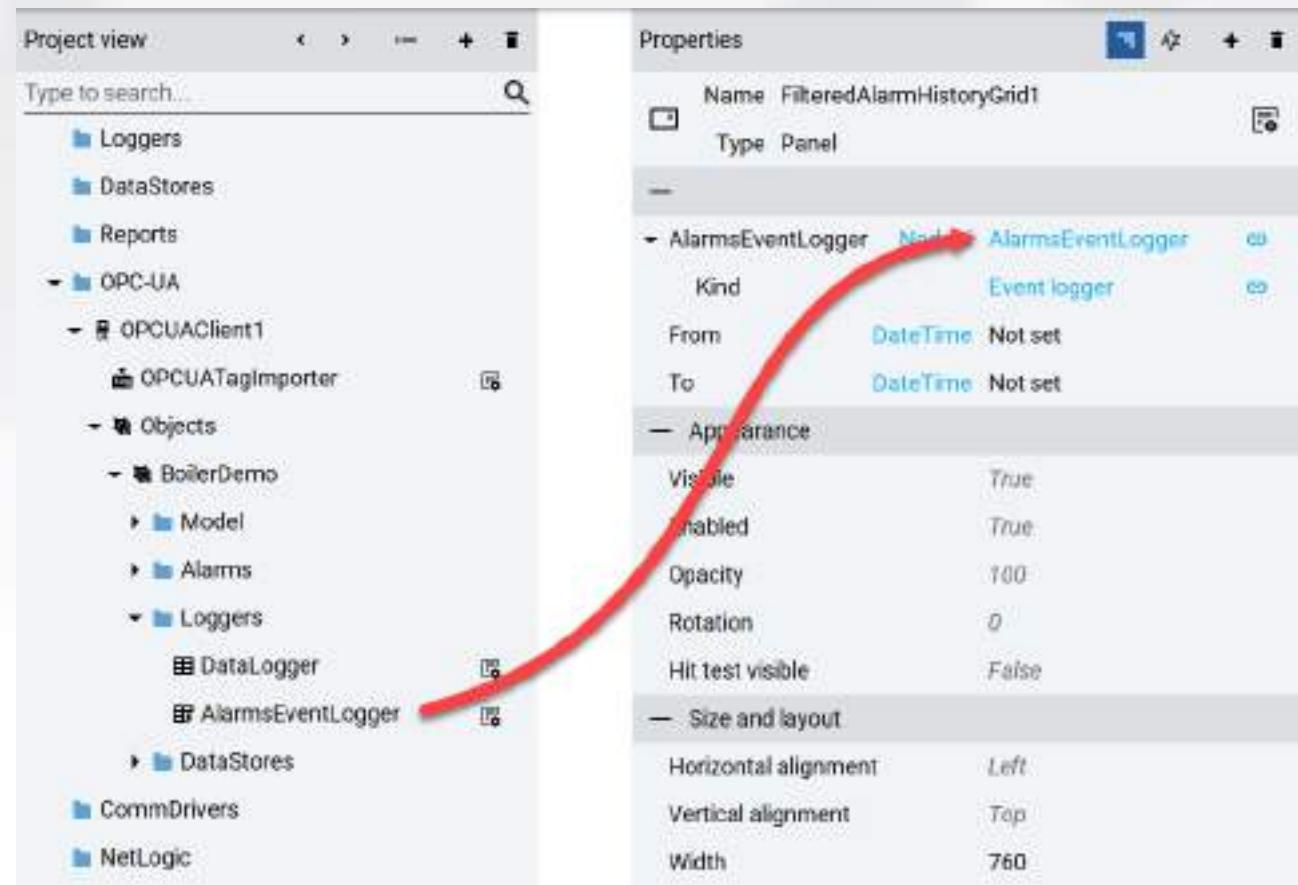
# Configure the client

- Using the OPCUA Tag importer, it's possible to import not only variables but also objects!
- For example, select the entire folders Model, Alarms, Loggers and DataStores
- Elements like Alarms, Datalogger and AlarmsEventLogger can be used as they were part of the Client project



# Configure the client

- Using the OPCUA Tag importer, it's possible to import not only variables but also objects!
- For example, select the entire folders Model, Alarms, Loggers and DataStores
- Elements like Alarms, Datalogger and AlarmsEventLogger can be used as they were part of the Client project



# Hands-on session

- Use the BoilerDemo project as Server
- Create a new project that acts as a Client
- Try to visualize and manage the Server's Alarms, Alarm History and Datalogger from the Client project



# Domain authentication



# Domain authentication

- Login can be forwarded to external services
  - Active Directory
  - LDAP with SSL (no LDAPS)
  - Oauth 2.0 (only with PKCE and JWKD endpoint)
- Need to configure the authentication server properties
  - Type
  - Address
  - Certificates
  - Tokens (for Oauth 2.0)



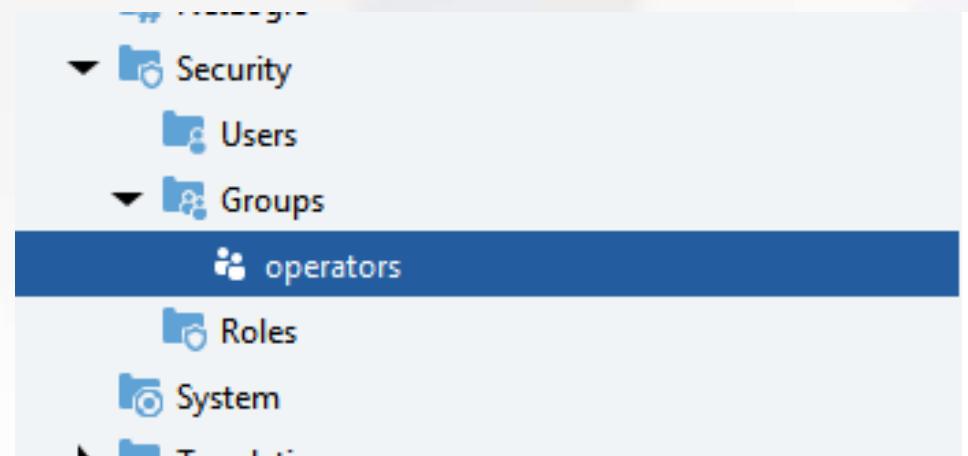
# Domain authentication

- Login can be forwarded to external services
  - Active Directory
  - LDAP with SSL (no LDAPS)
  - OAuth 2.0 (only with PKCE and JWKS endpoint)
- Need to configure the authentication server properties
  - Type
  - Address
  - Certificates
  - Tokens (for OAuth 2.0)



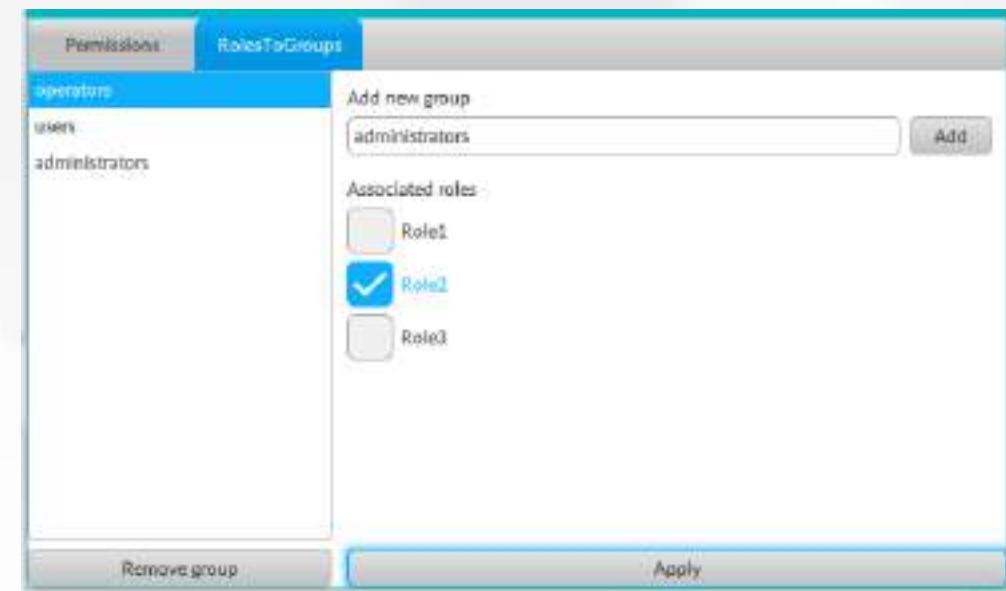
# Domain authentication

- No need to create local users
  - If the authentication is validated by the server, a local user is automatically added to the project
- Login is always validated by the authentication server
  - No password caching can be done
  - Cannot login if the authentication server is unreachable
- FactoryTalk Optix groups are automatically mapped to the authentication server groups
  - Designer must have the list of groups in advance
  - When logging to the authentication server, the list of groups where the users belongs is returned, a mapping to the Optix user with the same name is performed automatically
  - For example, if “user1” is part of the “operators” group on the authentication server, all controls linked to the local “operators” group will be accessible



# Domain authentication

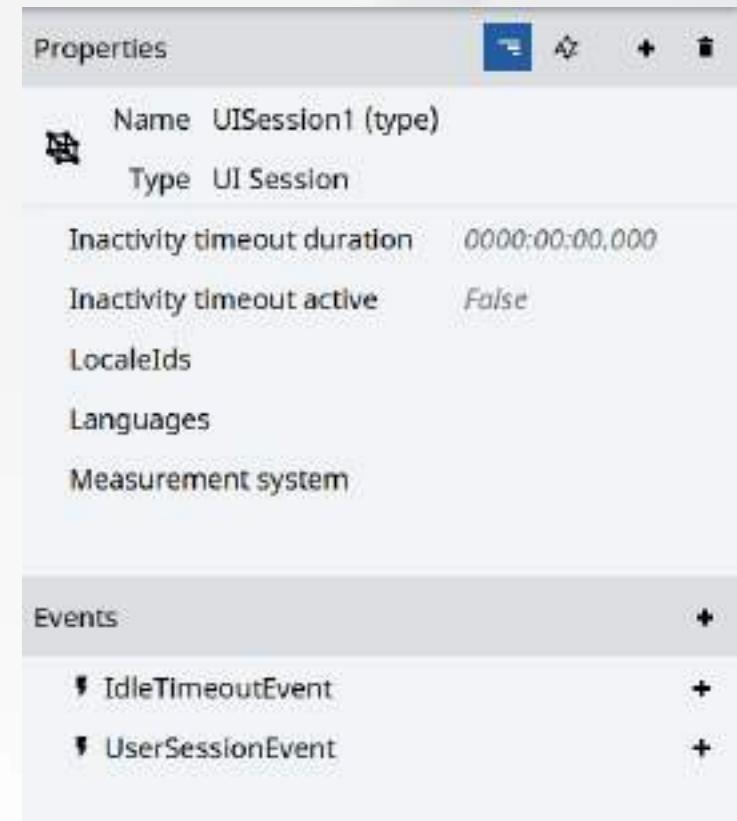
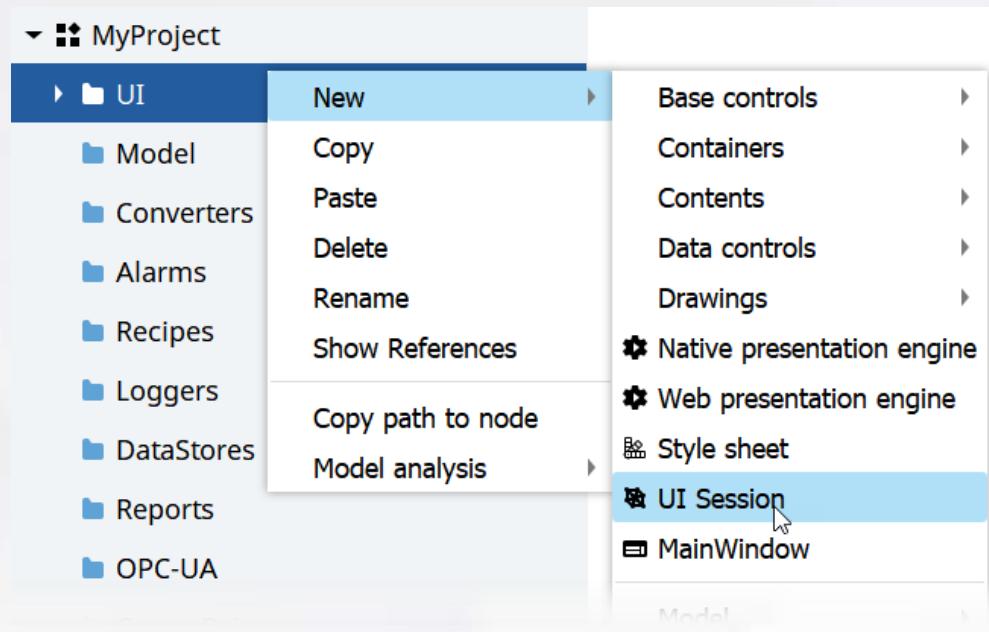
- If the designer does not have a list of the groups from the authentication server, roles can be used
  - All UI controls are tied to the Roles defined in the FactoryTalk Optix application
  - Groups can be created at runtime, then each group can be assigned to one or more roles during after deploying the project
- Full example is available in the GitHub organization
  - [Optix\\_Sample\\_AssignRolesToGroupsAtRuntime: How to associate Active Directory groups to FactoryTalk Optix roles at runtime](#)



# Define UI Session and Session variables



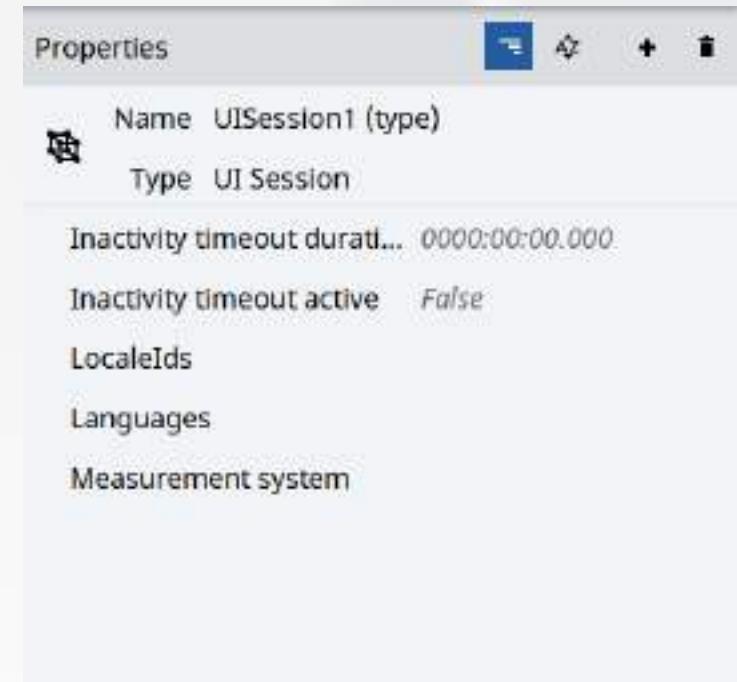
# UI Session



- UI Session allows defining a custom Session to be used on Native or Web presentation engine
- Allow customizing: Inactivity Timeout, Locale, Language and Measurement System

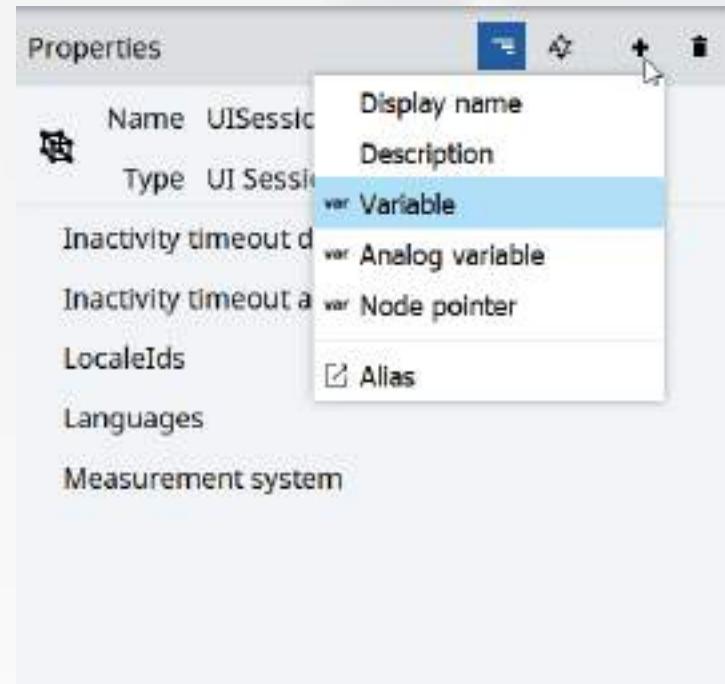
# Session variables vs others

- Model/Comm. Driver variables
  - belongs to the Project context
  - display the same value regardless of sessions or panels
- Panel variables
  - belongs to the Panel context
  - different sessions show different values,  
but when you change the panel the value is lost
- Session variables
  - belongs to the Session context
  - different sessions show different values,  
but when you change the panel the value is retained



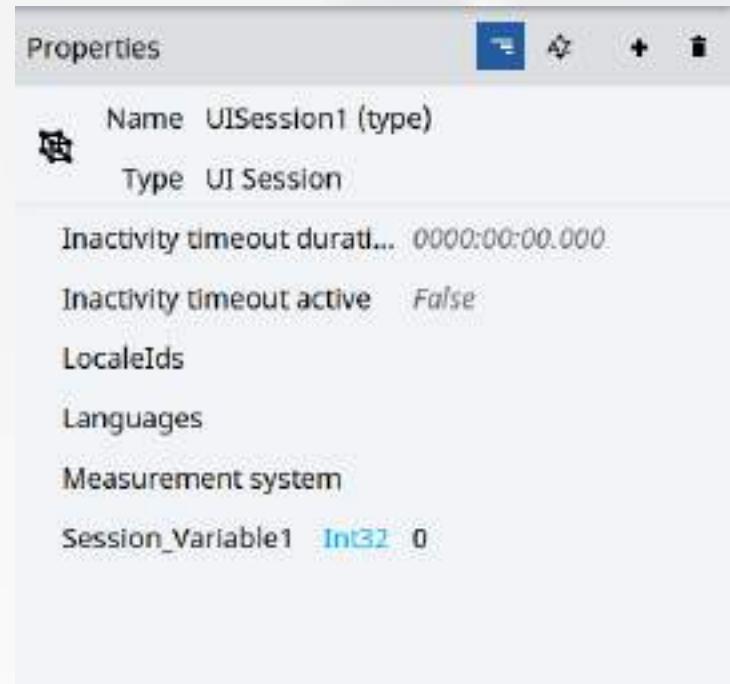
# Session variables vs others

- Model/Comm. Driver variables
  - belongs to the Project context
  - display the same value regardless of sessions or panels
- Panel variables
  - belongs to the Panel context
  - different sessions show different values,  
but when you change the panel the value is lost
- Session variables
  - belongs to the Session context
  - different sessions show different values,  
but when you change the panel the value is retained

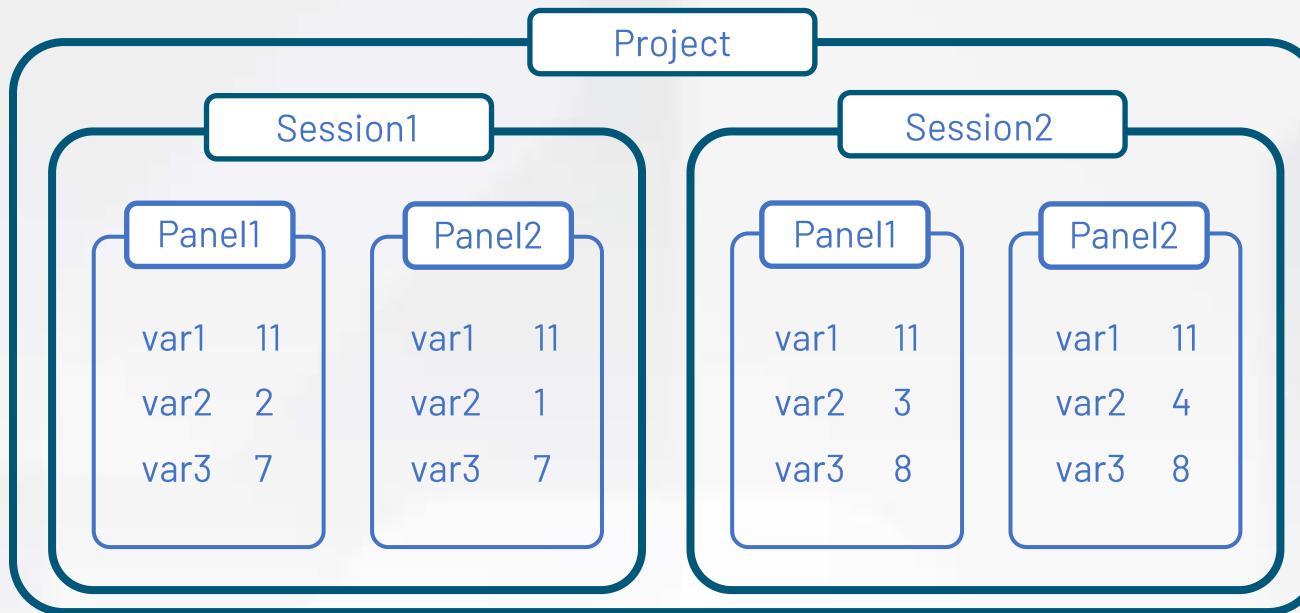


# Session variables vs others

- Model/Comm. Driver variables
  - belongs to the Project context
  - display the same value regardless of sessions or panels
- Panel variables
  - belongs to the Panel context
  - different sessions show different values,  
but when you change the panel the value is lost
- Session variables
  - belongs to the Session context
  - different sessions show different values,  
but when you change the panel the value is retained



# Session variables example



var1 = Model variable  
var2 = Panel variable  
var3 = Session variable

- Examples of different UI Sessions
  - Example1: one Session active on Native Presentation Engine, and another Session on the Web Presentation Engine
  - Example2: two browsers connected to the same Web Presentation Engine

# Work with NetLogic scripts



# .NET

## • What is .NET?

- .NET is a free, open-source development platform

## • What is .NET Standard?

- .NET Standard is an API specification that lets you develop class libraries for multiple implementations of .NET

## • What are .NET Framework, Mono and .NET Core?

- They are formally known as *implementations* of .NET Standard
- .NET Framework is the original implementation of .NET (runs only on Windows)
- Mono is used when a small runtime is required (formerly used by UNIQOO)
- .NET Core is the latest implementation and runs on any platform (used by OPTIX)

## • What is C#?

- it's one of the programming languages supported by .NET

## • What .NET version does Optix support?

- Depends on the FT Optix version, check requirements in the Install Guide
- Additional packages or libraries must be compatible with the specific .NET version (NuGet packages for example)

Version	Start Date	End Date
.NET 8 LTS	Nov 14, 2023	2026
.NET 7	Nov 8, 2022	May 14, 2024
.NET 6.0 LTS	Nov 8, 2021	Nov 12, 2024
.NET 5.0	Nov 10, 2020	May 10, 2022
.NET 3.1 LTS	Dec 3, 2019	Dec 13, 2022
.NET 3.0	Sep 23, 2019	Mar 3, 2020
.NET 2.2	Dec 4, 2018	Dec 23, 2019
.NET 2.1 LTS	May 30, 2018	Aug 21, 2021
.NET 2.0	Aug 14, 2017	Oct 1, 2018
.NET 1.1	Nov 16, 2016	Jun 27, 2019
.NET 1.0	Jun 27, 2016	Jun 27, 2019

# NetLogic

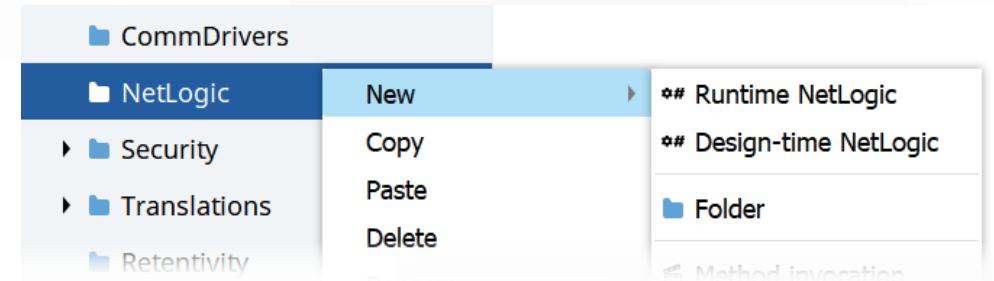
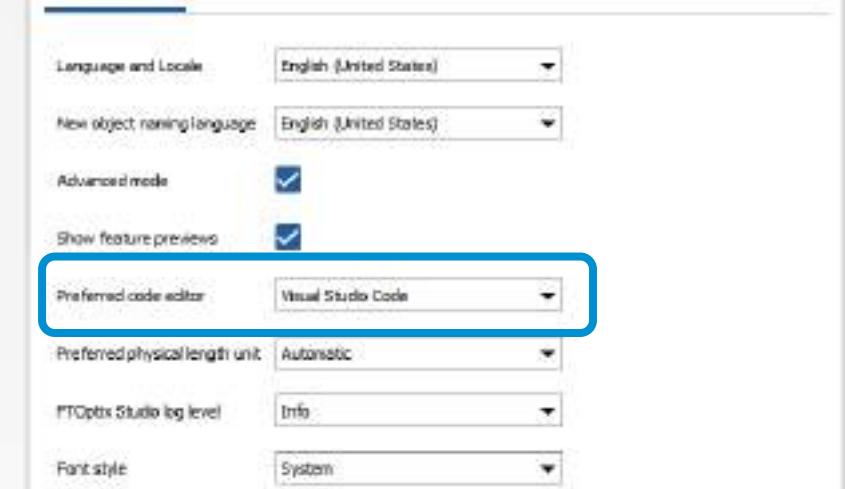
- It's an object that contains **C#** code compatible with the .NET version used by the current release of FactoryTalk Optix

- Supported NetLogic Editors:

- Microsoft Visual Studio Code ([Download](#))
  - C# Extension needs to be installed
- Microsoft Visual Studio 2022 (no previous versions)

- Types of NetLogic:

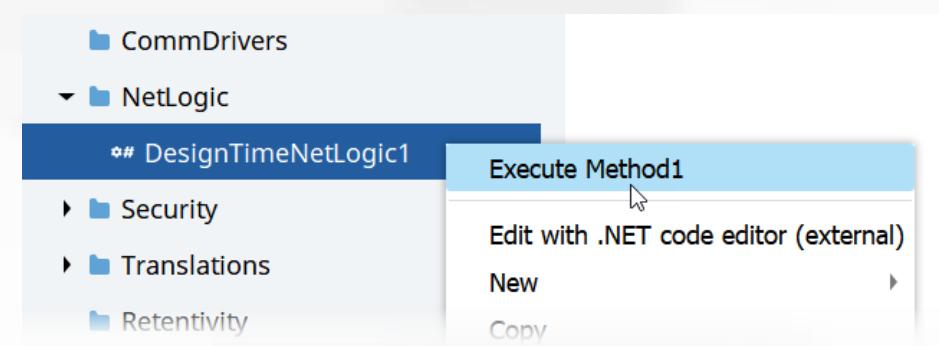
- **Runtime**: to be executed at Runtime
- **Design-time**: to be executed in IDE



# Design-time NetLogic

- **Where** can be placed?
  - Anywhere!  
But usually stay in NetLogic folder
- **When** is the method executed?
  - When the "exported method" is executed/called from Studio
- Notes on [ExportMethod]:
  - Allow exposing the method outside the script
  - Must be used before every method that need to be exposed
  - Exported methods, must be «public»
  - Methods must be void, output are passed as arguments using the «out» modifier

```
public class DesignTimeNetLogic1 : BaseNetLogic
{
    [ExportMethod]
    public void Method1()
    {
        // Insert code to be executed by the method
    }
}
```



# Design-time NetLogic: create simple variables

```
[ExportMethod]
public void CreateSimpleVariables()
{
    var myFolder = Project.Current.Get<Folder>("Model/Variables");
    if (myFolder!=null)
    {
        myFolder.Delete(); → Delete the folder(if any)
    }
    myFolder = InformationModel.Make<Folder>("Variables");
    for (int i=1; i<=3; i++)
    {
        var myVar = InformationModel.MakeVariable("Variable" + i, OpcUa.DataTypes.UInt16);
        myFolder.Add(myVar);
    }
    Project.Current.Get("Model").Add(myFolder);
}
```

TIP: Optix uses some custom APIs to create and manage project elements of any kind.

Do not use the `var myVar = new IUAVariable();` syntax from plain C# as it will not work, it will only create an useless blank class

TIP: When creating multiple elements under the same parent folder, a best practice is to create the container, add the element and finally add the container to the project. This will avoid executing multiple calls to the FactoryTalk Optix information model

# Design-time NetLogic: create simple variables APIs overview

```
[ExportMethod]
public void CreateSimpleVariables()
{
    var myFolder = Project.Current.Get<Folder>("Model/Variables");
    if (myFolder!=null)
    {
        myFolder.Delete();
    }
    myFolder = InformationModel.Make<Folder>("Variables");
    for (int i=1; i<=3; i++)
    {
        var myVar = InformationModel.MakeVariable("Variable" + i, OpcUa.DataTypes.UInt16);
        myFolder.Add(myVar);
    }
    Project.Current.Get("Model").Add(myFolder);
}
```

Map this C# method to an equivalent OPCUA method

Elements are in C# code only

Retrieve the element of type "Folder" at the path "Model/Variables"

Delete the folder (if any)

From the root of the current project

Create an object of type "Folder" called "Variables"

Create a new variable called "VariableX" of type UINT16

The Optix class to create project elements

Add the variable to the folder

Add the folder containing all the variables from the C# code to the project

# Design-time NetLogic: create Motor Type and instances

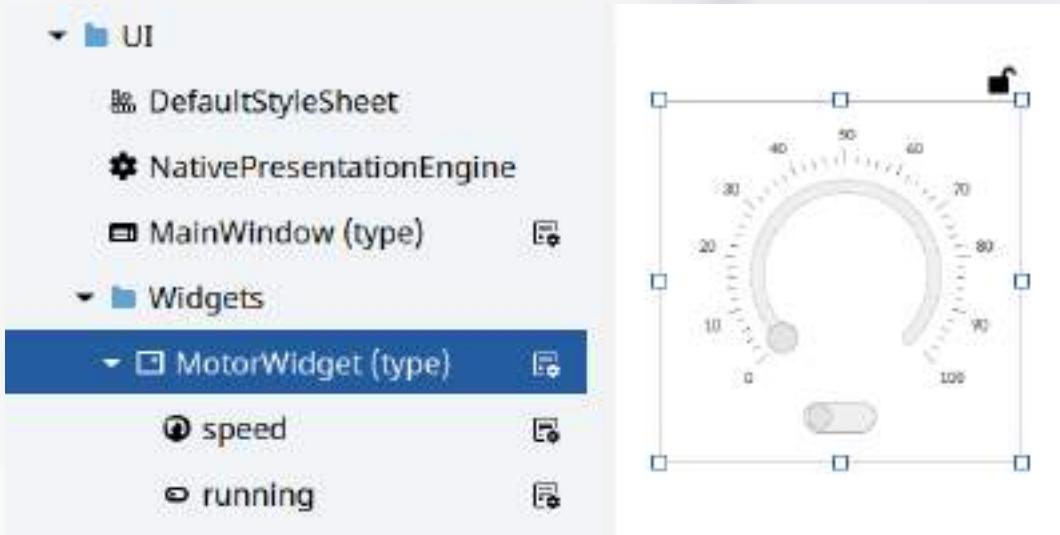
```
[ExportMethod]
public void CreateMotorType()
{
    // Create an UAObject and call it MotorType
    var motorType = InformationModel.MakeObjectType("MotorType");
    // Create some properties for the motor object
    var speed = InformationModel.MakeVariable("Speed", OpcUa.DataTypes.UInt16);
    var running = InformationModel.MakeVariable("Running", OpcUa.DataTypes.Boolean);
    // Attach the properties of the motor to the motor object
    motorType.Add(speed);
    motorType.Add(running);
    // Attach the motor type (with the child properties) to the project
    Project.Current.Get("Model").Add(motorType);
}

[ExportMethod]
public void CreateMotorInstances()
{
    // Create a new folder
    var motorInstancesFolder = InformationModel.Make<Folder>("MotorInstances");
    // Create some motors and add them to the new folder
    for (int i = 1; i <= 3; i++)
    {
        var motorType = Project.Current.Get("Model/MotorType");
        var motorInstance = InformationModel.MakeObject("Motor" + i, motorType.NodeId);
        motorInstancesFolder.Add(motorInstance);
    }
    // Attach the folder (with the motors) to the current project
    Project.Current.Get("Model").Add(motorInstancesFolder);
}
```

# Design-time NetLogic: create widget instances

```
[ExportMethod]
public void CreateMotorWidgetInstances()
{
    // NOTE: the "MotorWidget" type with "MotorAlias" alias,
    // must be already available at UI/Widgets

    for (int i = 1; i <= 3; i++)
    {
        var motorWidgetType = Project.Current.Get("UI/Widgets/MotorWidget");
        var motorWidgetInstance = InformationModel.MakeObject("MotorWidget" + i, motorWidgetType.NodeId);
        motorWidgetInstance.SetAlias("MotorAlias", Project.Current.Get("Model/MotorInstances/Motor" + i));
        ((Panel)motorWidgetInstance).LeftMargin = ((Panel)motorWidgetInstance).Width + 50) * (i - 1);
        Project.Current.Get("UI/MainWindow").Add(motorWidgetInstance);
    }
}
```

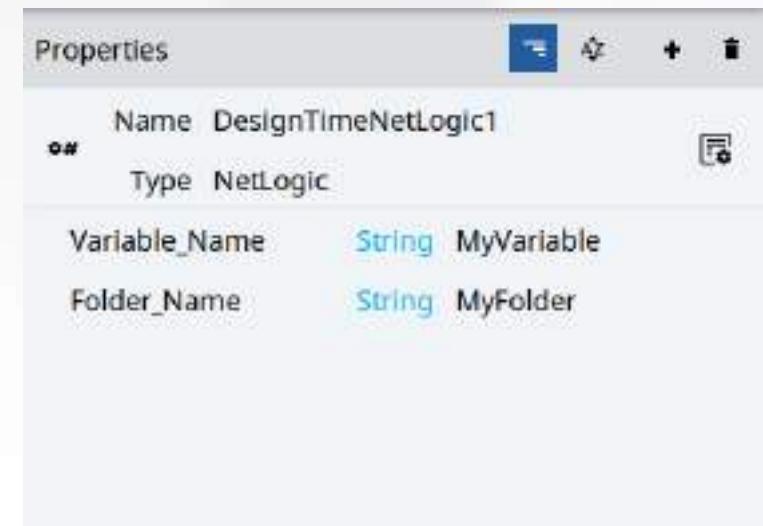


Cast to Panel is needed as the MakeObject would return a generic OPC-UA Object which does not contain the margin properties

# Design-time NetLogic: script parameters

- Variables added to the NetLogic object behaves like script parameters
- Value of these variables can be read using LogicObject.GetVariable("{variable}").Value

```
[ExportMethod]
public void ReadParameters()
{
    string varName = LogicObject.GetVariable("Variable_Name").Value;
    string folderName = LogicObject.GetVariable("Folder_Name").Value;
    Log.Info(LogicObject.BrowseName, "Variable_Name = " + varName);
    Log.Info(LogicObject.BrowseName, "Folder_Name = " + folderName);
}
```

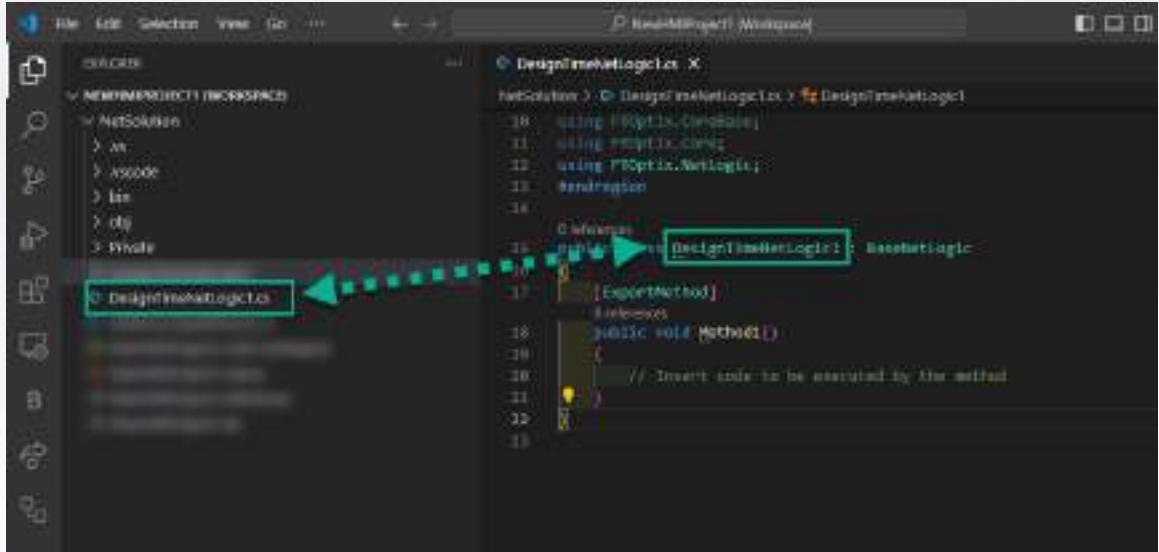


# Access node path

- Using `Log.Node(IUANode)`; a string is returned containing the project path to such node

```
var myNode = Project.Current.Get("Model/Variable1");
var nodePath = Log.Node(myNode); // returns /Objects/Model/Variable1
```

# NetLogic don'ts !



1. NetLogic class name must be the same of the NetLogic object name
    - **Don't change the name from the source code!**
    - Rename it from Studio instead!
  2. NetLogic objects are also browsable from the NetSolution
    - **Don't delete a NetLogic from the solution!**
    - Remove it from Studio instead!

# Cast vs Type parameter

- Use of a "**type parameter**"

- Type parameter can be used with "generic methods" that allows using different types
- A generic method can be called either without an argument or by specifying the type of argument within angle brackets
- Returns `null` if `MyFolder` is not an instance of `Folder`

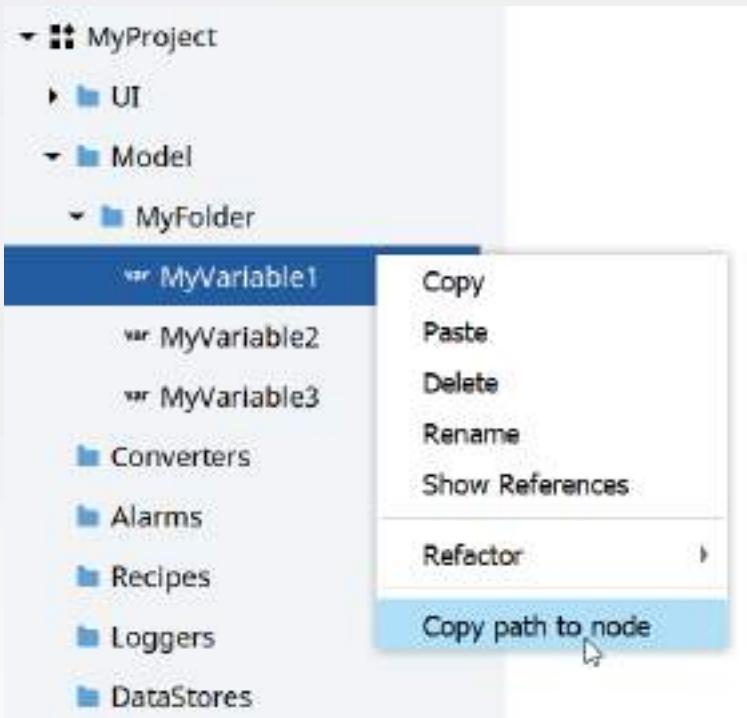
```
var myfolder = Project.Current.Get<Folder>("Model\MyFolder");
```

- Use of a "**cast**"

- a cast is an explicit conversion
- To perform a cast, specify the type that you are casting to in parentheses in front of the value or variable to be converted.
- Throws `InvalidCastException` if `MyFolder` cannot be assigned to `Folder` type

```
var myfolder = (Folder)Project.Current.Get("Model\MyFolder");
```

# Copy path to node



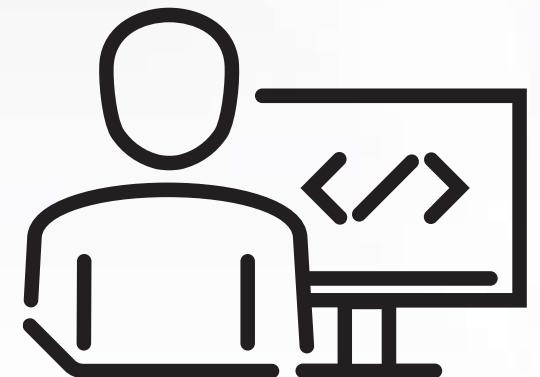
- Can be useful when using absolute addressing with `Project.Current.Get`
- Provides the absolute path of the node to clipboard  
`"MyProject/Model/MyFolder/MyVariable1"`
- `Project.Current.GetVariable("Model/MyFolder/MyVariable1");`
- NOTE: remember to remove the project name

# NetLogic(s) FAQ

- Can I add ActiveX controls to my FactoryTalk Optix application?
  - No, you can't add ActiveX to FactoryTalk Optix, this is due to safety concerns (deprecated technology)
- I have my custom WPF application, can I embed it into FactoryTalk Optix?
  - No, graphics can only come from FactoryTalk Optix UI, there's no chance to embed other applications
  - If the custom application exposes a HTML interface, the WebBrowser object can be used
- Can I use any NuGet package in my FactoryTalk Optix application?
  - Yes, if they are compatible with the .NET version used in FactoryTalk Optix
  - Yes, if they are cross-platform capable (compiler target must be AnyCPU)
  - Exceptions are:
    - Windows Forms (WPF and derivates are not cross-platform and cannot be used)
    - Microsoft.SQL (customers should use ODBC connector instead) → Limitation was removed in Optix 1.6.X
- Any limitations in a DesignTime script?
  - There is no way to access value of PLC tags (CommDriver only exist at Runtime)
  - There is no way to interact with ODBC connectors, EmbeddedDatabase or InfluxDB using Optix modules

# Hands-on session

- Write one or more Design-Time script
- See what happens if multiple variables are created with the same name



# Runtime Netlogic

- **Where** can it be placed?
  - into the NetLogic folder
  - into the Main Window
  - into a Panel
  - into a graphical object (like a Button)

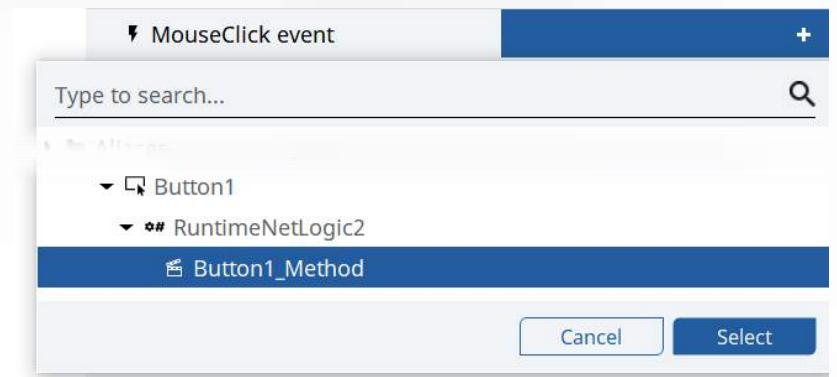
- **When** are the methods executed?

- *Start*: automatically executed when the node that contain the NetLogic is created/opened
- *Stop*: automatically executed when the node that contain the NetLogic is destroyed/closed
- *Exported Method*: executed when called by an Event

```
public class RuntimeNetLogic1 : BaseNetLogic
{
    public override void Start()
    {
        // code executed when the logic is started
    }

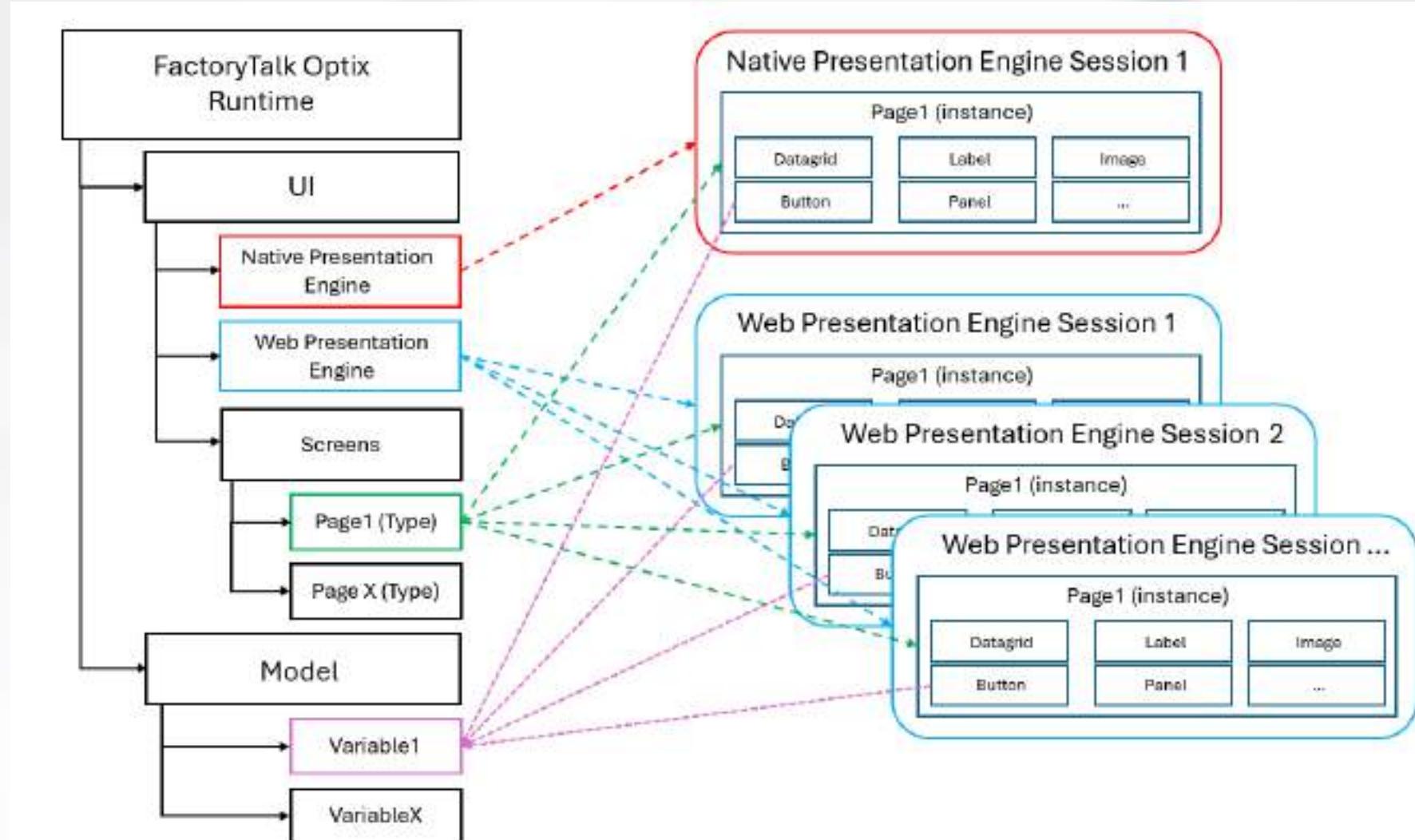
    public override void Stop()
    {
        // code executed when the logic is stopped
    }

    [ExportMethod]
    public void Button1_Method()
    {
        // code executed by the method
    }
}
```



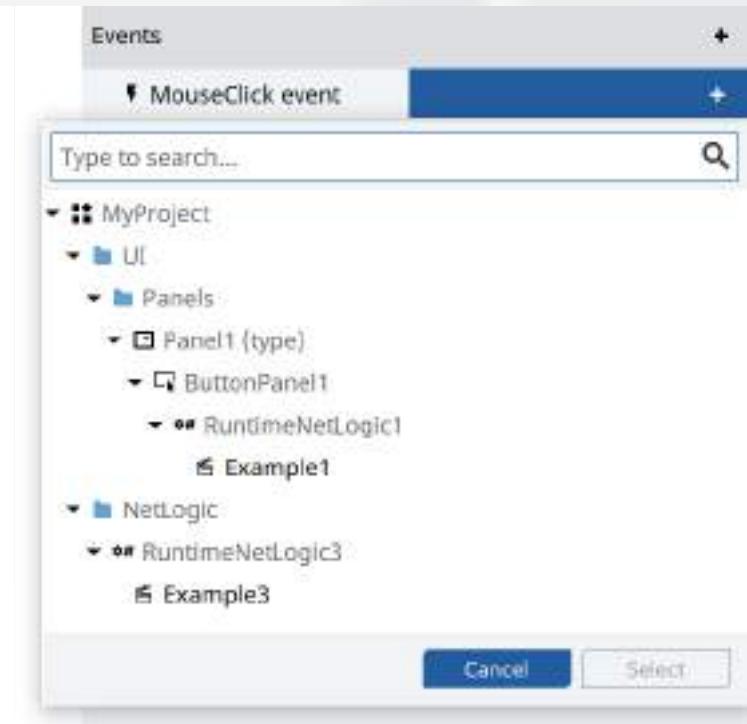
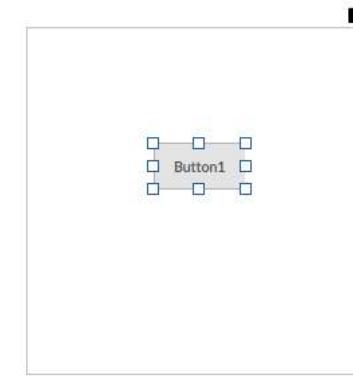
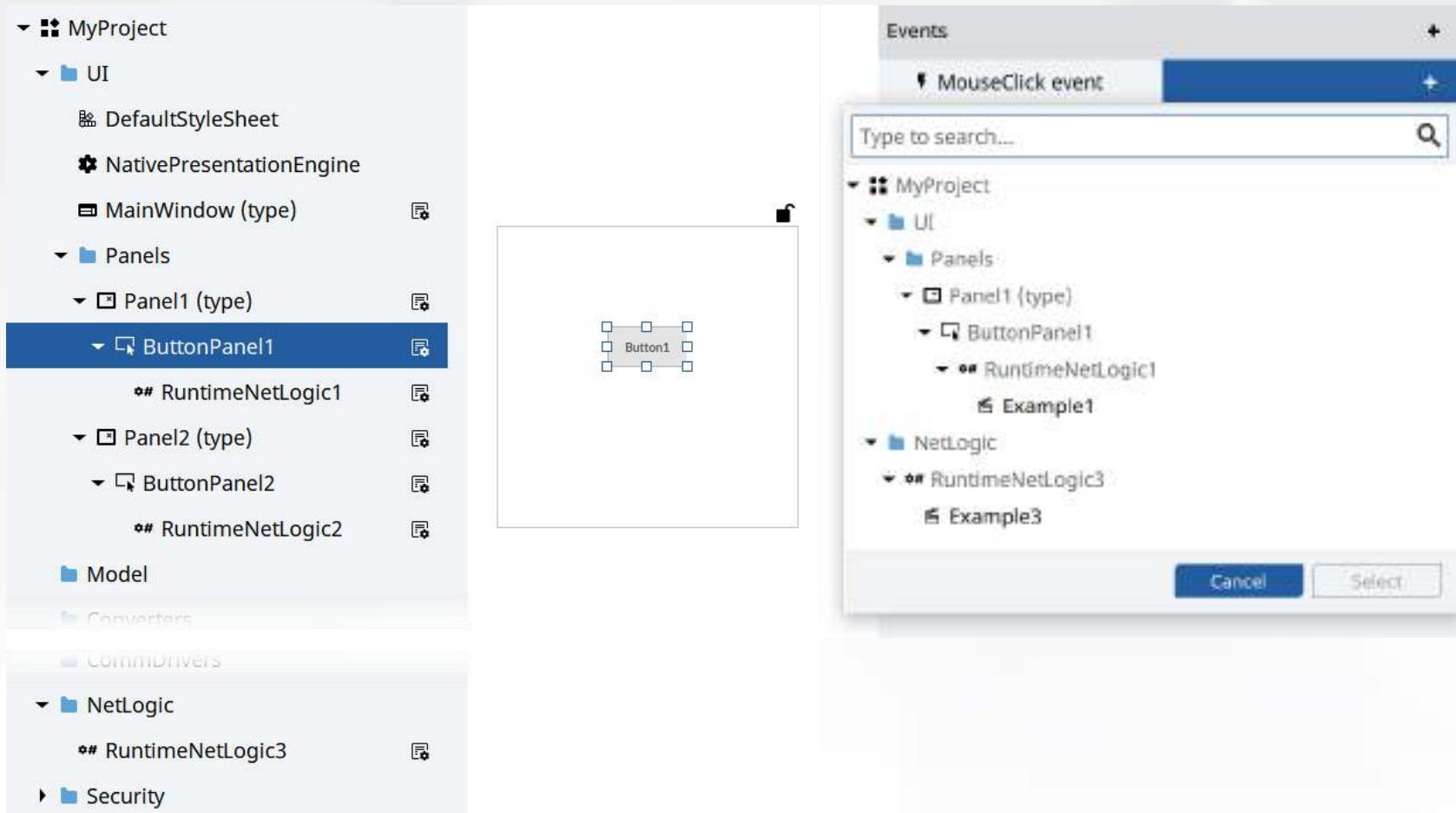
# Runtime NetLogic: scope

- Every session of the Runtime will create an instance of the UI type that was designed (session)
- Elements of a session can only interact with elements from the same session or global scope
- Elements from a global scope cannot interact with elements of a specific session



# Runtime NetLogic: scope

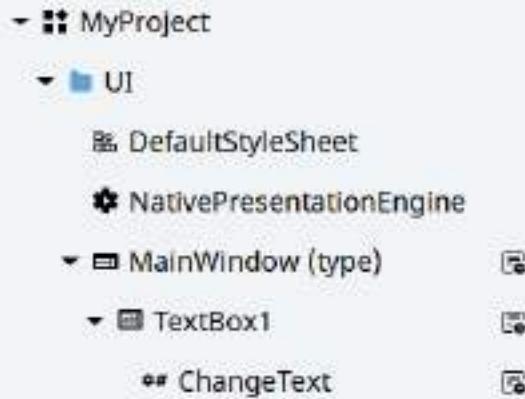
- Button1 of Panel1 cannot access to method exposed by RuntimeNetLogic2 because is related to Panel2 (different type and different scope).
- Panel2 and therefore RuntimeNetLogic2 do not exists when Button1 is loaded
- Dynamic links from a global scope (model variables or PLC tags) cannot point to objects from a session scope



# Runtime NetLogic: usage of "owner"

```
public class ChangeText : BaseNetLogic
{
    [ExportMethod]
    public void ChangeTextRelativePath()
    {
        // WORKING AT RUNTIME!
        Owner.GetVariable("Text").Value = "Text changed using Owner";
    }

    [ExportMethod]
    public void ChangeTextAbsolutePath()
    {
        // NOT WORKING AT RUNTIME!
        Project.Current.Get<TextBox>("UI/MainWindow/TextBox1").Text = "Text changed using Project.Current";
    }
}
```



	Project.Current...	Owner...
Type of path	Absolute	Relative
Runtime usage	can be used to refer nodes <u>outside the UI</u> *	<b>must be used</b> to refer nodes <u>inside the UI</u> *

\* This is due to the UI Sessions, dynamically created at Runtime

# Runtime NetLogic: read/write variables value



```
[ExportMethod]
public void Sum()
{
    var addend1 = (Int32)Project.Current.GetVariable("Model/Variable1").Value;
    var addend2 = (Int32)Project.Current.GetVariable("Model/Variable2").Value;
    Project.Current.GetVariable("Model/Variable3").Value = addend1 + addend2;
}
```

# Runtime NetLogic: subscribe to VariableChange event

```
private IUAVariable addend1;
private IUAVariable addend2;

public override void Start()
{
    addend1 = Project.Current.GetVariable("Model/Variable1");
    addend2 = Project.Current.GetVariable("Model/Variable2");
    addend1.VariableChange += addend1_VariableChange;
    addend2.VariableChange += addend2_VariableChange;
}

private void addend1_VariableChange(object sender, VariableChangeEventArgs e)
{
    Project.Current.GetVariable("Model/Variable3").Value = (Int32)e.NewValue + (Int32)addend2.Value;
}

private void addend2_VariableChange(object sender, VariableChangeEventArgs e)
{
    Project.Current.GetVariable("Model/Variable3").Value = (Int32)addend1.Value + (Int32)e.NewValue;
}

public override void Stop()
{
    addend1.VariableChange -= addend1_VariableChange;
    addend2.VariableChange -= addend2_VariableChange;
}
```

This example executes the Sum when an addend changes, without the need to click the Button

# Runtime NetLogic: set a variable always in use

```
private IUAVariable addend1;
private IUAVariable addend2;
private RemoteVariableSynchronizer variableSynchronizer;

public override void Start()
{
    addend1 = Project.Current.GetVariable("Model/Variable1");
    addend2 = Project.Current.GetVariable("Model/Variable2");
    addend1.VariableChange += addend1_VariableChange;
    addend2.VariableChange += addend2_VariableChange;

    variableSynchronizer = new RemoteVariableSynchronizer();
    variableSynchronizer.Add(addend1);
    variableSynchronizer.Add(addend2);
}

private void addend1_VariableChange(object sender, VariableChangeEventArgs e)
{
    Project.Current.GetVariable("Model/Variable3").Value = (Int16)e.NewValue + (Int16)addend2.Value;
}

private void addend2_VariableChange(object sender, VariableChangeEventArgs e)
{
    Project.Current.GetVariable("Model/Variable3").Value = (Int16)addend1.Value + (Int16)e.NewValue;
}
```

Starting from the previous example, here Variable1 and Variable2 are associated with Modbus registers, but they are not in use (because not linked to any object on page) so they need to be added to a RemoteVariableSynchronizer

# Runtime NetLogic: execute code in asynchronous mode \*

```
public override void Start()
{
    variable1 = Project.Current.GetVariable("Model/Variable1");
    myPeriodicTask = new PeriodicTask(IncrementVariable, 250, LogicObject);
    myPeriodicTask.Start();
    dCounter = 0;
}

private void IncrementVariable(PeriodicTask task)
{
    dCounter = dCounter + 0.05;
    variable1.Value = 50 + Math.Sin(dCounter) * 50;
}

public override void Stop()
{
    myPeriodicTask.Dispose();
}

private PeriodicTask myPeriodicTask;
private IUAVariable variable1;
private double dCounter;
```

\* The asynchronous classes available are:

**PeriodicTask** to create a task that executes code at regular time intervals.

**DelayedTask** to create a task that executes code after a time delay.

**LongRunningTask** to create tasks that require significant time and/or CPU resources.

The default “async” methods in C# cannot be used to access project nodes (to avoid concurrency issues), use the FactoryTalk Optix classes from above instead

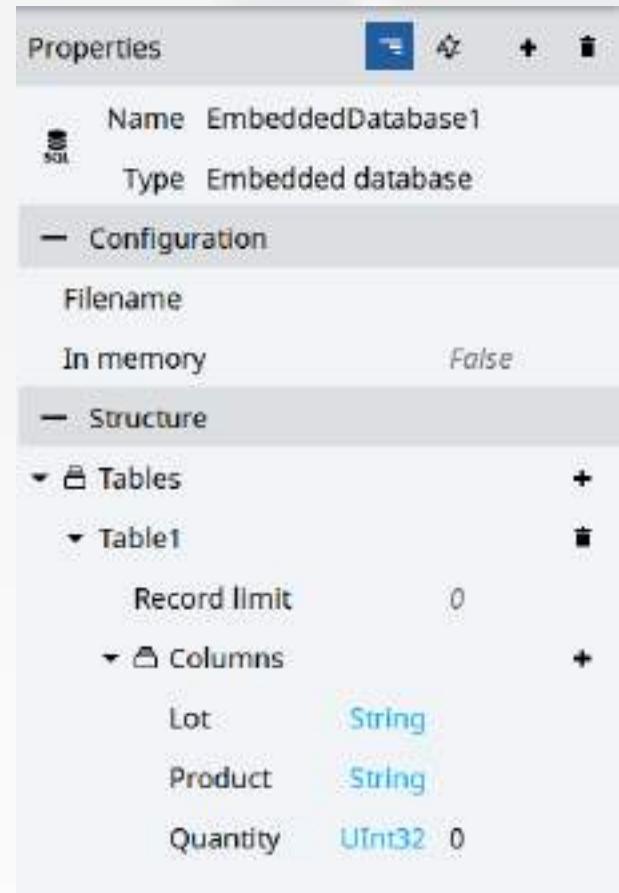
# Runtime NetLogic: execute DB query

```
private Store myStore;
private Table myTable;
private object[,] resultSet;
private string[] header;

public override void Start()
{
    myStore = Project.Current.Get<Store>("DataStores/EmbeddedDatabase1");
    myTable = myStore.Tables.Get<Table>("Table1");
}

[ExportMethod]
public void Insert(string Lot, string Prod, uint Quantity)
{
    object[,] rawValues = new object [1,3];      // insert 1 row with 3 columns
    rawValues[0,0] = Lot;
    rawValues[0,1] = Prod;
    rawValues[0,2] = Quantity;

    string[] columns = new string[3] {"Lot", "Product", "Quantity"};
    myTable.Insert(columns, rawValues);
}
```



# Runtime NetLogic: execute DB query

```
[ExportMethod]
public void Delete(string Lot)
{
    myStore.Query($"DELETE FROM Table1 WHERE Lot='{Lot}'", out header, out resultSet);
}

[ExportMethod]
public void Update(string Lot, uint Quantity)
{
    myStore.Query($"UPDATE Table1 SET Quantity='{Quantity}' WHERE Lot='{Lot}'", out header, out resultSet);
}

[ExportMethod]
public void Select(string Lot)
{
    myStore.Query($"SELECT Product, QUANTITY FROM Table1 WHERE Lot='{Lot}'", out header, out resultSet);
    for (int i = 0; i < resultSet.GetLength(0); i++)
    {
        Log.Info(LogicObject.BrowseName, $"Product = '{resultSet[i,0]}' - Quantity = '{resultSet[i,1]}'");
    }
}
```

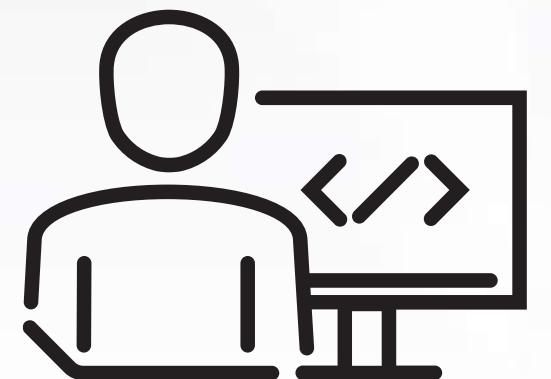
# NetLogic: look at some library scripts

- MQTT push-agent (Runtime)
- FTP Client/Server (Runtime)
- ClockLogic (Runtime)
- Import/Export Alarm (Design-time)
- Import/Export Translations (Design-time)

Most Design-time scripts from TemplateLibrary can be modified to be used at Runtime too!

# Hands-on session

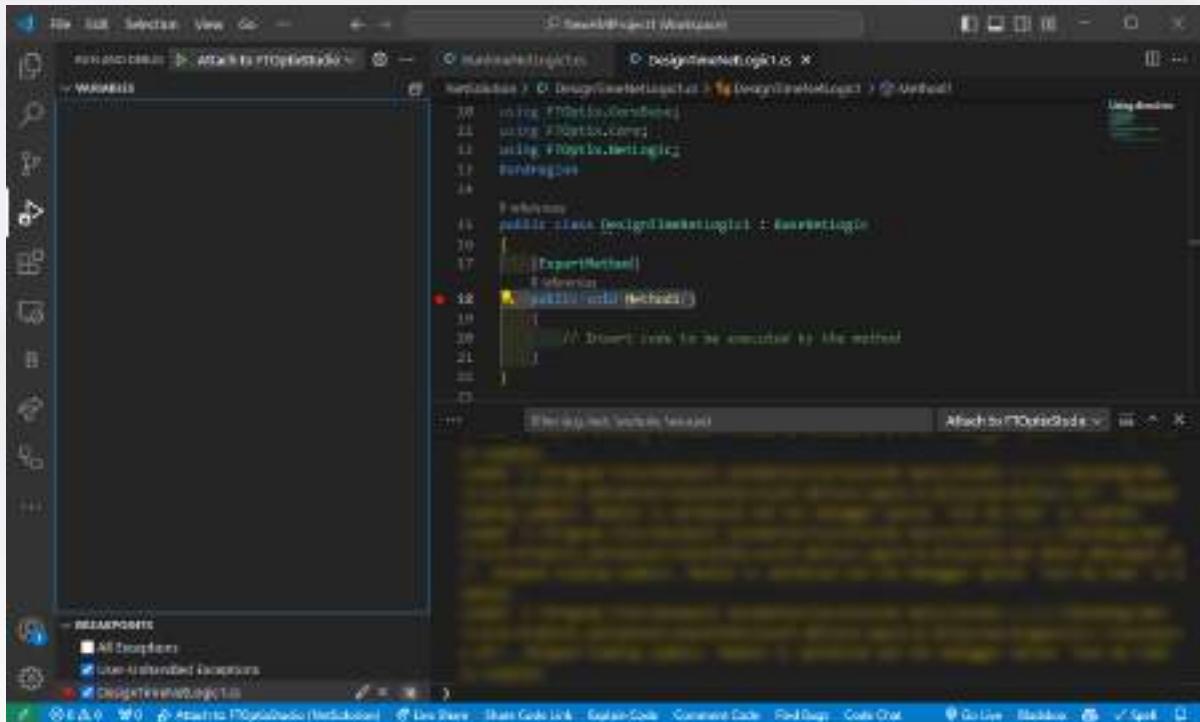
- Write one or more Runtime script
- See what happens when adding an element to a page using Project.Current() vs. Owner.Get()



# Debugging NetLogic scripts



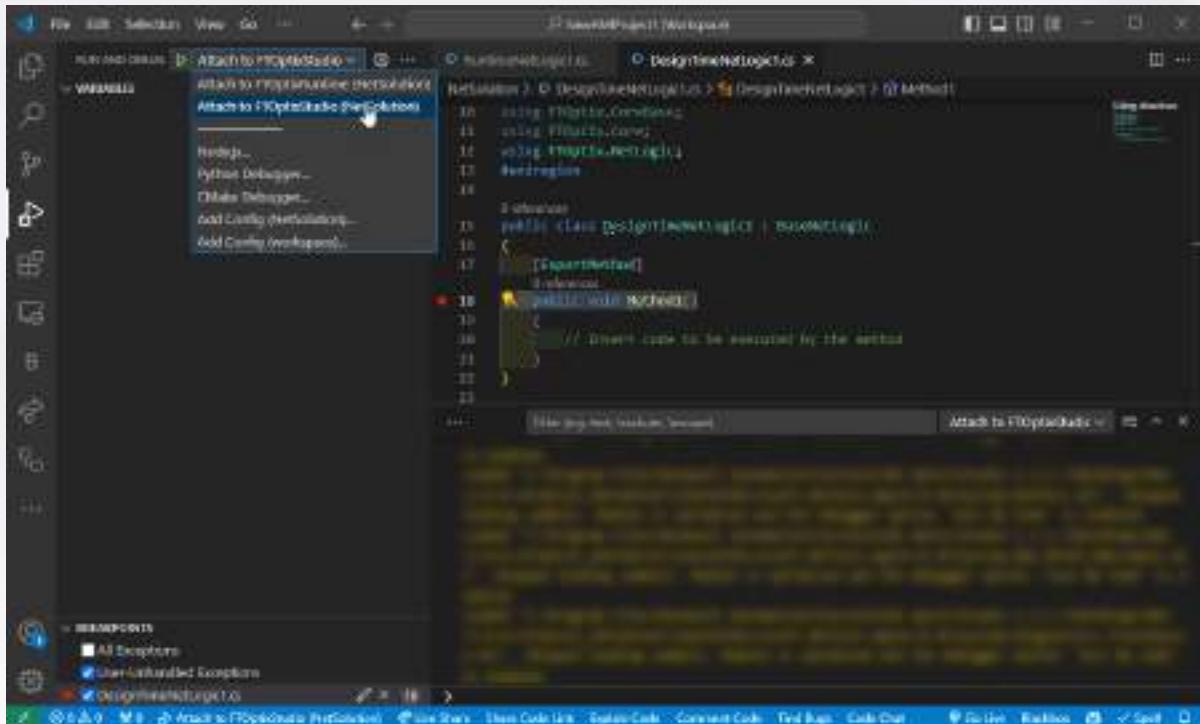
# Debug with Visual Studio Code



- Set a breakpoint by clicking to the left of the row number (a red dot will appear)
- Select
  - Attach to FTOptixStudio for Design time NetLogic
  - Attach to FTOptixRuntime for Runtime NetLogic
- Click "Run and Debug" on the left toolbar or hit F5 on the keyboard
- Execute the script from FactoryTalk Optix Studio or FactoryTalk Optix Runtime
- Execute step-by-step
  - Step Over (F10)
  - Step Into (F11)

Due to a bug of the C# Extension of VSCode, versions of the plugin between 2.4.4 and 2.34.10 may not work when debugging Designtime NetLogic, please upgrade or downgrade as needed.

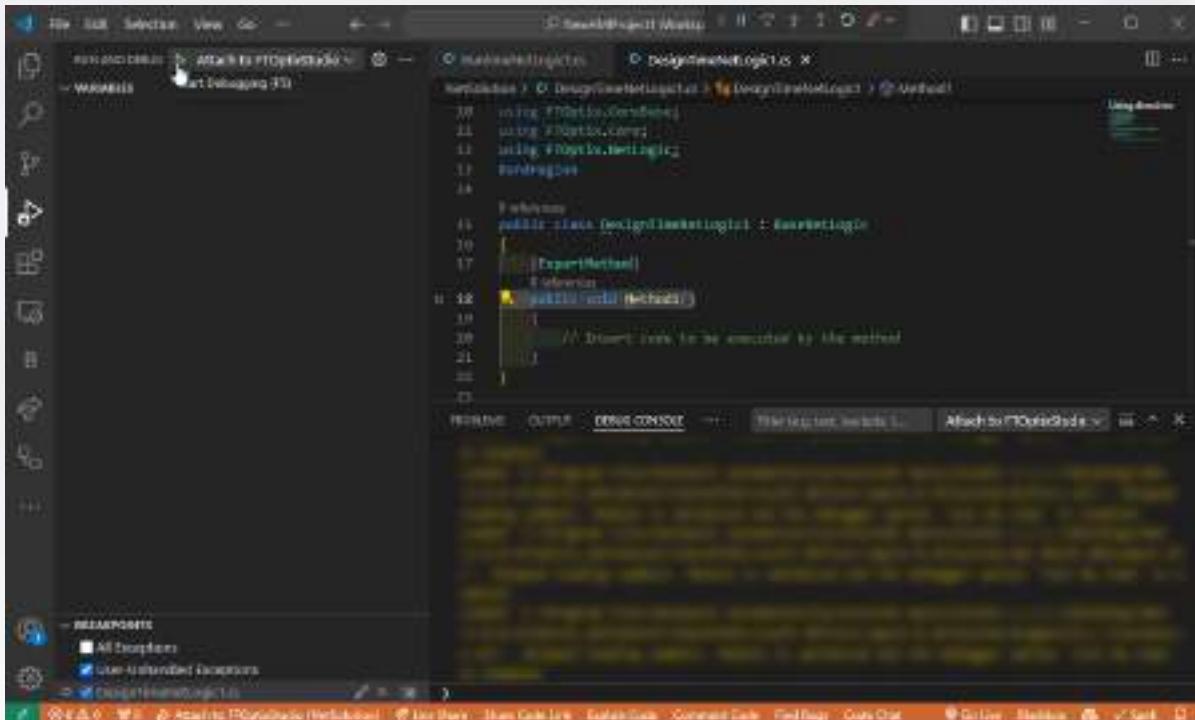
# Debug with Visual Studio Code



- Set a breakpoint by clicking to the left of the row number (a red dot will appear)
- Select
  - Attach to FTOptixStudio for Design time NetLogic
  - Attach to FTOptixRuntime for Runtime NetLogic
- Click "Run and Debug" on the left toolbar or hit F5 on the keyboard
- Execute the script from FactoryTalk Optix Studio or FactoryTalk Optix Runtime
- Execute step-by-step
  - Step Over (F10)
  - Step Into (F11)

Due to a bug of the C# Extension of VSCode, versions of the plugin between 2.4.4 and 2.34.10 may not work when debugging Designtime NetLogic, please upgrade or downgrade as needed.

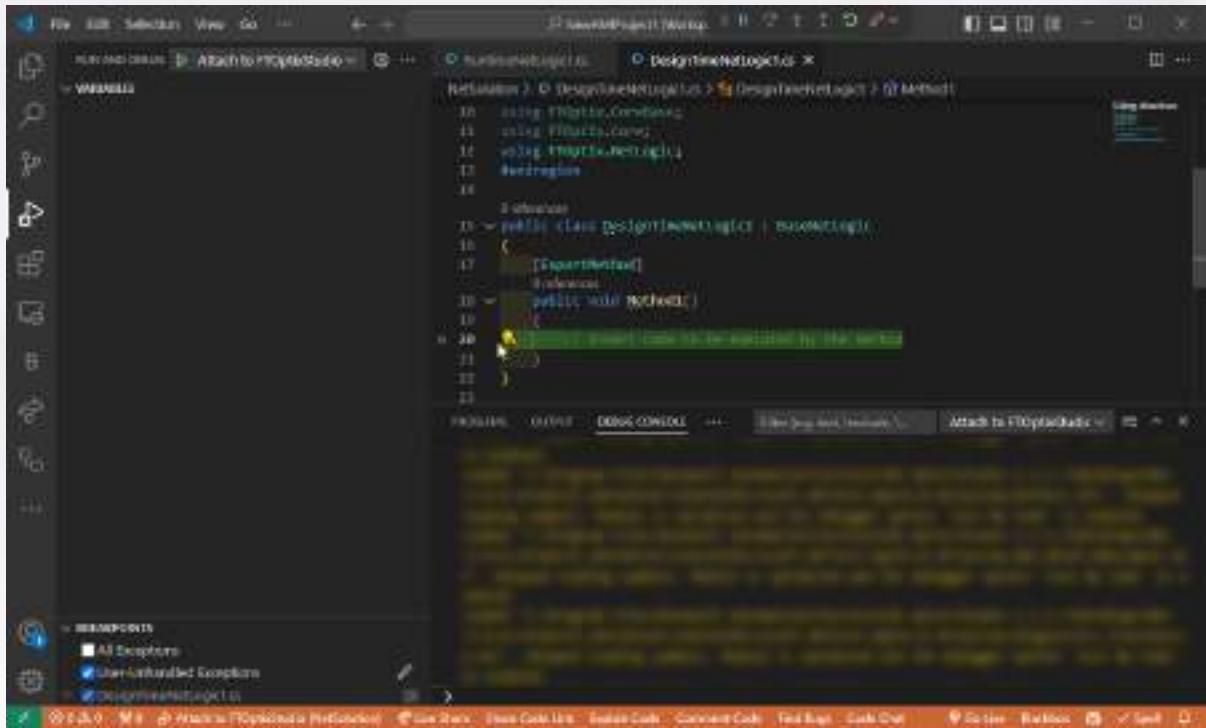
# Debug with Visual Studio Code



- Set a breakpoint by clicking to the left of the row number (a red dot will appear)
- Select
  - Attach to FTOptixStudio for Design time NetLogic
  - Attach to FTOptixRuntime for Runtime NetLogic
- Click "Run and Debug" on the left toolbar or hit F5 on the keyboard
- Execute the script from FactoryTalk Optix Studio or FactoryTalk Optix Runtime
- Execute step-by-step
  - Step Over (F10)
  - Step Into (F11)

Due to a bug of the C# Extension of VSCode, versions of the plugin between 2.4.4 and 2.34.10 may not work when debugging Designtime NetLogic, please upgrade or downgrade as needed.

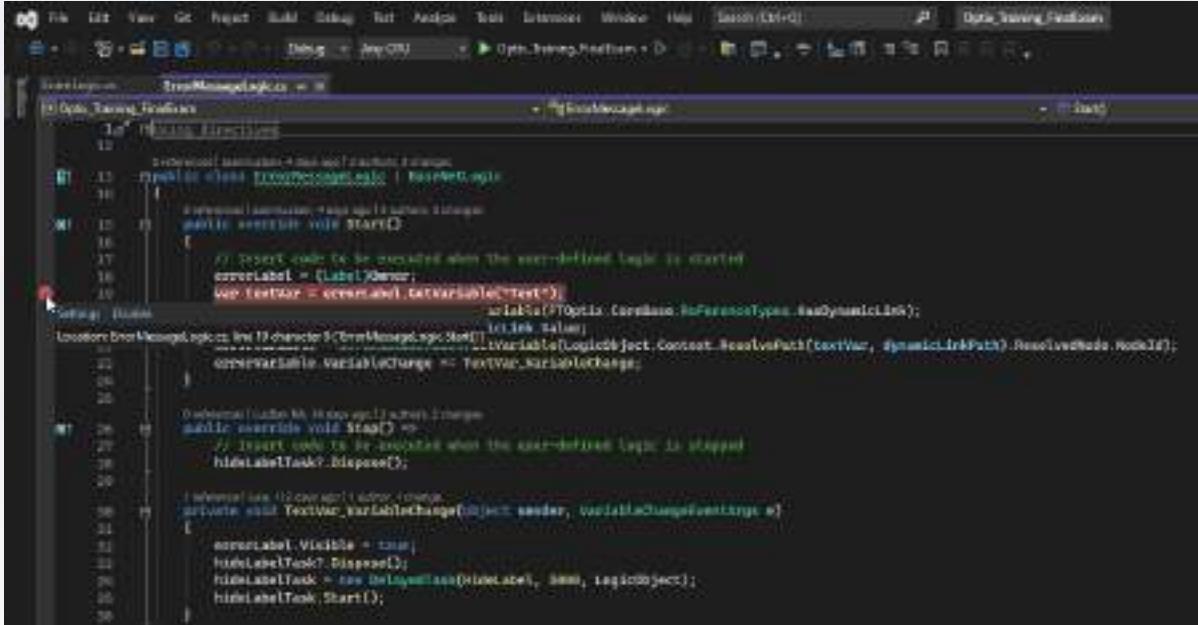
# Debug with Visual Studio Code



- Set a breakpoint by clicking to the left of the row number (a red dot will appear)
- Select
  - Attach to FTOptixStudio for Design time NetLogic
  - Attach to FTOptixRuntime for Runtime NetLogic
- Click "Run and Debug" on the left toolbar or hit F5 on the keyboard
- Execute the script from FactoryTalk Optix Studio or FactoryTalk Optix Runtime
- Execute step-by-step
  - Step Over (F10)
  - Step Into (F11)

Due to a bug of the C# Extension of VSCode, versions of the plugin between 2.4.4 and 2.34.10 may not work when debugging Designtime NetLogic, please upgrade or downgrade as needed.

# Debug with Visual Studio 2022

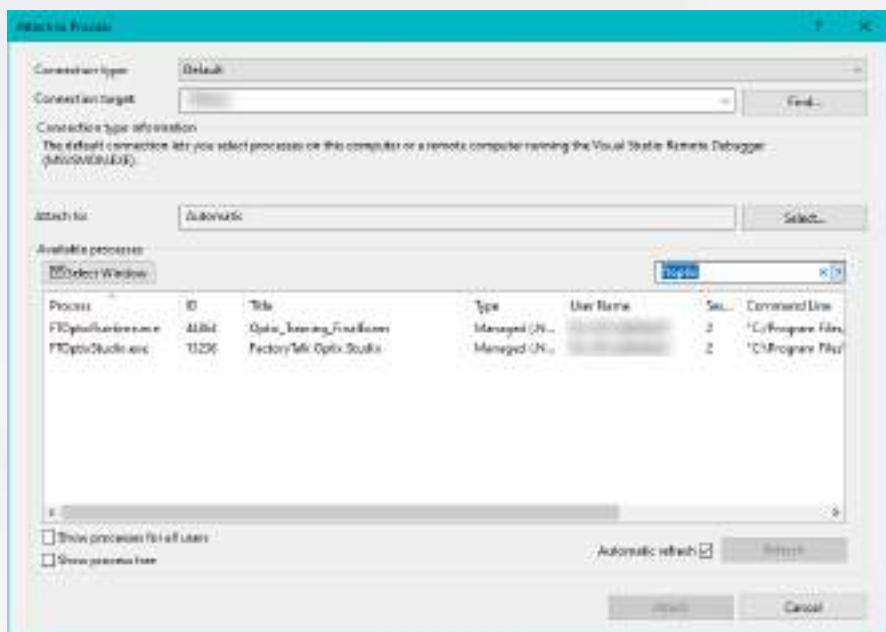


The screenshot shows the Visual Studio 2022 interface with the 'DesignTimeNetLogic.cs' file open in the code editor. The code is a C# class with several methods. Breakpoints are visible as red dots in the margin next to specific lines of code. The 'DesignTime' method is highlighted, indicating it is the current target for debugging.

```
1 // This file contains logic that runs during design time
2 // It is executed when the project is loaded or when the
3 // 'DesignTime' method is called.
4
5 public void DesignTime()
6 {
7     // ...
8 }
9
10 // This method is called when the project is loaded.
11 public void Start()
12 {
13     // ...
14 }
15
16 // This method is called when the project is unloaded.
17 public void Stop()
18 {
19     // ...
20 }
21
22 // This method is called when the project is saved.
23 public void Save()
24 {
25     // ...
26 }
27
28 // This method is called when the project is closed.
29 public void Close()
30 {
31     // ...
32 }
33
34 // This method is called when the project is loaded.
35 public void Load()
36 {
37     // ...
38 }
```

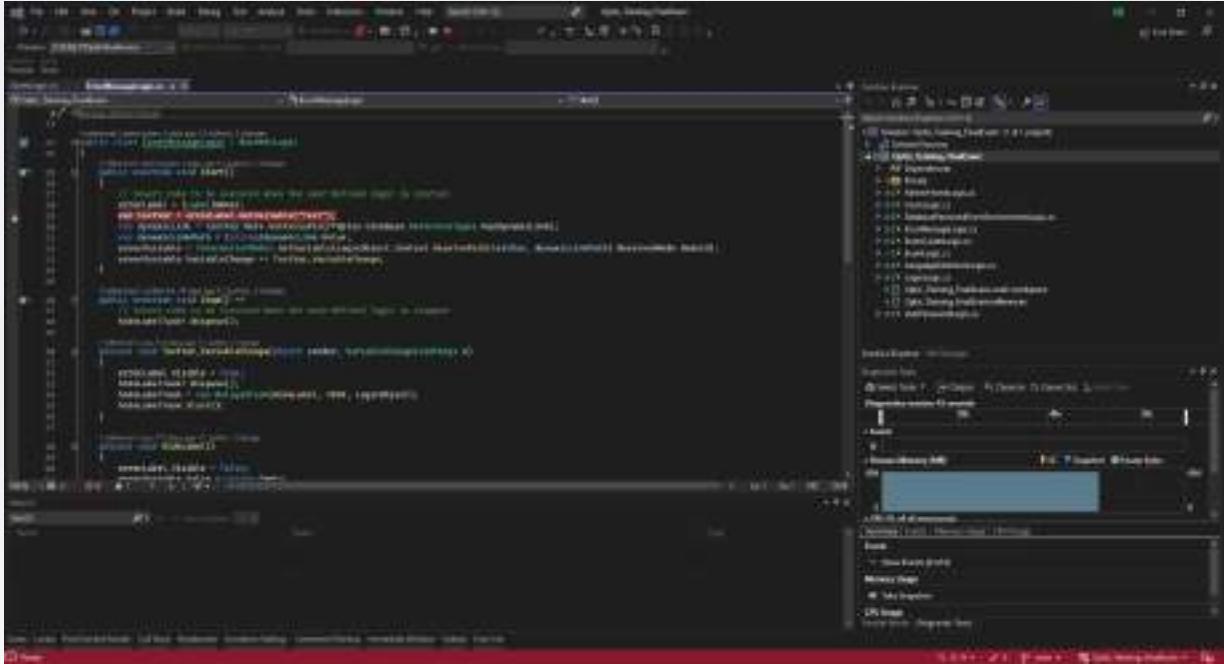
- Open the .NET Solution with VS
- Select the .cs file to debug
- Set a breakpoint by clicking on the left column of the code editor
- DesignTime NetLogic:
  - From the Debug menu, select «Attach to process»
  - Select FTOptixStudio from the menu
  - Click «Attach»
  - Execute the DesignTime method
- RunTime NetLogic
  - From the Debug menu select «Attach to process»
  - Select FactoryTalkOptixRuntime from the menu
  - Click «Attach»
  - Execute the RunTime method

# Debug with Visual Studio 2022



- Open the .NET Solution with VS
- Select the .cs file to debug
- Set a breakpoint by clicking on the left column of the code editor
- DesignTime NetLogic:
  - From the Debug menu, select «Attach to process»
  - Select FTOptixStudio from the menu
  - Click «Attach»
  - Execute the DesignTime method
- RunTime NetLogic
  - From the Debug menu select «Attach to process»
  - Select FactoryTalkOptixRuntime from the menu
  - Click «Attach»
  - Execute the RunTime method

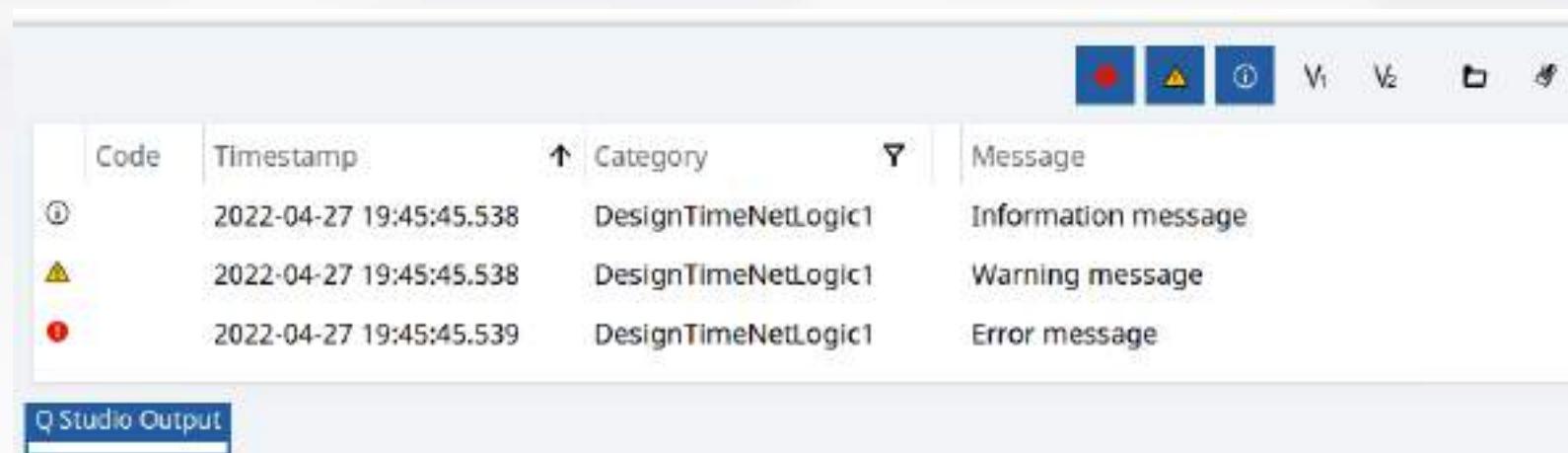
# Debug with Visual Studio 2022



- Open the .NET Solution with VS
- Select the .cs file to debug
- Set a breakpoint by clicking on the left column of the code editor
- DesignTime NetLogic:
  - From the Debug menu, select «Attach to process»
  - Select FTOptixStudio from the menu
  - Click «Attach»
  - Execute the DesignTime method
- RunTime NetLogic
  - From the Debug menu select «Attach to process»
  - Select FactoryTalkOptixRuntime from the menu
  - Click «Attach»
  - Execute the RunTime method

# Debug by logging to output

- Log.Info(LogicObject.BrowseName,"Information message");
- Log.Warning(LogicObject.BrowseName,"Warning message");
- Log.Error(LogicObject.BrowseName,"Error message");



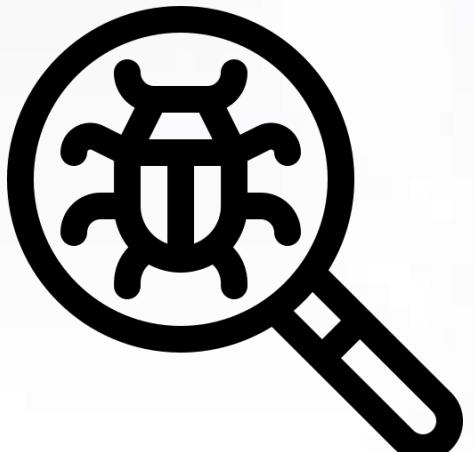
The screenshot shows the 'Studio Output' window from Rockwell Automation's FactoryTalk interface. The window displays a table of log messages with columns for Code, Timestamp, Category, and Message. The table has three rows: one for an Information message, one for a Warning message, and one for an Error message. Each row is preceded by a small colored icon (blue for information, yellow for warning, red for error) and a timestamp (2022-04-27 19:45:45.538 or 2022-04-27 19:45:45.539). The 'Category' column shows 'DesignTimeNetLogic1' for all entries. The 'Message' column contains the text 'Information message', 'Warning message', and 'Error message' respectively. The window also features a toolbar at the top with icons for Stop, Start, and Refresh, and buttons for V1, V2, and a file icon.

Code	Timestamp	Category	Message
Info	2022-04-27 19:45:45.538	DesignTimeNetLogic1	Information message
Warning	2022-04-27 19:45:45.538	DesignTimeNetLogic1	Warning message
Error	2022-04-27 19:45:45.539	DesignTimeNetLogic1	Error message

Studio Output

# Why to debug?

- Debugger is a powerful tool to understand what's happening to the project
  - Helps to understand why some «weird» errors appear, like «Object not set to an instance of an object» (when trying to access a variable that does not exist) or «Unhandled exception»
- When debugging some logic in the Start method of a RunTime NetLogic multiple options are available
  - Add the «ExportMethod» modifier to the Start method and use a button to call it
  - Use a NavigationPanel or PanelLoader to first connect to the debugger and then load the page containing the script
- If you get «The break point will not currently be hit» please check for:
  - Make sure you attached to the right process
  - Make sure the .NET solution compiles
  - Make sure you saved the project both on VS/VSCode and FactoryTalk Optix
  - Make sure the code in the FactoryTalk Optix Runtime is the latest (restart the runtime if needed)
- Remote debugging is also available via SSH on iPC
  - Cannot be used on closed systems (like OptixPanel) where SSH is not available



# Use Retentivity



# Retentivity usage

- Provide storage of changes made at Runtime

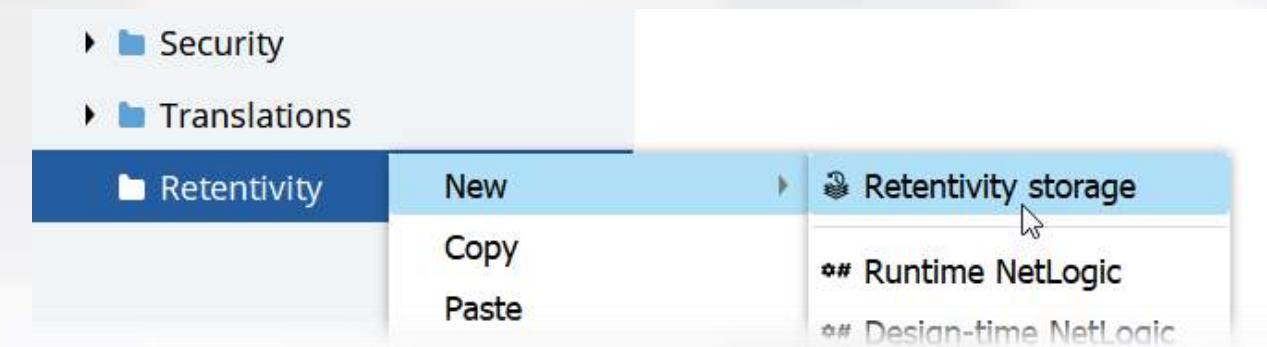
- Retentivity applies to:

- **Variables**,

- **Users**,

- **Dictionary of Translations**,

- or every node of the project



- Every Retentivity storage object is related to an embedded database

# Retentivity properties and location

- Properties

- Nodes:** Folder/node of the project to be retained.  
It's possible to define several nodes on the same Retentivity Storage, or create separated storage
- Write delay:** define how often to save into the database  
Zero means values are saved as soon as a change is detected
- Delta observer enabled:** allow to enable/disable the Retentivity

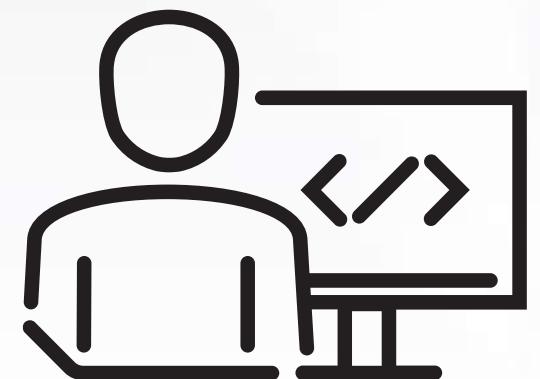
Properties	
Name	RetentivityStorage1
Type	Retentivity storage
Nodes	
Node1	NodeId Model
Node2	NodeId Security
Node3	NodeId LocalizationDictionary1
Write delay	0000:00:00.000
Delta observer enabled	True

- Location:

- Database file can be found in the ApplicationFiles folder of FTOptixApplication
- Example: %localappdata%\Rockwell Automation\FactoryTalk Optix\Emulator\Projects\<ProjectName>\ApplicationFiles

# Hands-on session

- Add retentivity storage to the project
- Add a project's node to the retentivity (like the Model folder)
- Verify that changes made on Model's variable are reteined

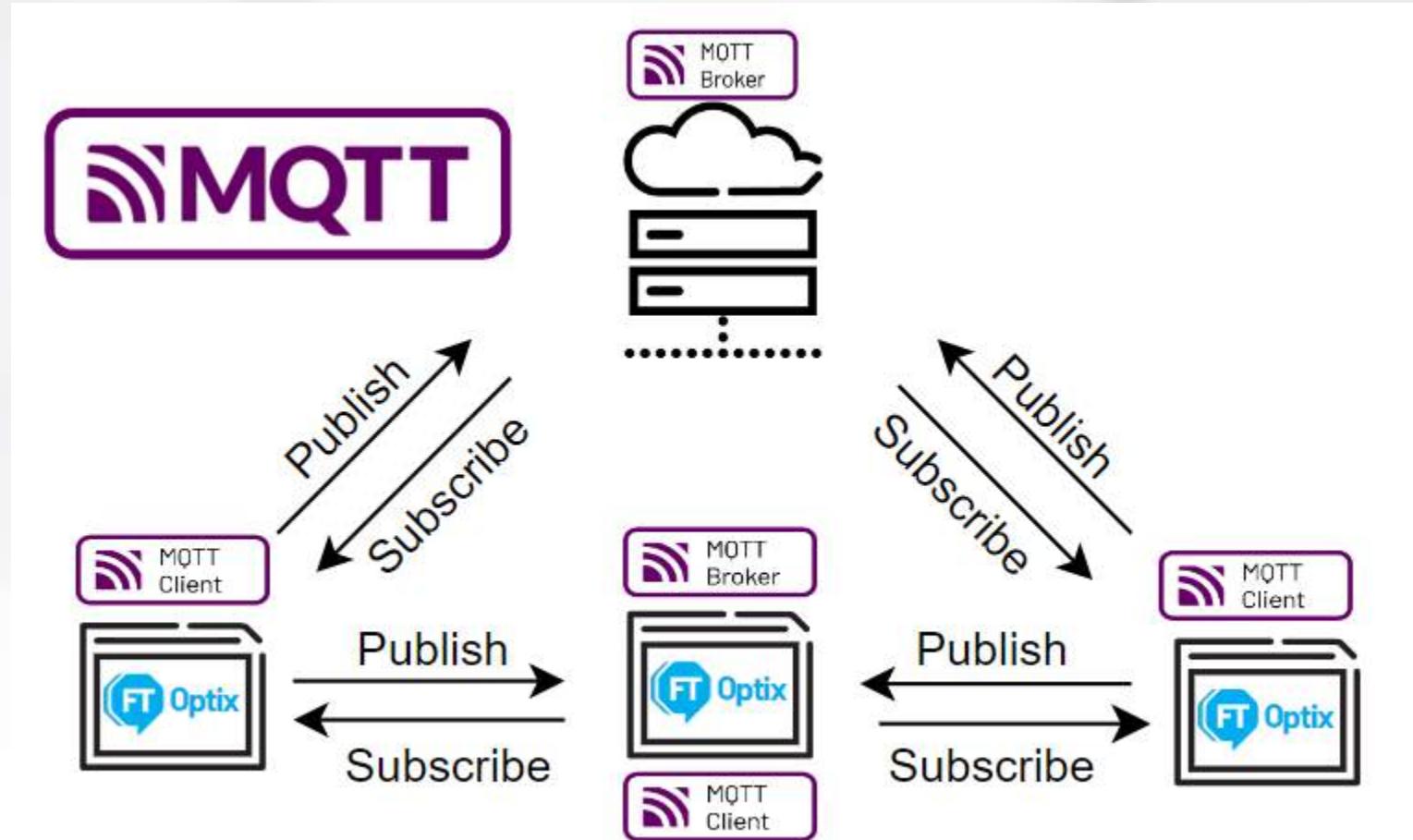


MQTT



# Basic configuration

- **MQTT Broker:** The central server that routes messages between clients in the publish/subscribe model.
- **MQTT Client:** Interacts with the broker to send or receive messages.
- **Publisher:** A client that sends messages to a specific topic on the broker.
- **Subscriber:** A client that listens to topics and receives messages from the broker.



# MQTT broker

## General Configuration

**Listener port:** The port on which the MQTT broker listens for incoming client connections (default: 8883 for secure communication using SSL/TLS).

**Max connections:** The maximum number of simultaneous client connections that the broker can handle (e.g., 25).

**Max QoS:** The highest Quality of Service level supported by the broker:

0: At most once (fire-and-forget).

1: At least once (acknowledged delivery).

2: Exactly once (guaranteed delivery).

**Max in-flight messages:** The maximum number of unacknowledged messages per client that the broker allows to remain "in flight."

**Max queued messages:** The maximum number of messages that can be queued for offline clients.

**Message size limit:** The maximum size (in bytes) of a single MQTT message. A value of 0 indicates no limit.

**Retain available:** Indicates whether the broker supports retained messages. Retained messages store the last published message for a topic and send it to new subscribers.

Properties	
 Name	MQTTBroker
Type	MQTT Broker
General configuration	
Listener port	8883
Max connections	25
Max QoS	2 - Exactly once
Max in-flight messages	20
Max queued messages	1000
Message size limit	0
Retain available	True
Security	
Require client certificate	False
CA certificate file	<a href="#">Browse</a>
Broker certificate file	<a href="#">Browse</a>
Broker private key file	<a href="#">Browse</a>

# MQTT broker

## Security

**Require client certificate:** Determines whether clients must provide a certificate for mutual TLS authentication (True or False).

**CA certificate file:** The path to the Certificate Authority (CA) certificate file, used to verify the authenticity of client certificates.

**Broker certificate file:** The path to the broker's SSL/TLS certificate file, used to encrypt communication.

**Broker private key file:** The path to the private key file corresponding to the broker's SSL/TLS certificate, used for secure communication.

This setup is typical for a secure and configurable MQTT broker deployment, ensuring flexibility and security.

Properties	
Name	MQTTBroker
Type	MQTT Broker
General configuration	
Listener port	8883
Max connections	25
Max QoS	2 - Exactly once
Max in-flight messages	20
Max queued messages	1000
Message size limit	0
Retain available	True
Security	
Require client certificate	False
CA certificate file	<a href="#">Browse</a>
Broker certificate file	<a href="#">Browse</a>
Broker private key file	<a href="#">Browse</a>

# MQTT Client

## Client

**Broker address:** The address of the broker to connect to.

**Port:** Communication port number.

**ClientID:** The Client identifier that is unique on the server.

Properties	
 M0	Name MQTTClient1
	Type MQTT Client
Client	
Broker address	localhost
Port	8883
Client Id	FTOptix-1
Security	
SSL/TLS enabled	True
Validate broker certificate	True
CA certificate file	<a href="#">Browse</a>
Client certificate file	<a href="#">Browse</a>
Client private key file	<a href="#">Browse</a>
User identity	
User identity type	Anonymous

# MQTT Client

## Security

**SSL/TLS Enabled:** Enabling secure connection between FT Optix MQTT Client and MQTT Broker. Once activated 'mqqt' protocol is being used.

**Validate broker certificate:** Activates/deactivates validation process of MQTT broker certificate. During the SSL certificate verification process, the FT Optix MQTT client checks the digital signature of the broker certificate to ensure that it has been issued by a trusted certificate authority (CA). The client also verifies that the certificate has not expired.

**CA certificate file:** A Certificate Authority certificate that has signed the server certificate on the MQTT Broker. It is required once SSL/TLS is activated to establish secure connection with the broker. It must be found in \FactoryTalk Optix\Projects\ProjectName\ProjectFiles\PKI\Own\Certs folder.

**Client certificate file:** FT Optix MQTT Client certificate. A client certificate identifies the client just like the server certificate identifies the server. If this property stays empty only broker certificate is used for authentication. It must be found in \FactoryTalk Optix\Projects\ProjectName\ProjectFiles\PKI\Own\Certs folder.

**Client private key file:** FT Optix MQTT Client private key. It must be found in \FactoryTalk Optix\Projects\ProjectName\ProjectFiles\PKI\Own\Certs folder.

Properties	
 Name	MQTTClient1
Type	MQTT Client
Client	
Broker address	localhost
Port	8883
Client Id	FTOptix-1
Security	
SSL/TLS enabled	True
Validate broker certificate	True
CA certificate file	<a href="#">Browse</a>
Client certificate file	<a href="#">Browse</a>
Client private key file	<a href="#">Browse</a>
User identity	
User identity type	Anonymous

# MQTT Client

## User identity

**User Identity type:** Specify user when accessing the MQTT Broker:

- Anonymous (default) - No login is performed, access the server as Anonymous user
- Username/Password - Specify a username and password combination for logon.

Properties	
 M0	Name <b>MQTTClient1</b>
	Type <b>MQTT Client</b>
Client	
Broker address	<i>localhost</i>
Port	<i>8883</i>
Client Id	<i>FTOptix-1</i>
Security	
SSL/TLS enabled	<i>True</i>
Validate broker certificate	<i>True</i>
CA certificate file	<a href="#">Browse</a>
Client certificate file	<a href="#">Browse</a>
Client private key file	<a href="#">Browse</a>
User identity	
User identity type	<i>Anonymous</i>

# MQTT Client/Publisher

## Sampling mode:

- **None.** Automatic recording is disabled, can record by invoking the Sample method;
- **Periodic.** At regular intervals records the values of all the selected variables;
- **Change in value.** At regular intervals records only the values of the selected variables that have changed with respect to the previous sampling.

**Sampling period:** Interval in hours, with millisecond precision, between one sampling and the next, in the Periodic mode.

**Pooling time:** Interval in hours, with millisecond precision, between one sampling and the next, in the Value change mode.

**Folder:** This folder contains variables whose values will be sent to the broker. The Folder node must be a global object and cannot be a session-based object.

**Topic:** Defines MQTT topic on which we are sending/publishing data. (Topic is required. Once topic is empty Publisher will generate warning message during first publish. RT will not stop and other mqtt functions will keep working, only this specific Publisher will be stopped.)

Properties	
 MQ	Name MQTTPublisher1
	Type MQTT Publisher
Sampling mode	Periodic
Sampling period	0000:00:01.000
Folder	
Topic	
QoS	0 - At most once
Retain	False

# MQTT Client/Publisher

**QoS:** MQTT Quality of Service (QoS), defines the guarantee of delivery for a specific message (0, 1, 2).

**0 - At most once** (May lose messages).

**1 - At least once** (Guarantees the message delivery but potentially duplicate messages).

**2 - Exactly once** (Ensures messages are delivered exactly once without duplication).

**Retain:** Retain message on the topic even after read

True - Enables Retain functionality

False - Disables Retain functionality

Properties	
 MQ	Name <b>MQTTPublisher1</b>
	Type <b>MQTT Publisher</b>
Sampling mode	<b>Periodic</b>
Sampling period	<b>0000:00:01.000</b>
Folder	
Topic	
QoS	<b>0 - At most once</b>
Retain	<b>False</b>

# MQTT Client/Subscriber

**Folder:** This folder node contains variables on which the Subscriber will copy the values read from the broker.

**Topic:** A single topic on a Broker to subscribe. Topic is required.

**QoS:** MQTT Quality of Service (QoS), defines the guarantee of delivery for a specific message (0, 1, 2).

**0 - At most once** (May lose messages).

**1 - At least once** (Guarantees the message delivery but potentially duplicate messages).

**2 - Exactly once** (Ensures messages are delivered exactly once without duplication).

Properties	
 M0	Name MQTTPublisher1
	Type MQTT Publisher
Sampling mode	Periodic
Sampling period	0000:00:01.000
Folder	
Topic	
QoS	0 - At most once
Retain	False

# Current Limitations

## Supported Data Types:

- The publish/subscribe mechanism currently supports **only scalar variables**.
- Structures(UDTs), folders, or other complex data types are **not supported**.

## Custom Payloads:

- It is not possible to configure a custom payload for MQTT messages currently.
- Workaround: use the Publish method and provide a valid payload

## Sparkplug B:

- Not supported yet.

# MQTT Token usage

Basic Concept of the License

## 1. MQTT Clients without a Broker (pointing to localhost):

- Each client that connects to its local broker (localhost) consumes 1 token each.
- If you have  $n$  clients, the total cost will be  $n$  tokens.

## 2. MQTT Clients with a Central Broker:

- If you configure a central broker and all clients connect to it:
  - The clients do not consume tokens.
  - Only the broker consumes 1 token, regardless of the number of connected clients.

## 3. MQTT Clients with Different Brokers:

- If the clients connect to different brokers (e.g., Broker A, Broker B, etc.):
  - Each broker consumes 1 token.
  - If there are  $n$  different brokers, the total cost will be  $n$  tokens, regardless of the number of clients connected to each broker.

# MQTT Utilities

MQTT Official website: [MQTT - The Standard for IoT Messaging](#)

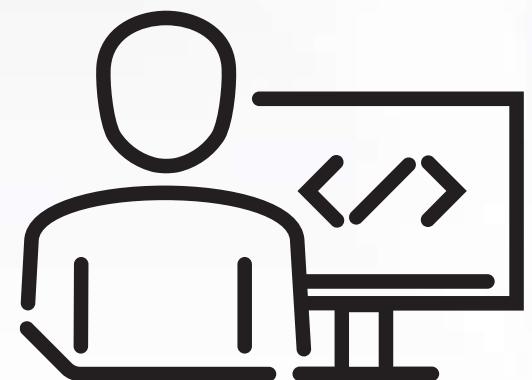
MQTT 5 Standard: [MQTT Version 5.0](#)

This tool helps us to connect to a broker and to subscribe to a specific topic, checking all the packets sent. It also allows us to publish to a topic.

[MQTTX: Your All-in-one MQTT Client Toolbox](#)

# Hands-on session

- Define an MQTT Broker: Set up the MQTT broker to act as the central hub for communication.
  - Key configurations to cover:
    - Listener port (e.g., 1883 for standard or 8883 for SSL/TLS).
    - Retained messages (optional, if applicable to your example).
    - Max connections (adjust as needed based on class size).
- Define an MQTT Client: introduce the concept of an MQTT client, explaining its dual roles as a Publisher and Subscriber.
  - Publisher: configure the client to publish updates to a specific topic (e.g., sensor/temperature).
  - Create a variable that triggers a message when its value changes.
  - Subscriber: Configure the client to subscribe to the same topic (sensor/temperature) and react to updates published by the Publisher.
- Key learning points: Ensure the topic names match exactly (case-sensitive).
- Create Variables that Synchronize via MQTT: Demonstrate how to bind variables to MQTT topics in the software or platform being used.
- Example: create a "source" variable (e.g., temperature) that updates periodically or based on variable change. Create a "target" variable (e.g., displayTemperature) that listens for updates via MQTT and synchronizes automatically. Ensure the variable changes are observable in real-time.
- Test the configuration: Publish a value change (e.g., simulate a sensor reading update). Verify that the subscriber's variable updates automatically when the publisher sends data.



# REST APIs



# REST APIs – Intro to REST

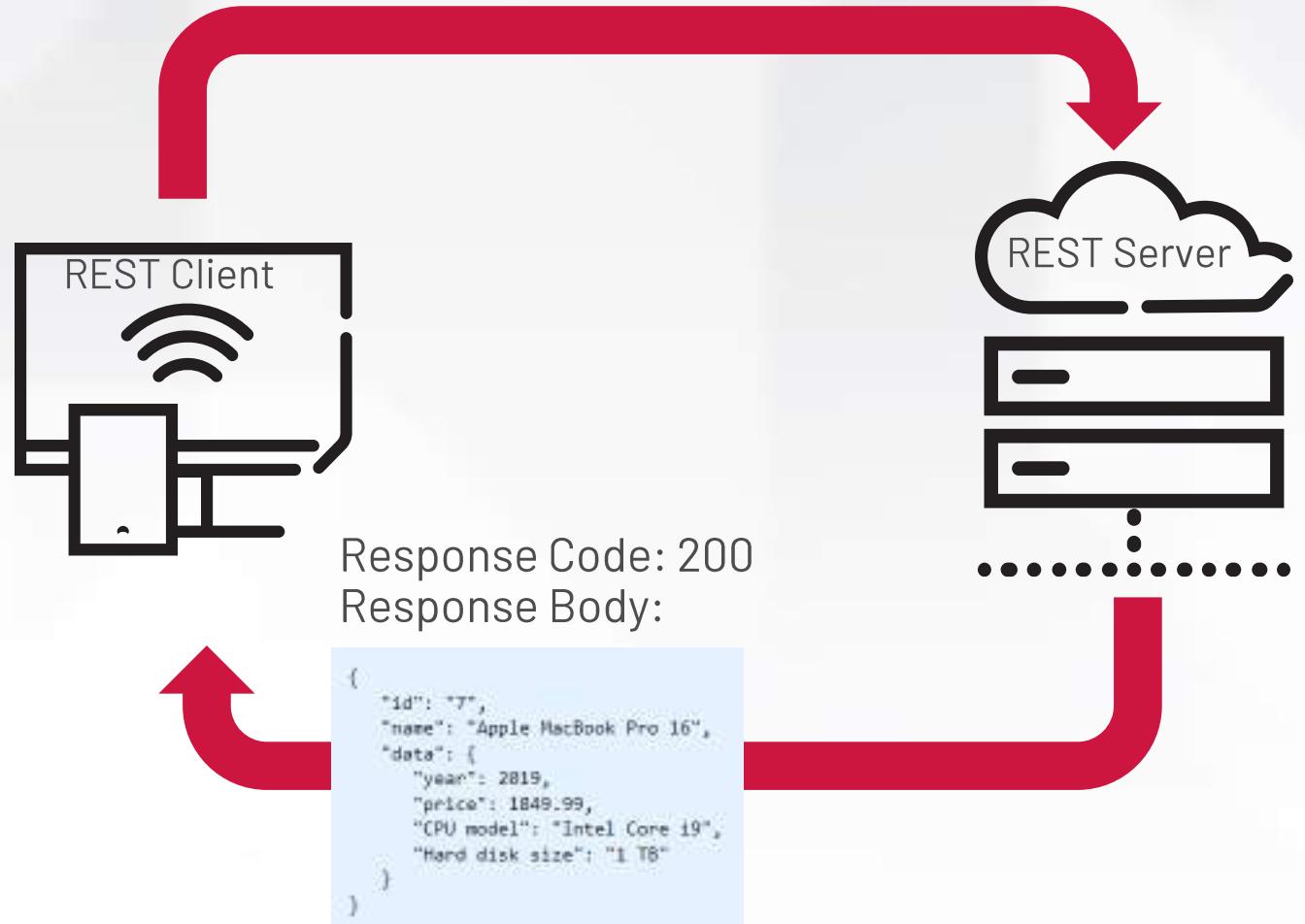
- REST: Representational State Transfer
- Most commonly used protocol for interacting with Web Services
- Protocol uses HTTP / HTTPS methods as the delivery mechanism
- Example Web Services:
  - Google Search (Search the web)
  - Google Maps (Get directions)
  - National Weather Service (Get weather conditions)
  - Twilio (Send Email / Text)
  - Yahoo Finance (Get stock information)
  - SAP (Get business data)
  - ...



# REST APIs – HTTP Methods / Response

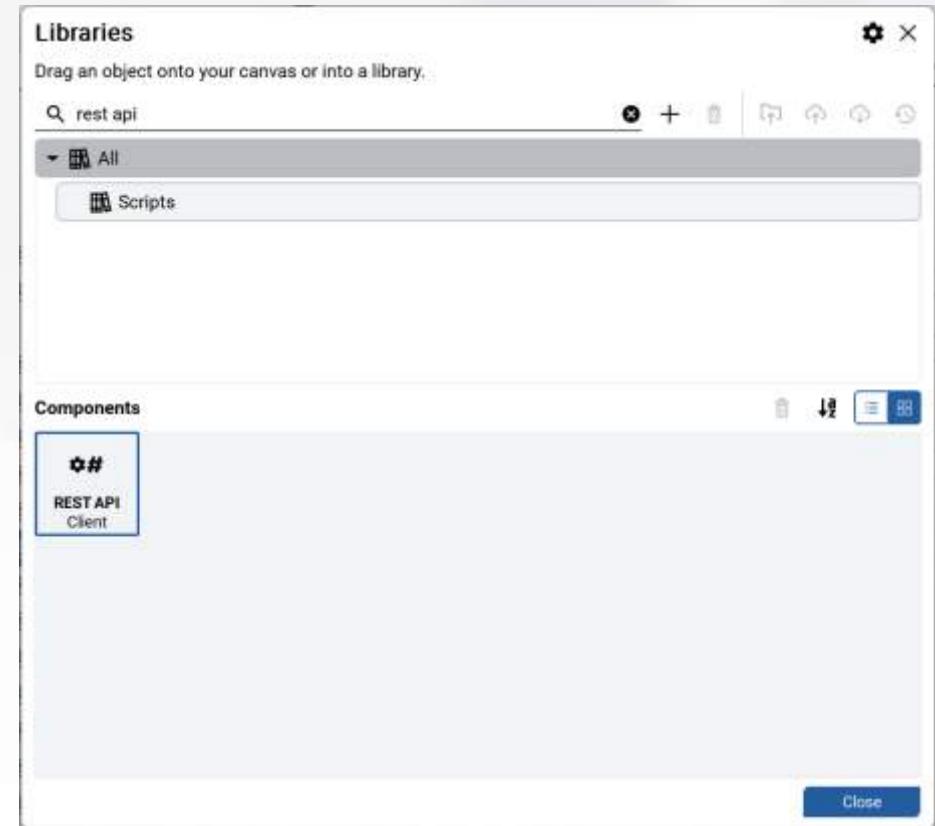
- CRUD: CREATE, READ, UPDATE, DELETE
- HTTP CRUD Methods
  - CREATE: **POST**
  - READ: **GET**
  - UPDATE: **PUT**
  - DELETE: **DELETE**
- HTTP Response:
  - Status Code Integer
    - 200: OK
    - 404: Not Found
    - ...
  - Response Body
    - Typically JSON format

GET https://api.restful-api.dev/objects/7



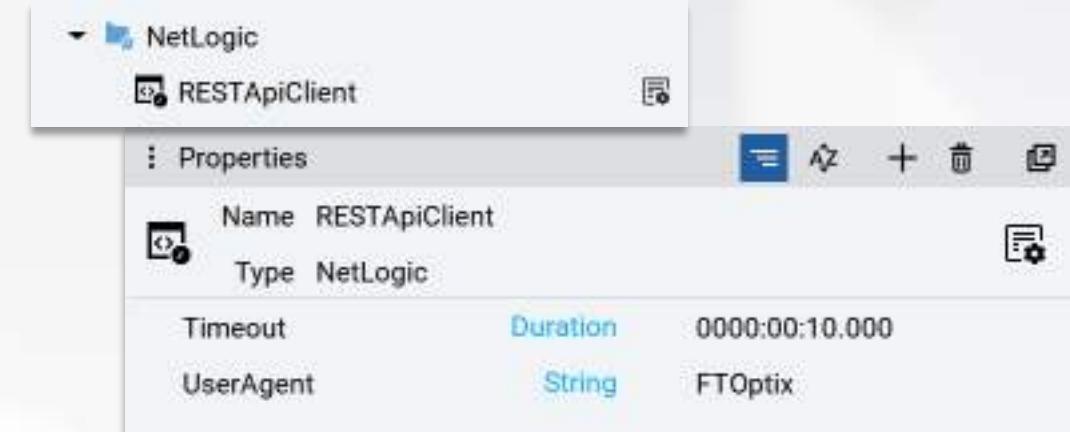
# REST APIs – Implementing in Optix

- Prebuilt NetLogic script in Optix Scripts Library
- ‘REST API Client’ Script supported HTTP methods:
  - GET
  - POST
  - PUT



# REST APIs – Import / Configure RESTApiClient

- Add RESTApiClient to Project
- Configure:
  - Timeout: Defaults to 10s
  - UserAgent: User defined string for client identification. Most REST web services consider this optional. Default: "FTOptix"



# REST APIs - RESTApiClient NetLogic Exploration

- Three Methods available: Get, Post, Put
- Get contains 3 input parameters:
  - apiURL
  - queryString
  - bearerToken
- Post and Put contain 4 input parameters:
  - apiURL
  - requestBody
  - bearerToken
  - contentType
- Each Method contains 'out' parameters with the Response Code (code) and Response Body(response)

```
[ExportMethod]
public void Get(string apiUrl, string queryString, string bearerToken, out string response, out int code)
{
    TimeSpan timeout = TimeSpan.FromMilliseconds(GetTimeout());
    UriBuilder uriBuilder = new UriBuilder(apiUrl);
    uriBuilder.Query = queryString;

    var requestMessage = BuildMessage(HttpMethod.Get, uriBuilder.Uri, "", bearerToken, "");
    var requestTask = PerformRequest(requestMessage, timeout);
    var httpResponse = requestTask.Result;

    (response, code) = (httpResponse.Payload, httpResponse.Code);
}

[ExportMethod]
public void Post(string apiUrl, string requestBody, string bearerToken, string contentType, out string response, out int code)
{
    TimeSpan timeout = TimeSpan.FromMilliseconds(GetTimeout());
    UriBuilder uriBuilder = new UriBuilder(apiUrl);

    var requestMessage = BuildMessage(HttpMethod.Post, uriBuilder.Uri, requestBody, bearerToken, contentType);
    var requestTask = PerformRequest(requestMessage, timeout);
    var httpResponse = requestTask.Result;

    (response, code) = (httpResponse.Payload, httpResponse.Code);
}

[ExportMethod]
public void Put(string apiUrl, string requestBody, string bearerToken, string contentType, out string response, out int code)
{
    TimeSpan timeout = TimeSpan.FromMilliseconds(GetTimeout());
    UriBuilder uriBuilder = new UriBuilder(apiUrl);

    var requestMessage = BuildMessage(HttpMethod.Put, uriBuilder.Uri, requestBody, bearerToken, contentType);
    var requestTask = PerformRequest(requestMessage, timeout);
    var httpResponse = requestTask.Result;

    (response, code) = (httpResponse.Payload, httpResponse.Code);
}
```

# REST APIs - GET Example (Get a list of all objects)

- Get contains 3 input parameters:
  - apiURL: "https://api.restful-api.dev/objects"
  - queryString: NULL (No Parameters)
  - bearerToken: NULL (No Security)

The screenshot shows a REST API documentation interface. On the left, there is a sidebar with various endpoints:

- GET** LIST OF ALL OBJECTS
- GET** LIST OF OBJECTS BY IDS
- GET** SINGLE OBJECT
- POST** ADD OBJECT
- PUT** UPDATE OBJECT
- PATCH** PARTIALLY UPDATE OBJECT
- DELETE** DELETE OBJECT

In the center, there is a **Request** section with the URL <https://api.restful-api.dev/objects>. To the right, there is a **Response** section showing a JSON array of objects:

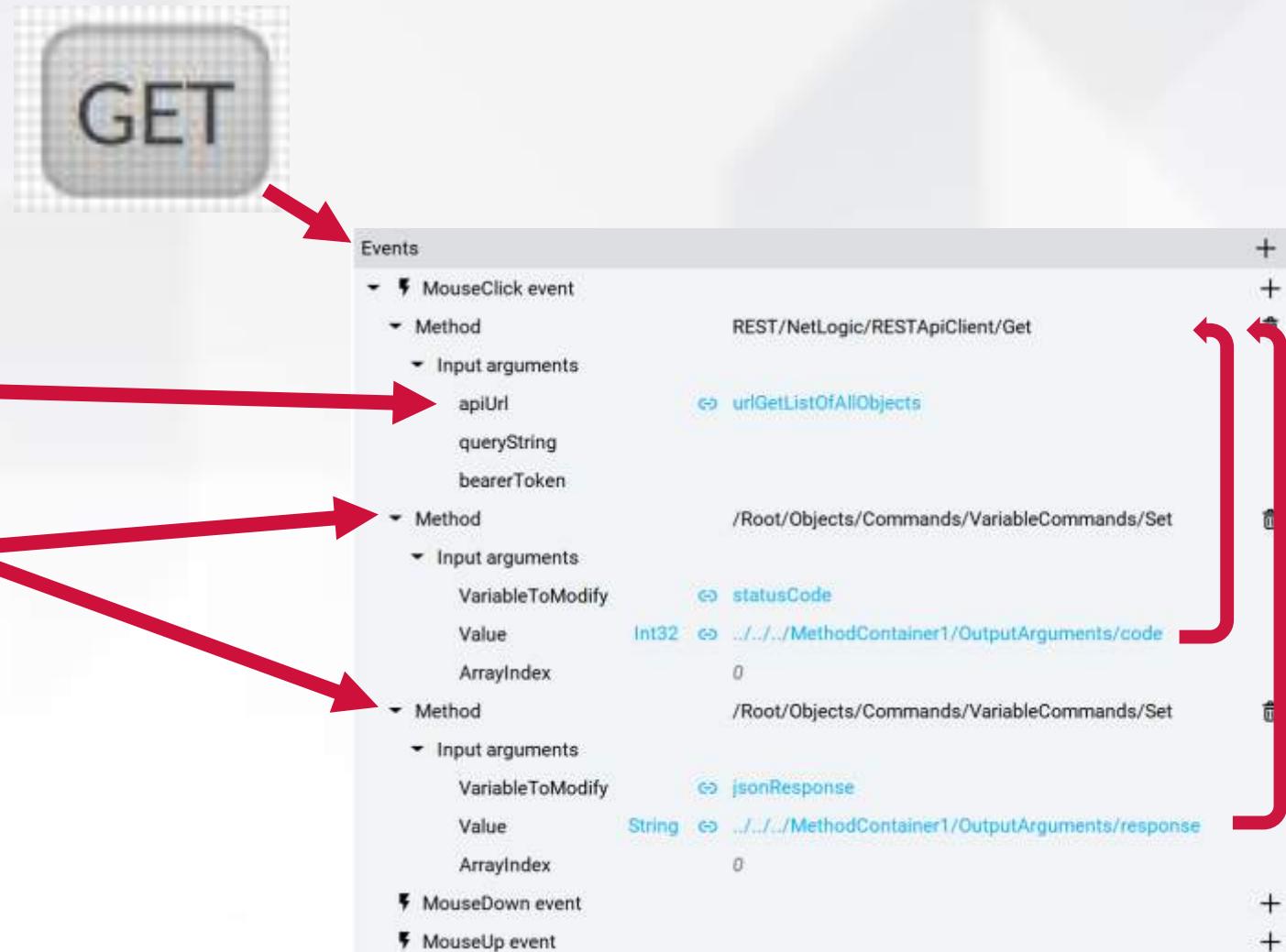
```
[{"id": "1", "name": "Google Pixel 6 Pro", "data": {"color": "Cloudy White", "capacity": "128 GB"}}, {"id": "2", "name": "Apple iPhone 12 Mini, 256GB, Blue", "data": null}, {"id": "3", "name": "Apple iPhone 12 Pro Max", "data": {"color": "Cloudy White", "capacity": "512 GB"}]}
```

\*<https://restful-api.dev>

# REST APIs - GET Example (Get a list of all objects)

- Get contains 3 input parameters:
  - apiURL: "https://api.restful-api.dev/objects"
  - queryString: NULL (No Parameters)
  - bearerToken: NULL (No Security)

1. Call the RESTApiClient/Get method with 'apiURL' set.
2. Use VariableCommands/Set to gather the RESTApiClient/Get Output Parameters



# REST APIs - GET Example (Get a list of specific objects)

- Get contains 3 input parameters:
  - apiURL: "https://api.restful-api.dev/objects"
  - queryString: "id=3&id=5&id=10" (everything after "?" in request URL)
  - bearerToken: NULL (No Security)

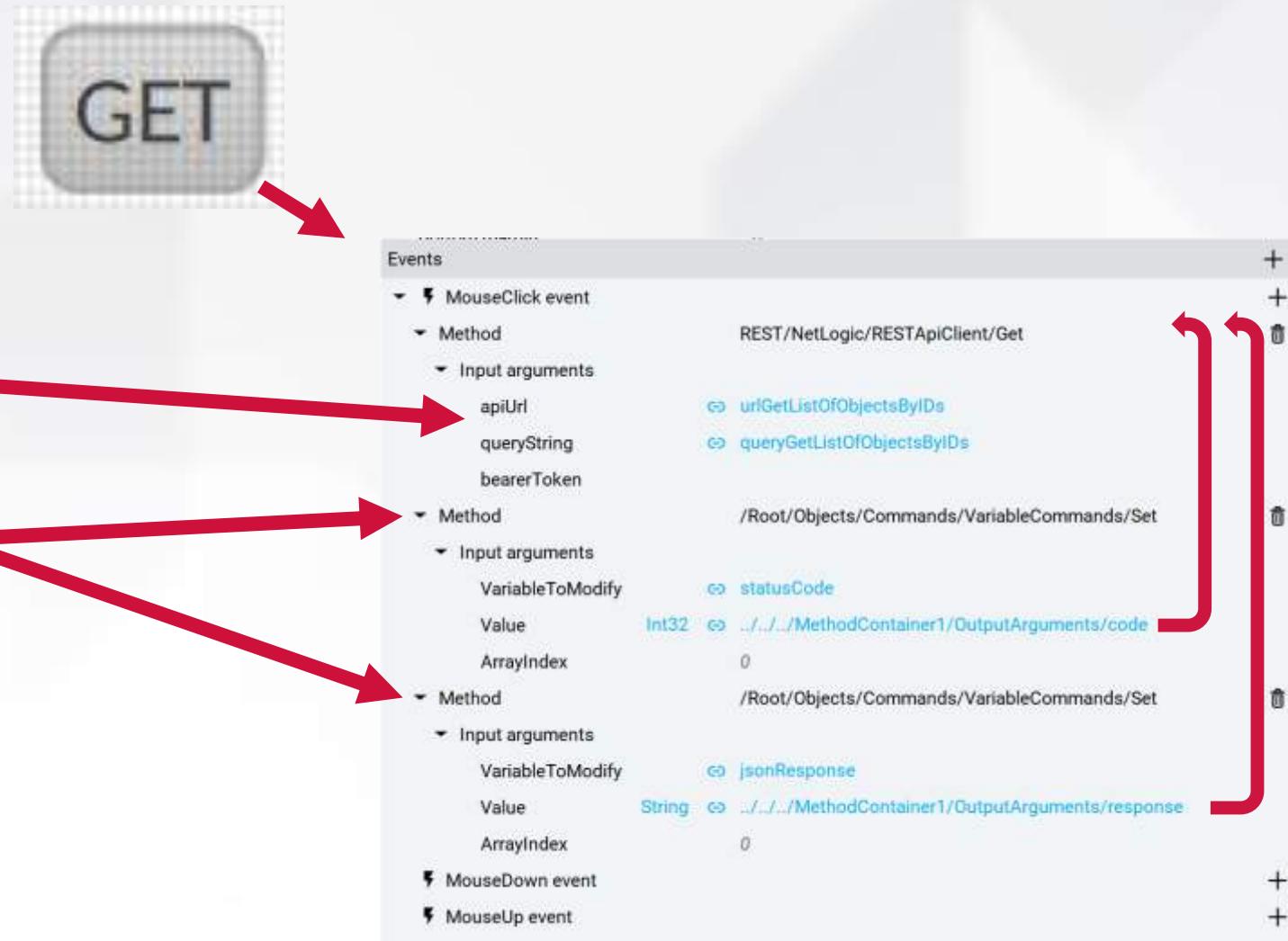


\*<https://restful-api.dev>

# REST APIs - GET Example (Get a list of specific objects)

- Get contains 3 input parameters:
  - apiURL: "https://api.restful-api.dev/objects"
  - queryString: "id=3&id=5&id=10" (everything after "?" in request URL)
  - bearerToken: NULL (No Security)

1. Call the RESTApiClient/Get method with 'apiURL' and 'queryString' set.
2. Use VariableCommands/Set to gather the RESApiClient/Get Output Parameters



# REST APIs – POST Example (Add new object)

- POST contains 4 input parameters:
  - apiURL: "https://api.restful-api.dev/objects"
  - requestBody: "{ "name": "Apple MacBook Pro 16", "data": { "year": 2019, "price": 1849.99, "CPU model": "Intel Core i9", "Hard disk size": "1 TB" } }"
  - bearerToken: NULL (No Security)
  - contentType: "application/json" (See [Media type - Wikipedia](#) for other content-type examples)

The screenshot shows a REST API documentation interface with a sidebar of methods and their descriptions:

- GET LIST OF ALL OBJECTS
- GET LIST OF OBJECTS BY IDS
- GET SINGLE OBJECT
- POST ADD OBJECT** (highlighted in blue)
- PUT UPDATE OBJECT
- PATCH PARTIALLY UPDATE OBJECT
- DELETE DELETE OBJECT

On the right, the **Request** pane shows the API endpoint `https://api.restful-api.dev/objects` and the JSON `requestBody`:

```
{  
  "name": "Apple MacBook Pro 16",  
  "data": {  
    "year": 2019,  
    "price": 1849.99,  
    "CPU model": "Intel Core i9",  
    "Hard disk size": "1 TB"  
  }  
}
```

The **Response** pane shows the JSON `responseObject` returned with status code 200:

```
{  
  "id": "7",  
  "name": "Apple MacBook Pro 16",  
  "data": {  
    "year": 2019,  
    "price": 1849.99,  
    "CPU model": "Intel Core i9",  
    "Hard disk size": "1 TB"  
  },  
  "createdAt": "2022-11-21T20:06:23.986Z"  
}
```

\*<https://restful-api.dev>

# REST APIs – POST Example (Add new object)

- POST contains 4 input parameters:
  - apiURL: "https://api.restful-api.dev/objects"
  - requestBody: "{ "name": "Apple MacBook Pro 16", "data": { "year": 2019, "price": 1849.99, "CPU model": "Intel Core i9", "Hard disk size": "1TB" } }"
  - bearerToken: NULL (No Security)
  - contentType: "application/json" (See [Media type – Wikipedia](#) for other content-type examples)

1. Call the RESTApiClient/Post method with 'apiURL', 'requestBody', and contentType set.
2. Use VariableCommands/Set to gather the RESTApiClient/Post Output Parameters



# REST APIs – PUT Example (Update object)

- PUT contains 4 input parameters:
  - apiURL: "https://api.restful-api.dev/objects/7"
  - requestBody: "{ "name": "Apple MacBook Pro 16", "data": { "year": 2019, "price": 2049.99, "CPU model": "Intel Core i9", "Hard disk size": "1 TB", "color": "silver" } }"
  - bearerToken: NULL (No Security)
  - contentType: "application/json" (See [Media type - Wikipedia](#) for other content-type examples)

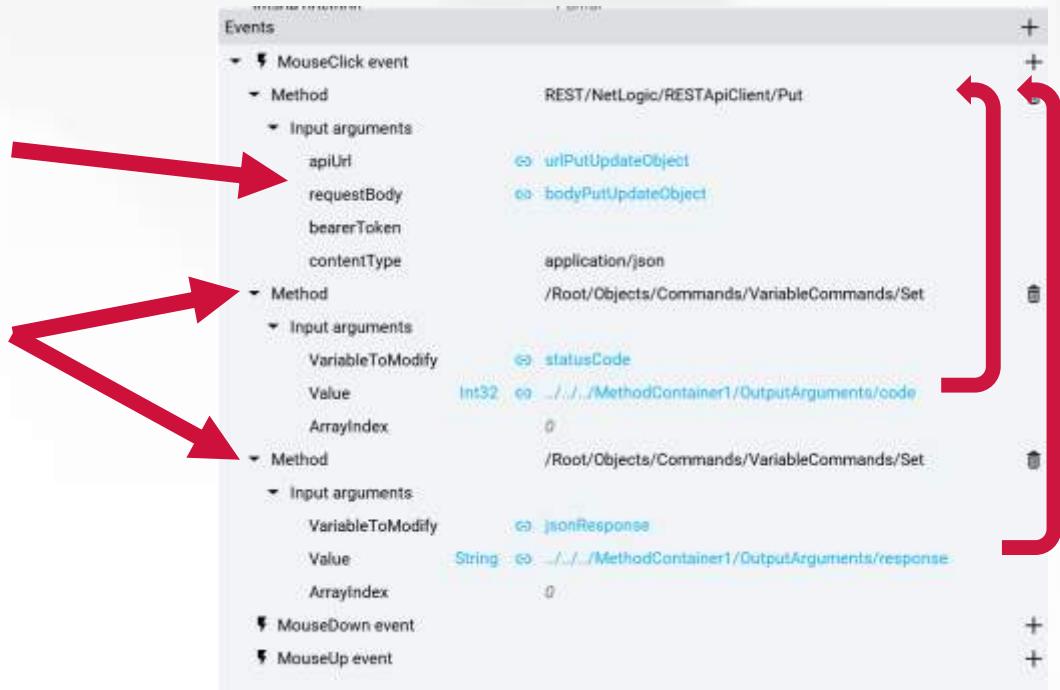


\*<https://restful-api.dev>

# REST APIs – PUT Example (Update object)

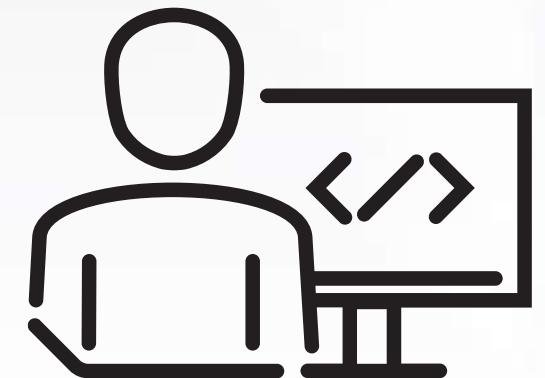
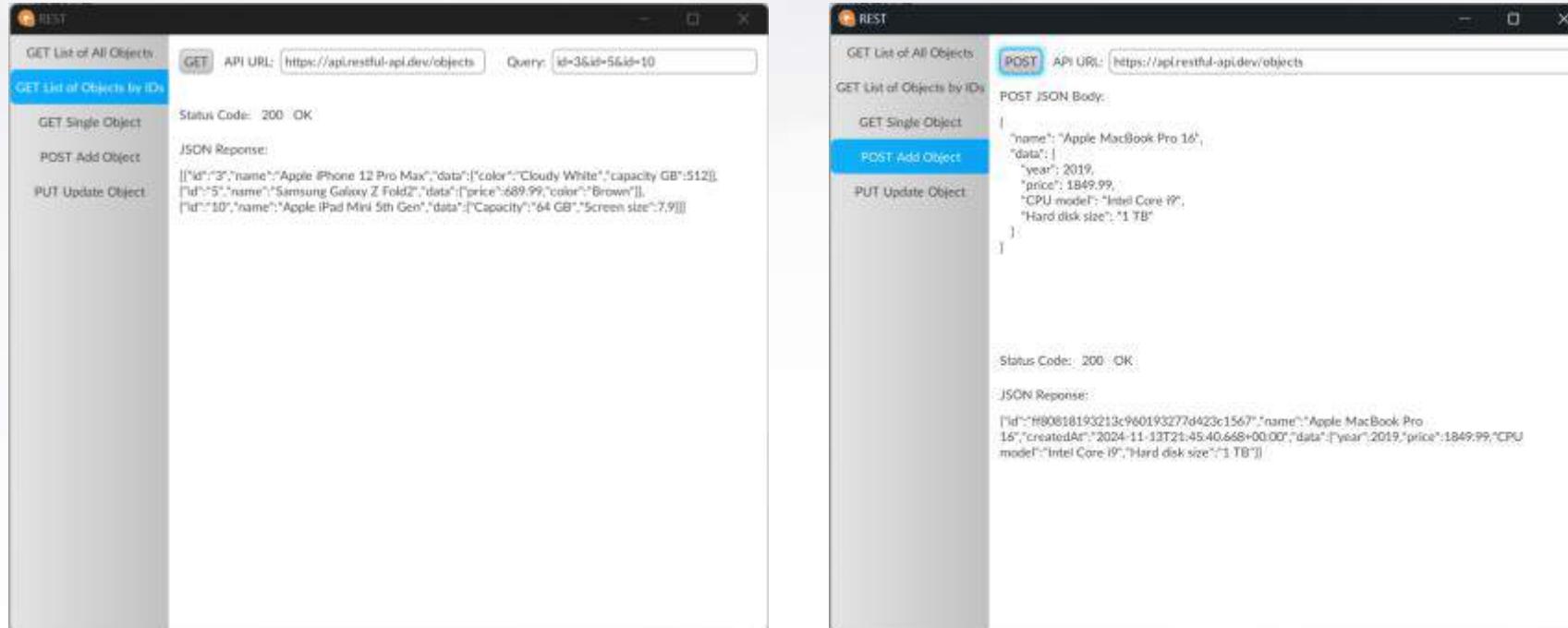
- PUT contains 4 input parameters:
  - apiURL: "https://api.restful-api.dev/objects/7"
  - requestBody: "{ "name": "Apple MacBook Pro 16", "data": { "year": 2019, "price": 2049.99, "CPU model": "Intel Core i9", "Hard disk size": "1 TB", "color": "silver" } }"
  - bearerToken: NULL (No Security)
  - contentType: "application/json" (See [Media type – Wikipedia](#) for other content-type examples)

1. Call the RESTApiClient/Put method with 'apiURL', 'requestBody', and contentType set.
2. Use VariableCommands/Set to gather the RESTApiClient/Put Output Parameters



# Hands-on session

- Open D01\_01 REST\_Start
- Add RESTAPIClient NetLogic from Library
- Create method calls on premade GET, POST, and PUT push buttons to complete their associated tasks against the RESTAPIClient NetLogic



# Appendix – API Key vs Bearer Token

- Many Public REST APIs require an API Key and or Bearer Token for access
- API Keys are used for Application Authorization
- Bearer Tokens are used for User Authorization
- More Detail here: [Why and when to use API keys | Cloud Endpoints with OpenAPI | Google Cloud](#)

# Appendix – Optix as REST Web Service

- [FactoryTalk-Optix/REST\\_WEB\\_WS\\_Server: Sample of hosting a REST, WEB and Web Socket Server in Optix](#)

## Features

- All-in-one object to host WEB Server, REST Server, and Web Sockets in FTOptix
- Currently, no TLS is supported, but maybe in the future
- Provided as sample code
- When you use a REST Server, you will need to adjust the C# code to deal with all the different resources (Endpoints) of the server.
- If used as a Web Socket, there is a variable “Message” which is updated automatically.

# Optix Runtime on Docker containers



# Optix Runtime on Docker containers

- Optix supports running on x86\_64 Ubuntu 22.04 devices
- Creating a container with Ubuntu 22.04 LTS allows the Update Server to be installed in the container
- Two ways to create containers:
  - Container with FactoryTalk Optix Update Server: the Runtime can be downloaded from the IDE to the container
  - Container with the FactoryTalk Optix Runtime: the exported Linux binaries are packed to a Docker container



# Optix Runtime on Docker containers

- Procedure overview:

1. Create base container with Ubuntu 22.04
2. Install required libraries and dependencies
3. Copy the FactoryTalk Optix Application Update Service installer
4. Install the Update Service process
5. Set the Update Service as entrypoint for the container
6. Deploy the container



Full procedure with links  
and images

# Optix Runtime on Docker containers

- Limitations:
  - Cannot use Native Presentation Engine
  - Containers can be licensed only if permanently connected to the internet
    - No offline licensing mechanism (as per V1.6)
    - License is verified to the Rockwell infrastructure every 30 mins
    - Entitlement number is passed as Environment variable to the container
  - No support for arm64 devices (Raspberry Pi, etc)
  - Update Server cannot be upgraded from the IDE
  - A new container must be rebuilt

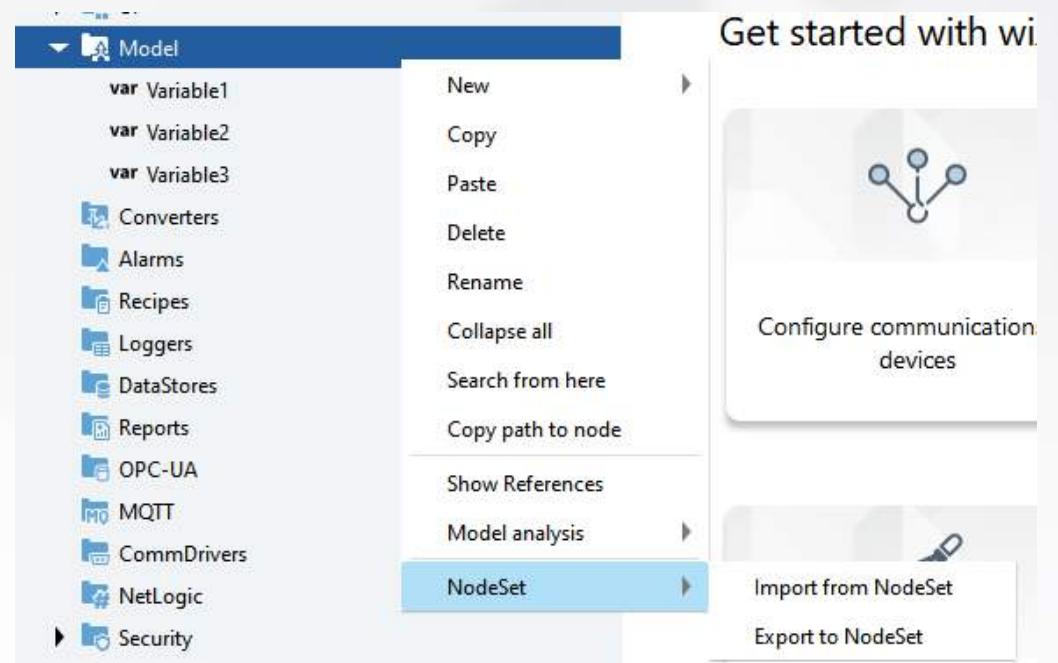


# OPC-UA NodeSet



# OPC-UA NodeSet

- A NodeSet is an XML file which describes a set of OPC-UA elements
- Can be used to:
  - Replicate a portion of the project tree
  - Import a set of tags from an OPC-UA server (offline import)
  - Perform bulk changes on project elements

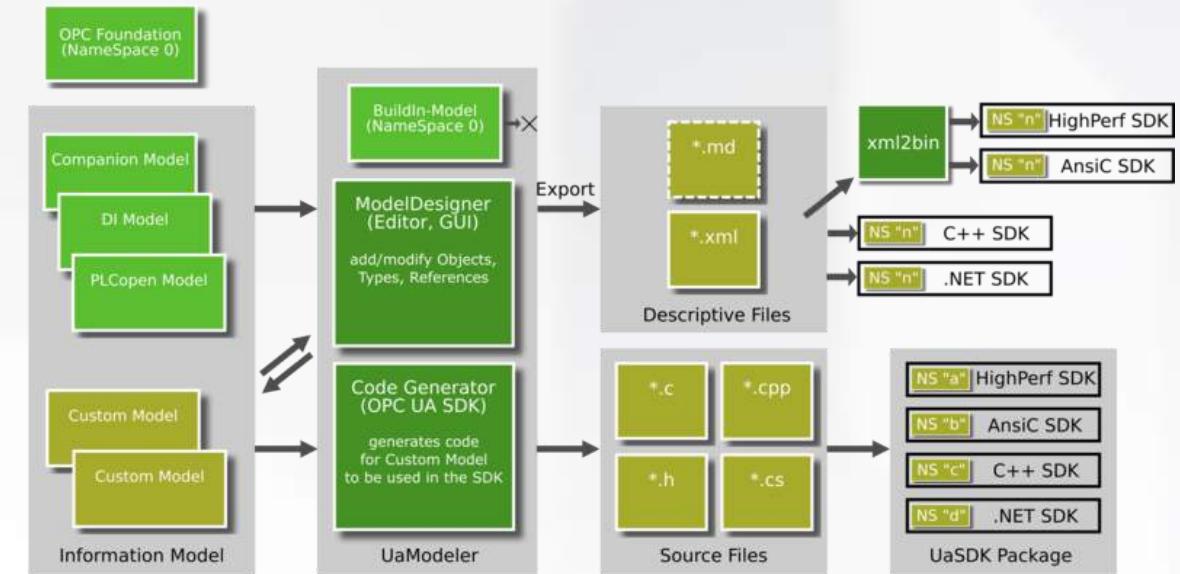


# OPC-UA NodeSet

- A NodeSet is an XML file which describes a set of OPC-UA elements
  - Can be used to:
    - Replicate a portion of the project tree
    - Import a set of tags from an OPC-UA server (offline import)
    - Perform bulk changes on project elements

# OPC-UA NodeSet

- NodeSet can be created with a variety of tools like UaModeler from the OPC Foundation
- Most third-parties software can generate NodeSet files

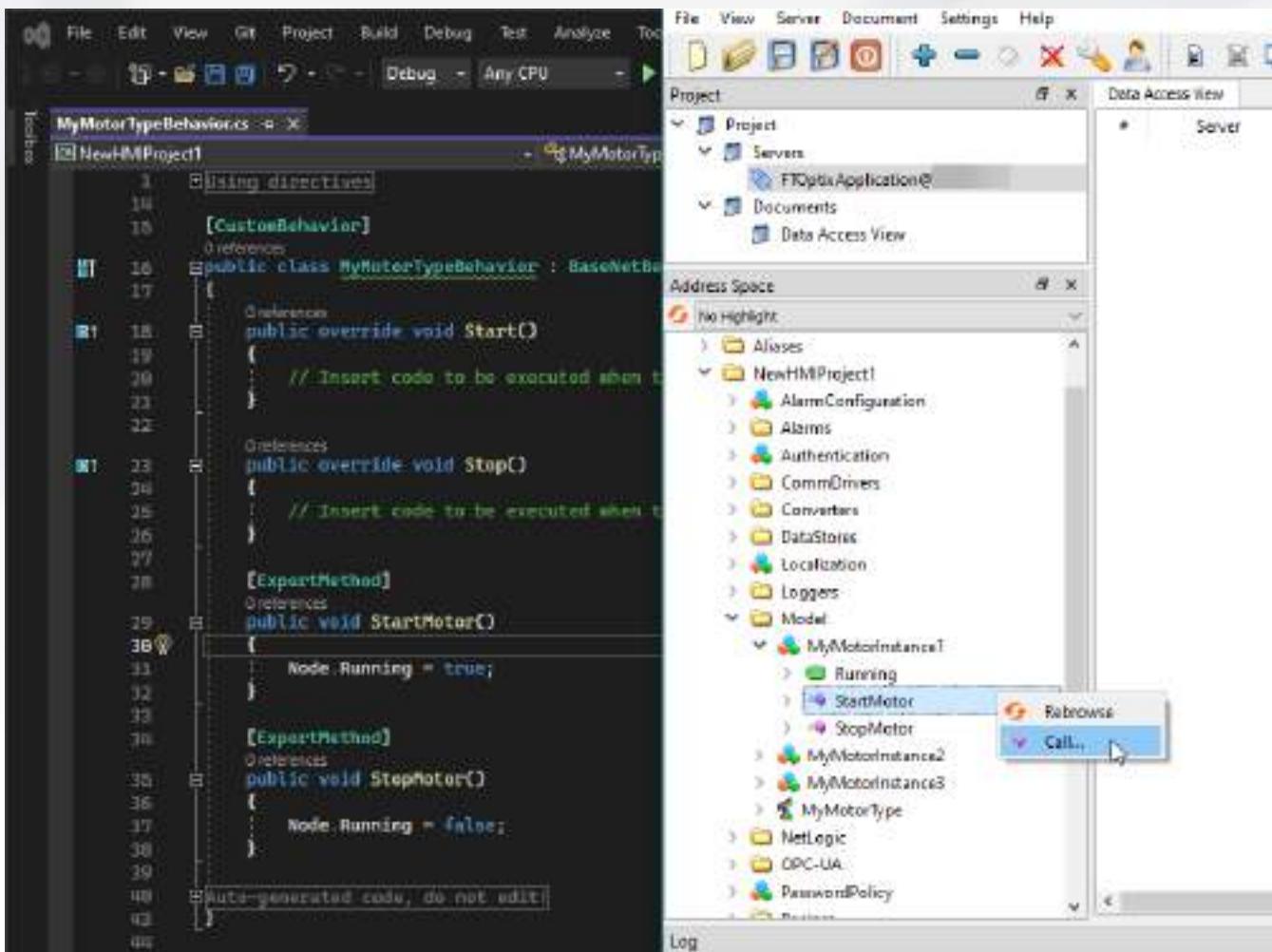


# OPC-UA Custom Behavior



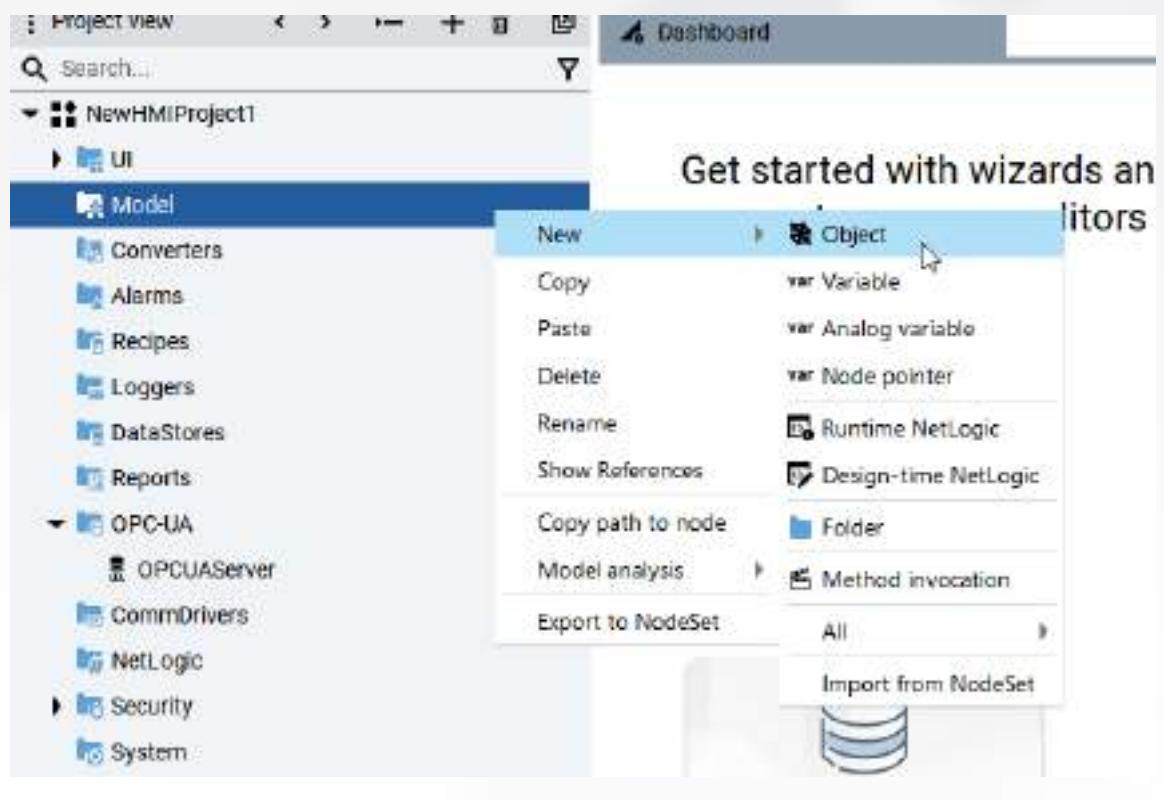
# OPC-UA Custom Behavior

- When creating custom OPC-UA Object Types in FactoryTalk Optix, some methods can be exposed via OPC-UA to be called from a client
- Custom behaviors can only be added to IUAObject types
- Instances will inherit all the exposed methods
- Methods must return void, output shall be passed via the out modifier of method's arguments
- Properties of the IUAObject can be accessed using the «node» syntax



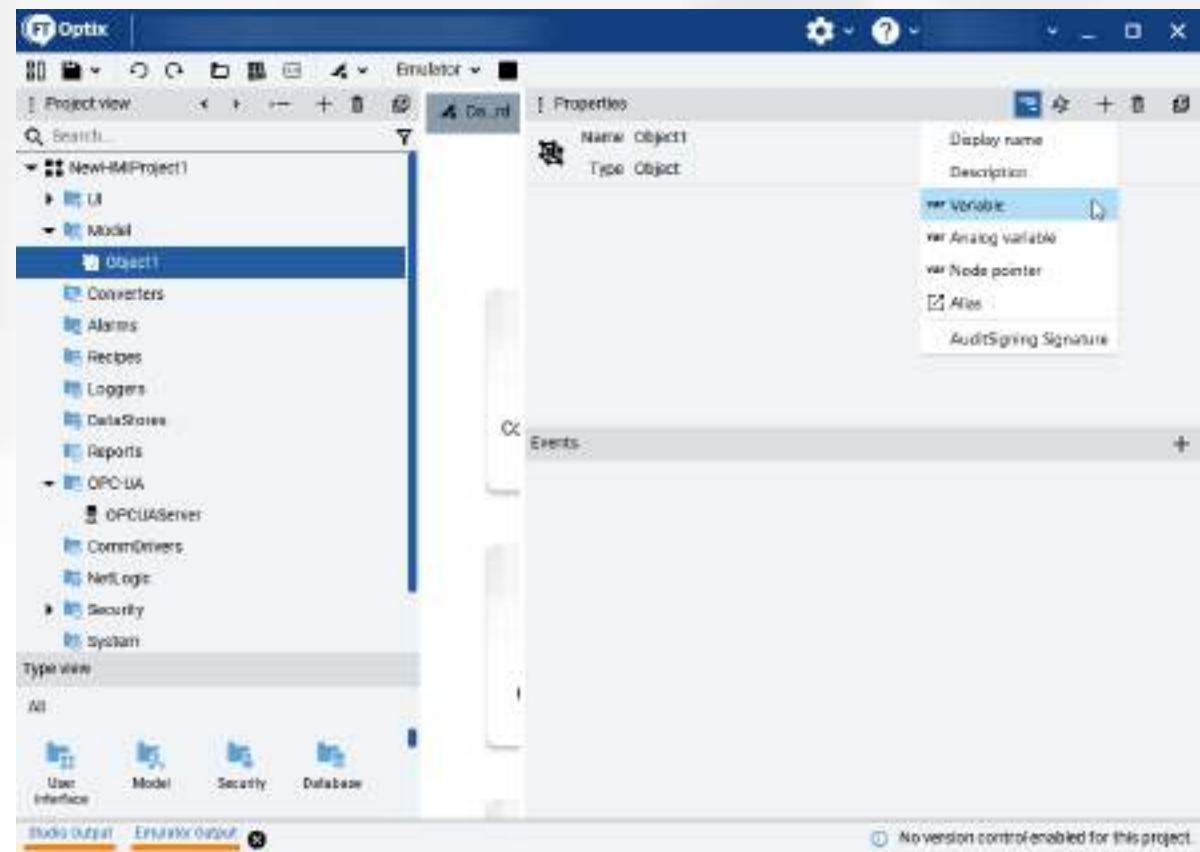
# Creating an IUAObject with a custom behavior

- Create new IUAObject
- Add properties to the object
- Convert object to object type
- Right click the object type to add the custom behavior
- Double click the object type to open the code editor and configure the methods



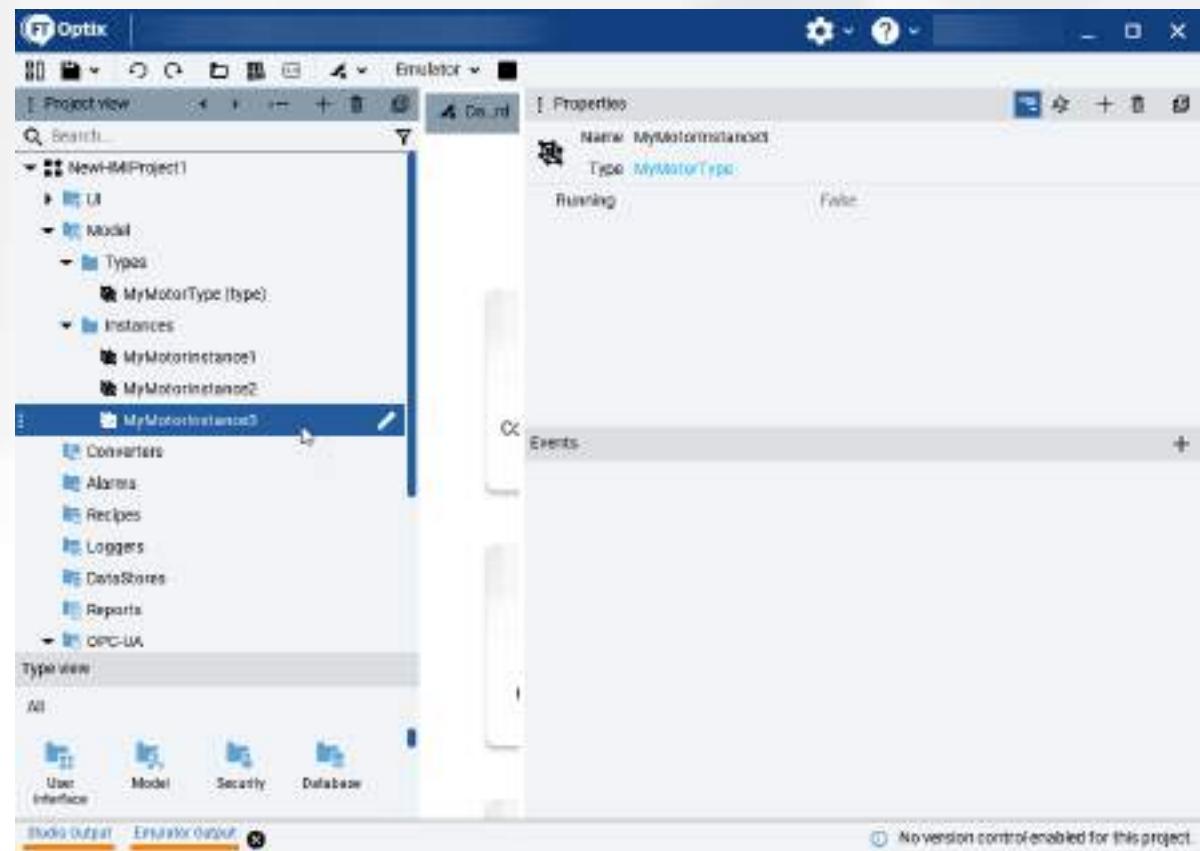
# Creating an IUAObject with a custom behavior

- Create new IUAObject
- Add properties to the object
- Convert object to object type
- Right click the object type to add the custom behavior
- Double click the object type to open the code editor and configure the methods



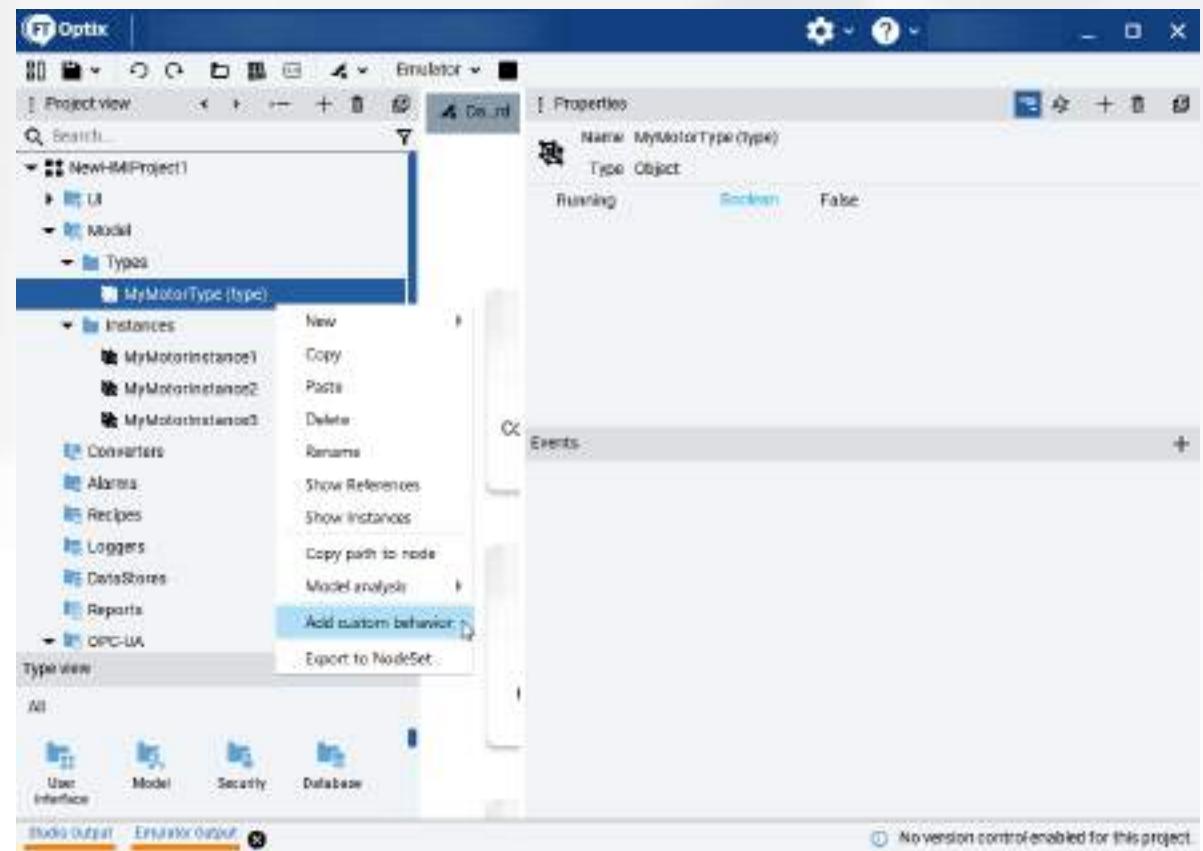
# Creating an IUAObject with a custom behavior

- Create new IUAObject
- Add properties to the object
- Convert object to object type
- Right click the object type to add the custom behavior
- Double click the object type to open the code editor and configure the methods



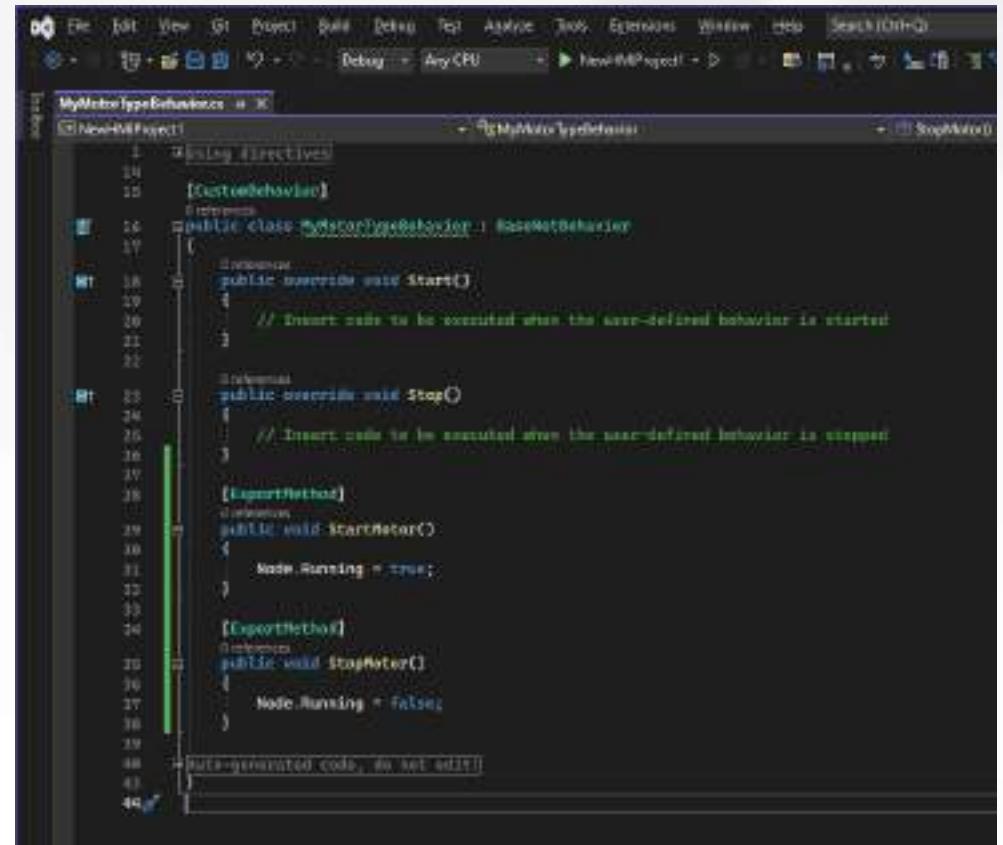
# Creating an IUAObject with a custom behavior

- Create new IUAObject
- Add properties to the object
- Convert object to object type
- Right click the object type to add the custom behavior
- Double click the object type to open the code editor and configure the methods



# Creating an IUAObject with a custom behavior

- Create new IUAObject
- Add properties to the object
- Convert object to object type
- Right click the object type to add the custom behavior
- Double click the object type to open the code editor and configure the methods

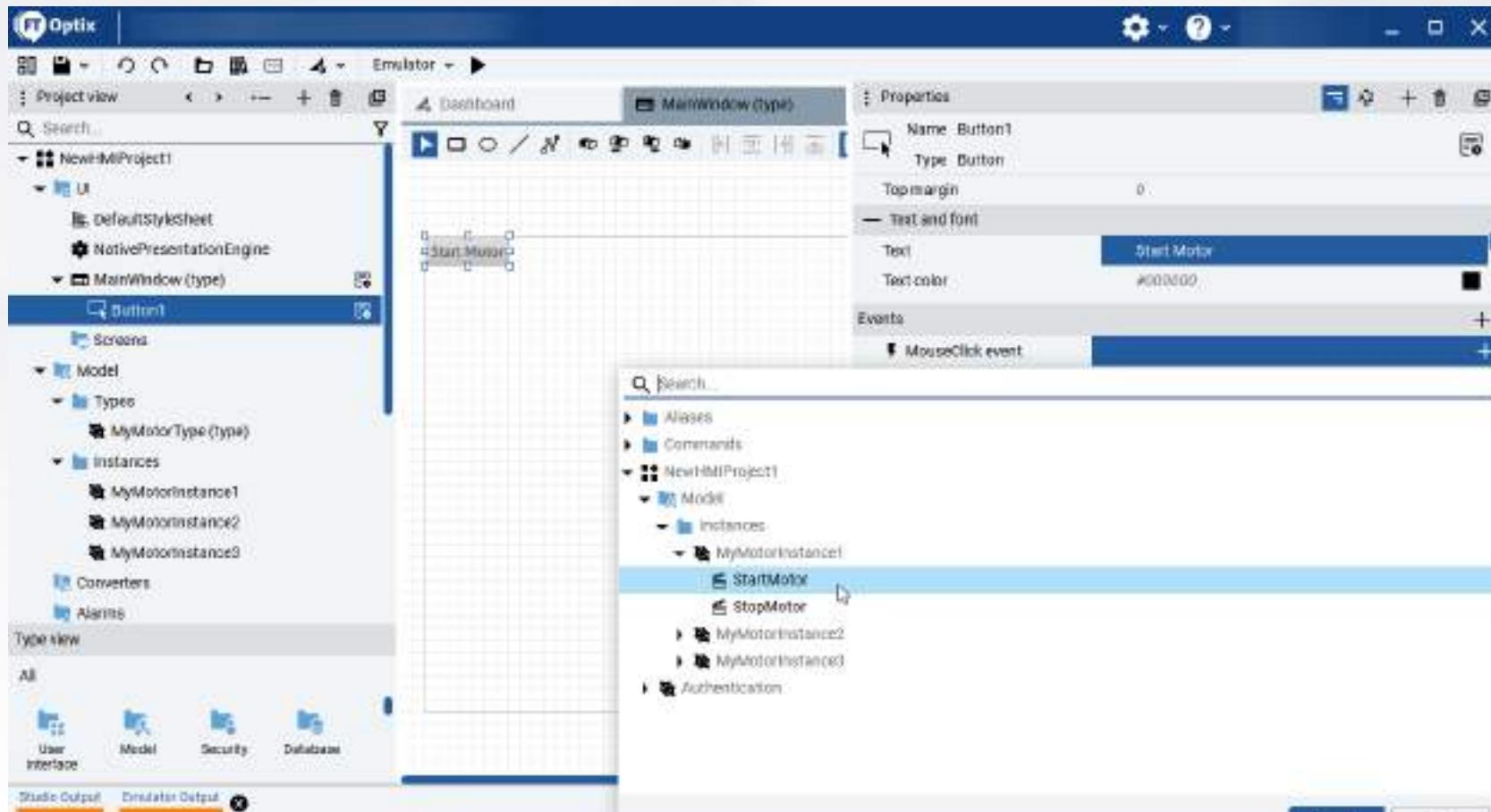


The screenshot shows a Microsoft Visual Studio code editor window with the following code:

```
MyMotorTypeBehavior.cs 30
NewMILProject1 -> MyMotorTypeBehavior.cs + StopMotor()
File Edit View Go Project Build Debug Test Analyze Tools Exercises Window Help Search (F5)
MyMotorTypeBehavior.cs 30
1  using directives;
2
3  [CustomBehavior]
4  class MyMotorTypeBehavior : BaseMotorBehavior
5
6  {
7      public override void Start()
8      {
9          // Insert code to be executed when the user-defined behavior is started
10     }
11
12     public override void Stop()
13     {
14         // Insert code to be executed when the user-defined behavior is stopped
15     }
16
17     [ExportMethod]
18     public void startMotor()
19     {
20         Node.Running = true;
21     }
22
23     [ExportMethod]
24     public void stopMotor()
25     {
26         Node.Running = false;
27     }
28
29     // auto-generated code, do not edit!
30 }
```

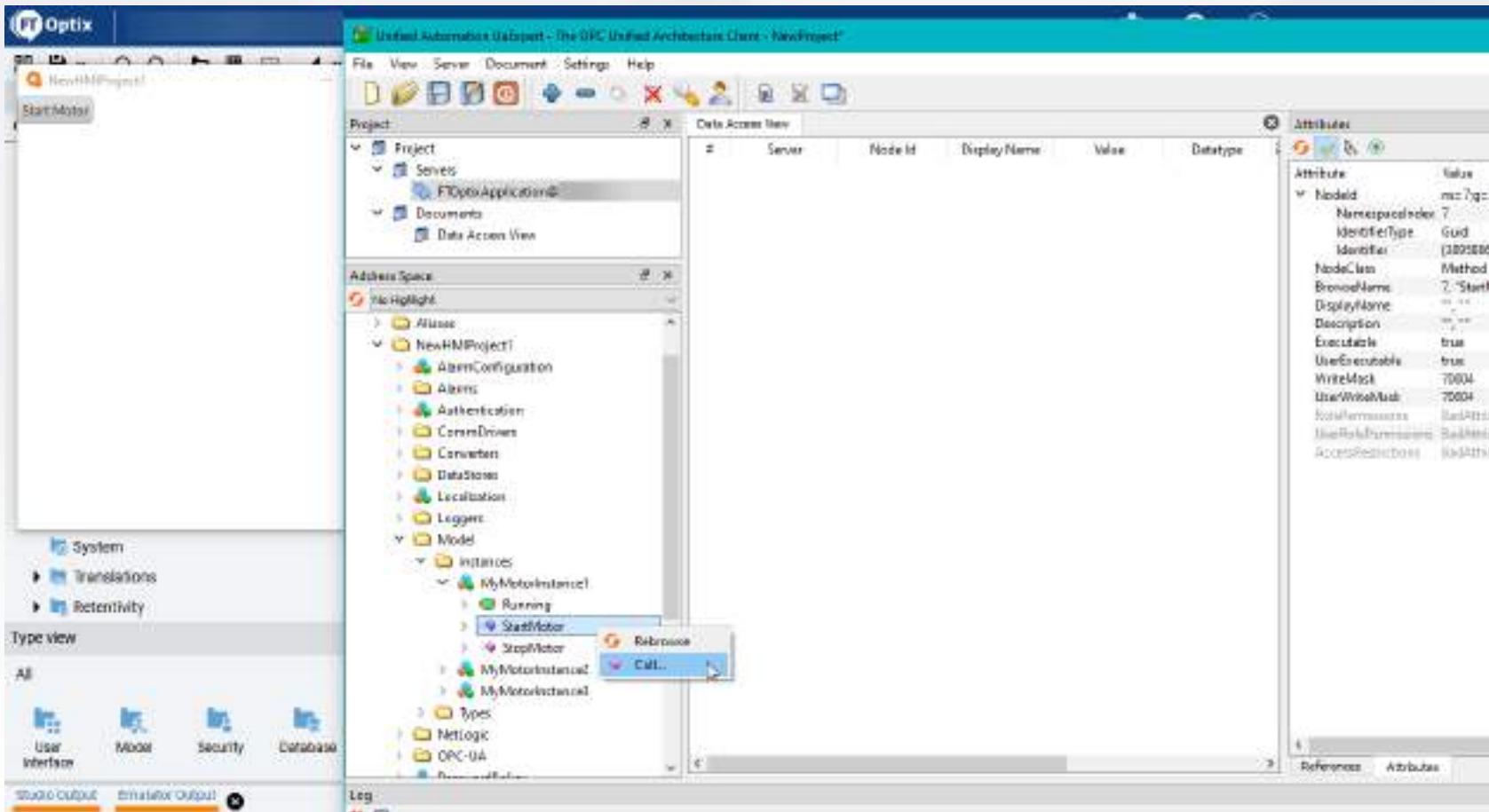
# Usage of a custom behavior

- Custom behaviors can be used to call methods via OPC-UA
- Custom behaviors can be used to call methods from FactoryTalk Optix UI Controls or events



# Usage of a custom behavior

- Custom behaviors can be used to call methods via OPC-UA
- Custom behaviors can be used to call methods from FactoryTalk Optix UI Controls or events

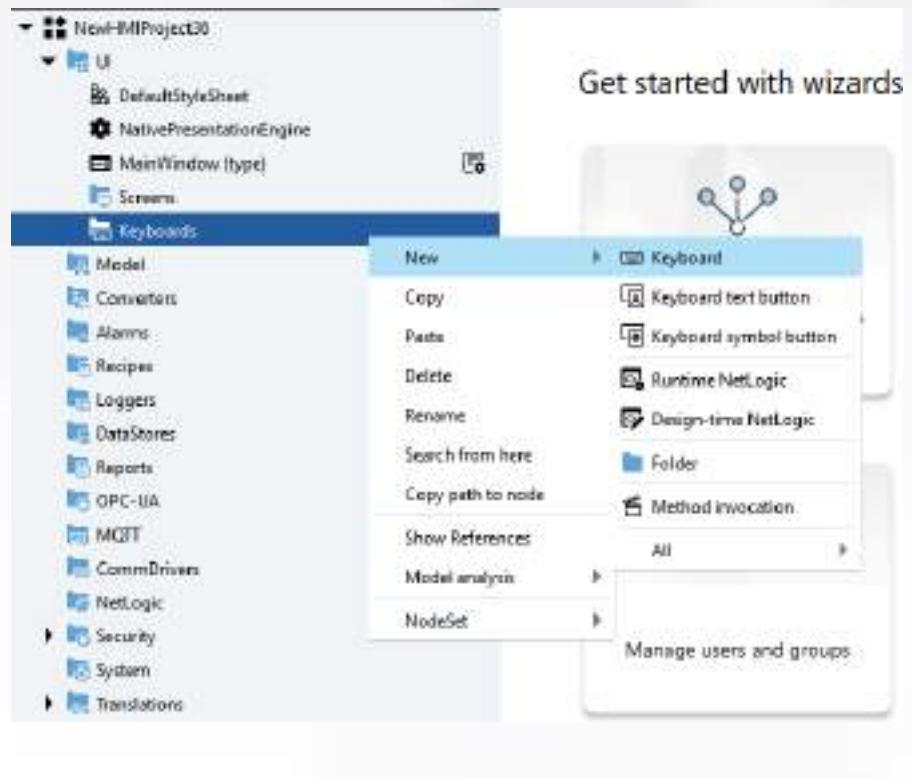


# Custom keyboards



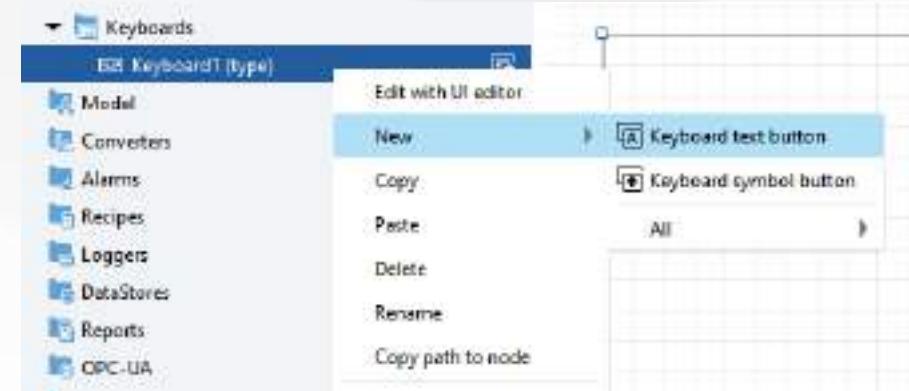
# Custom keyboards

- Allows overriding the default Qt keyboard
- Allows creating custom logics before data is passed to the field
- Works both on Native UI and Web UI
- Can be fully customized
  - Custom layouts
  - Custom keys for characters
  - Custom keys for actions or symbols



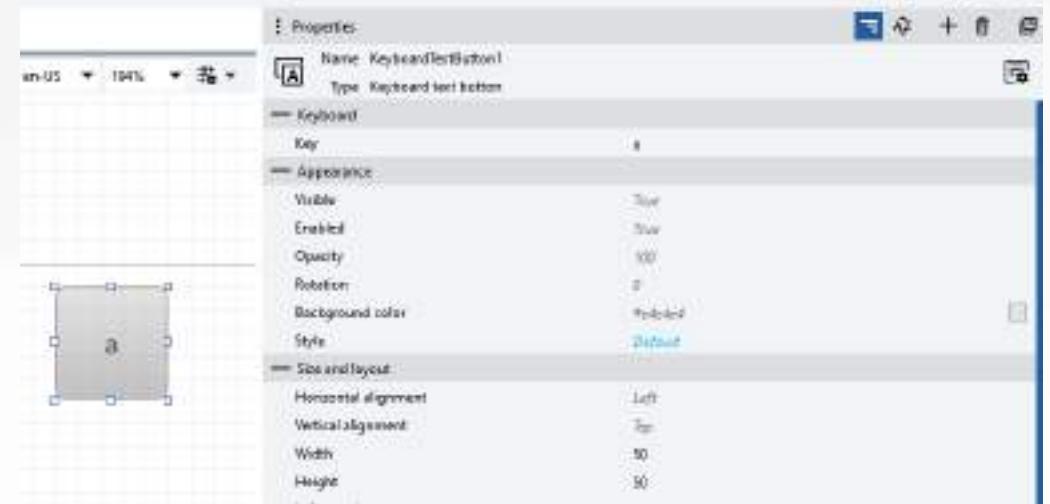
# Custom keyboards

- Allows overriding the default Qt keyboard
- Allows creating custom logics before data is passed to the field
- Works both on Native UI and Web UI
- Can be fully customized
  - Custom layouts
  - Custom keys for characters
  - Custom keys for actions or symbols



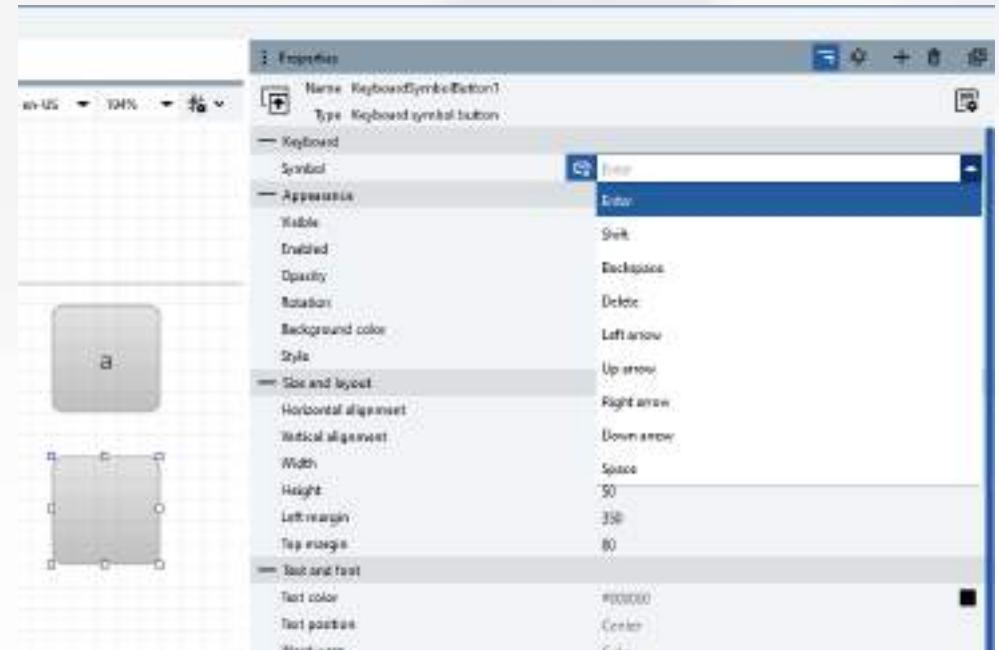
# Custom keyboards

- Allows overriding the default Qt keyboard
- Allows creating custom logics before data is passed to the field
- Works both on Native UI and Web UI
- Can be fully customized
  - Custom layouts
  - Custom keys for characters
  - Custom keys for actions or symbols



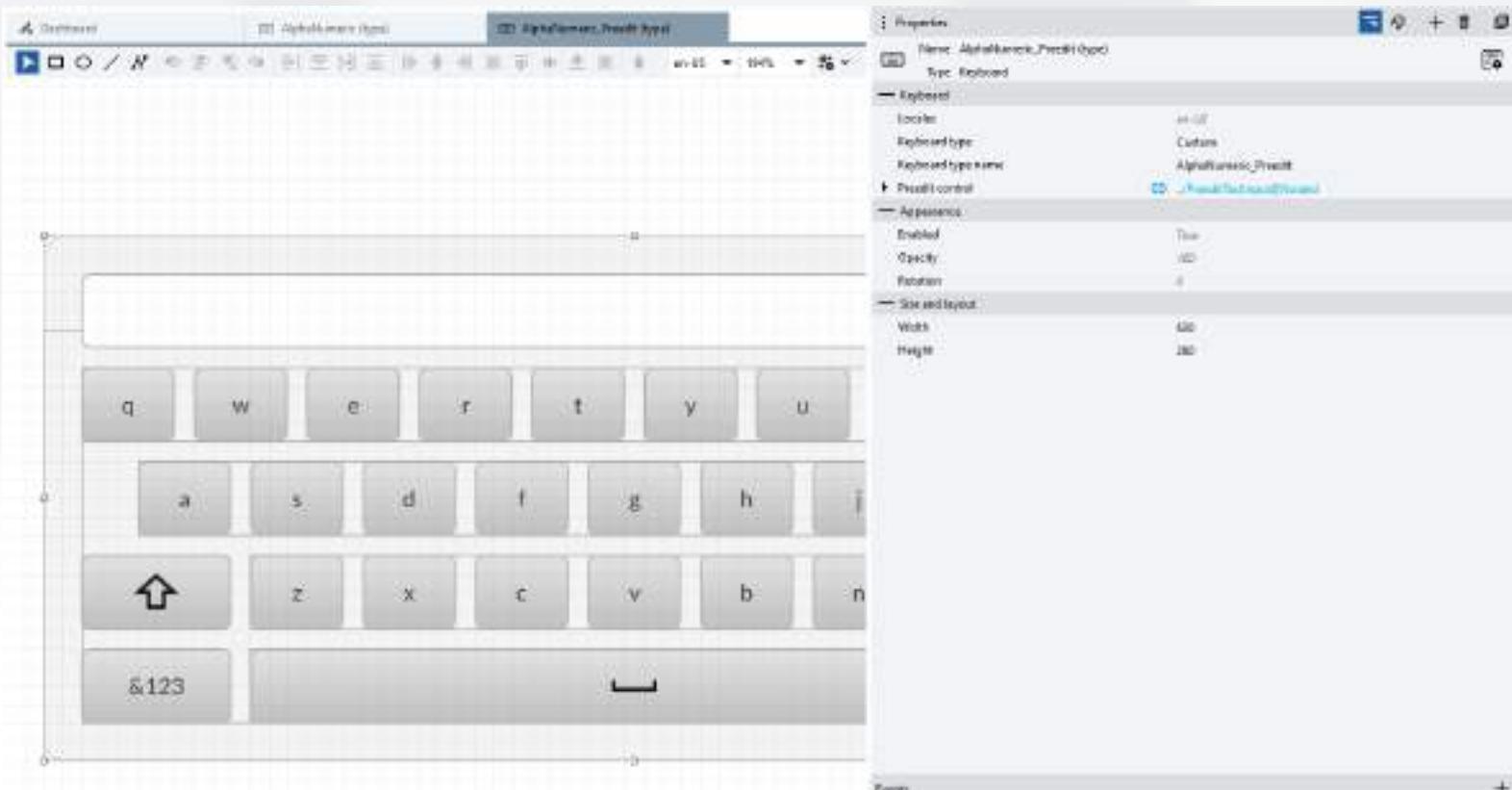
# Custom keyboards

- Allows overriding the default Qt keyboard
- Allows creating custom logics before data is passed to the field
- Works both on Native UI and Web UI
- Can be fully customized
  - Custom layouts
  - Custom keys for characters
  - Custom keys for actions or symbols



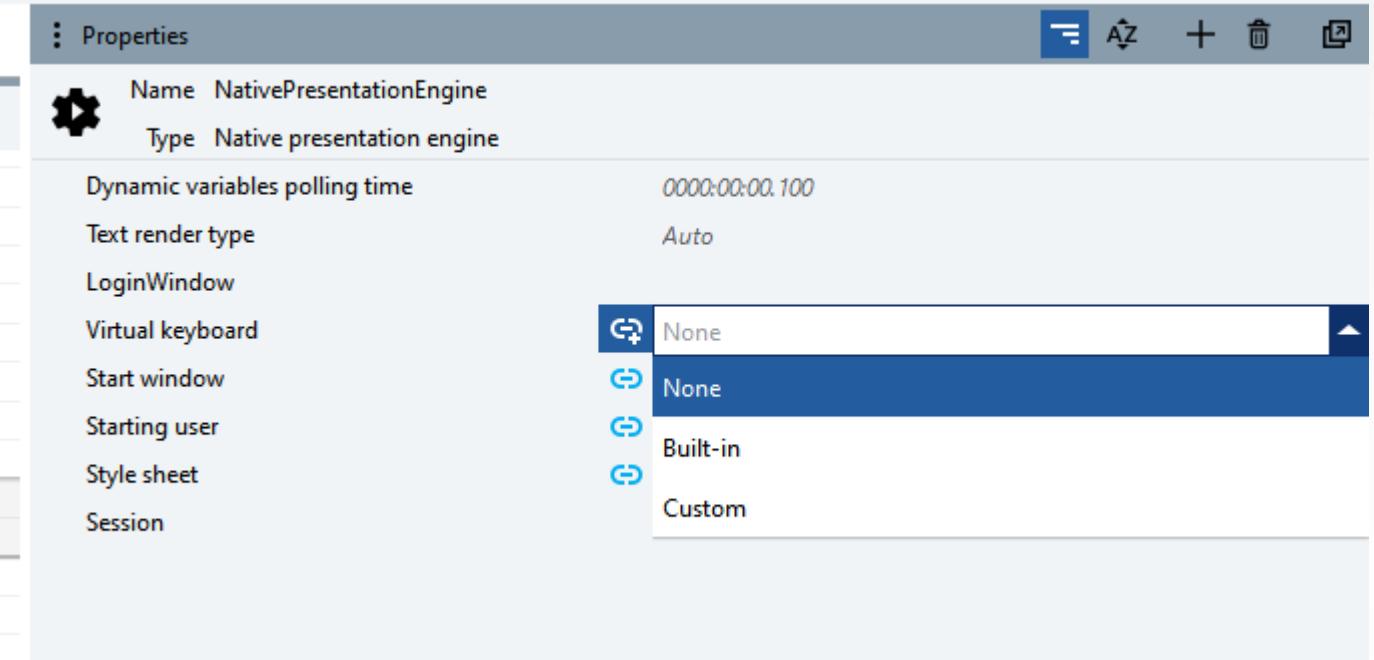
# Custom keyboards

- A control (TextBox) can be added as preedit interface
  - Useful to double check the value before sending it to the field
  - Useful to create custom logic to the value (length check, RegEx, etc)



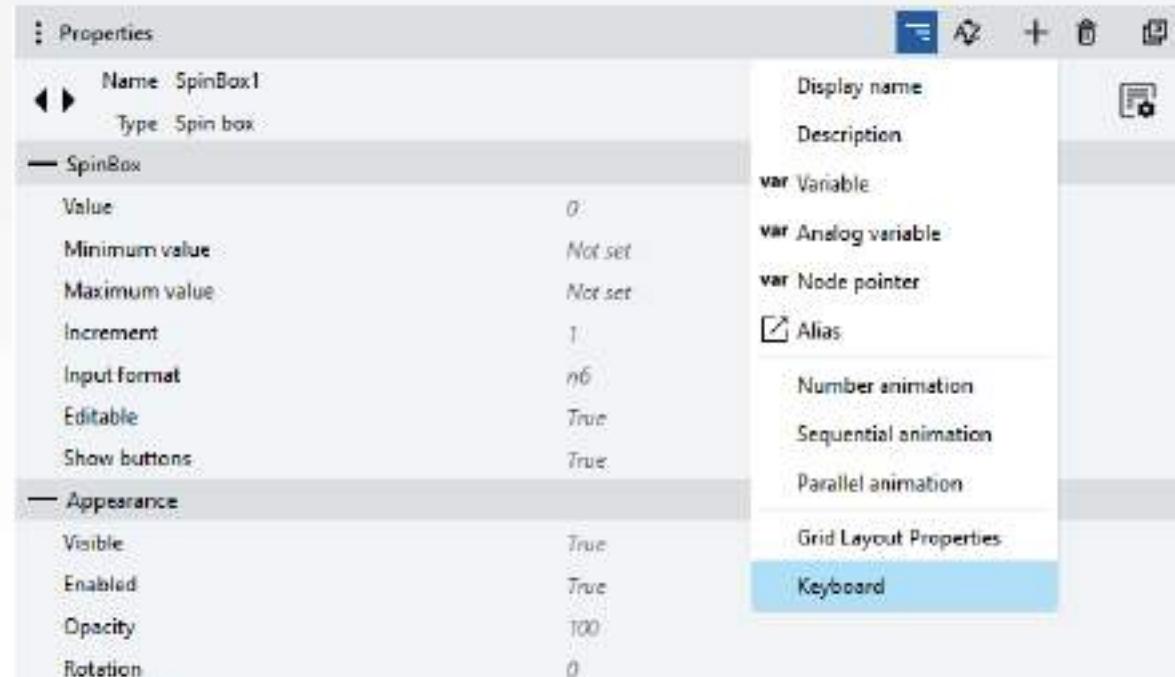
# Custom keyboards

- Keyboard selection is done at the Presentation Engine level
  - Built-in: default Qt keyboard
  - Custom: the custom keyboard for that control type



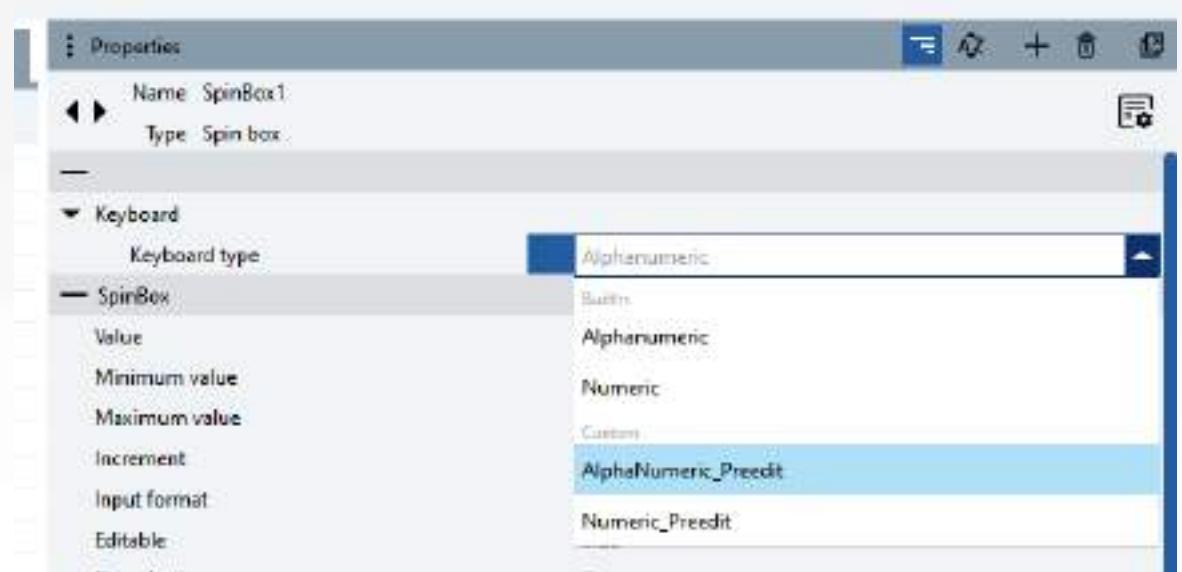
# Custom keyboards

- Different UI controls can trigger different keyboards
  - Need to add the attached property “Keyboard” to the control
  - Configure the specific keyboard to be invoked
  - If no custom attached property is added, the default custom keyboard for that kind of control (numeric or alphanumeric) is triggered



# Custom keyboards

- Different UI controls can trigger different keyboards
  - Need to add the attached property “Keyboard” to the control
  - Configure the specific keyboard to be invoked
  - If no custom attached property is added, the default custom keyboard for that kind of control (numeric or alphanumeric) is triggered

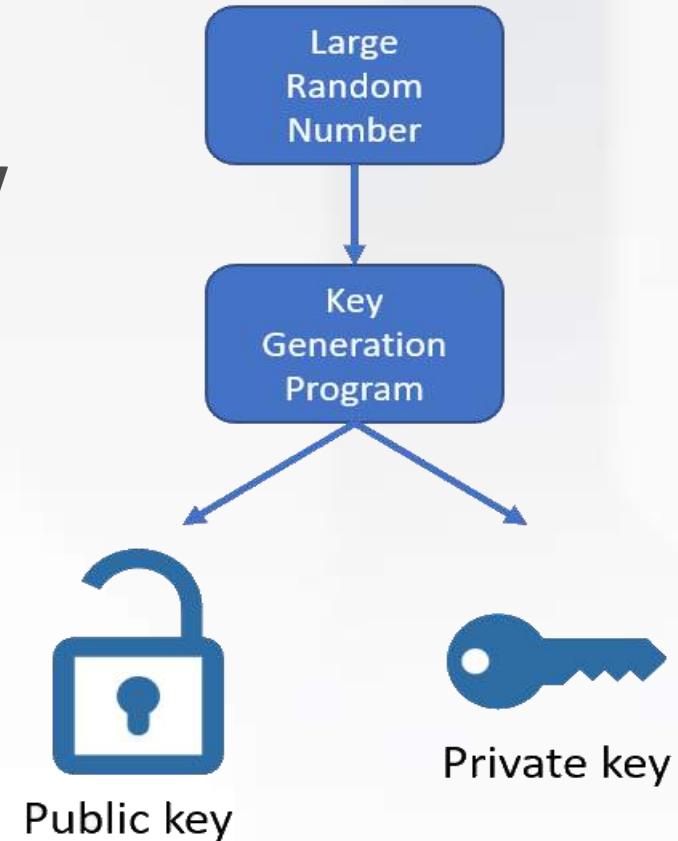


# Appendix: see the light about OPC-UA Security



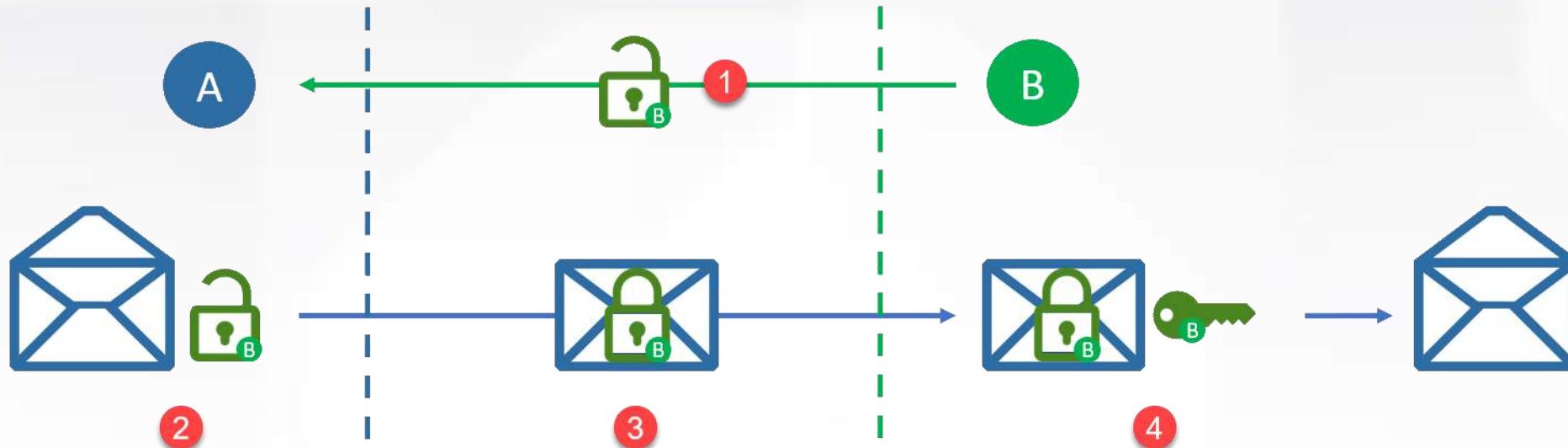
# Asymmetric cryptography

- is a cryptographic system that **uses pairs of keys**
- each pair consists of a **public key** and a **private key**
- the generation of such key pairs depends on cryptographic algorithms



# How does encryption work ?

1. Alice asks and receives Bob's padlock (Public Key exchange)
2. Alice uses the padlock to close the letter
3. Alice sends the locked letter to Bob
4. When Bob receives the letter, he can open it with his key (Private Key) of which he is the only owner.



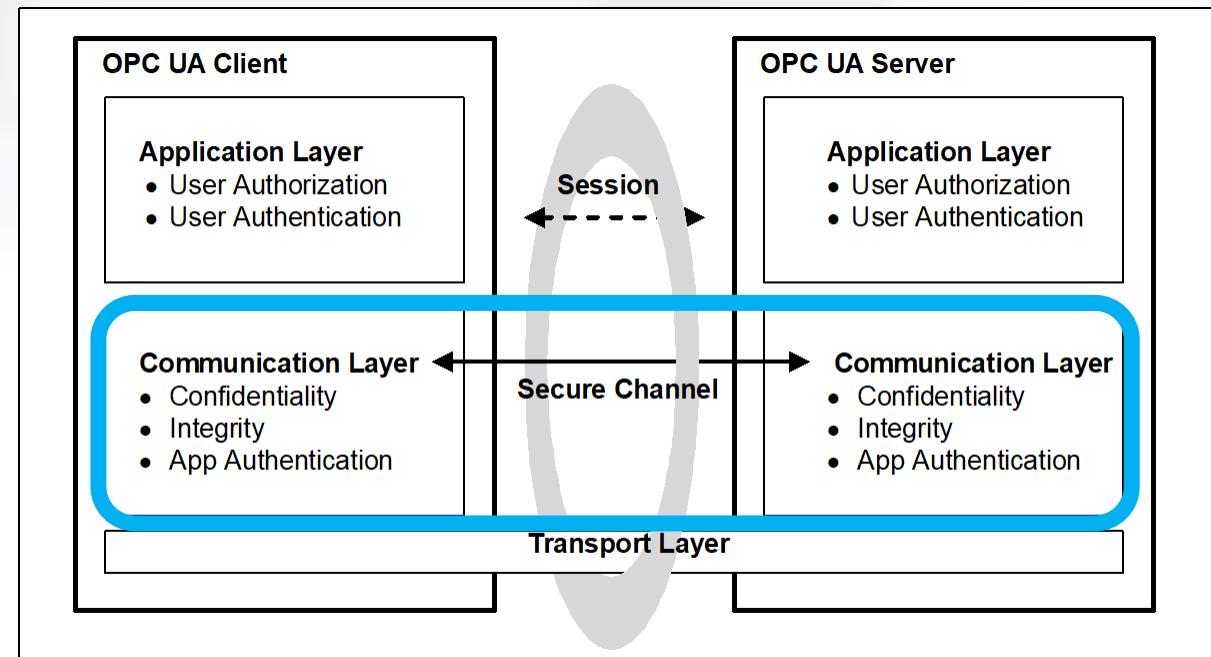
# How does encryption apply to OPC-UA?

- When the OPC-UA Client connects to the OPC-UA Server:

- X.509 Certificates are exchanged so a **Secure Channel** is established

Certificates used here are called  
*"Application Instance Certificates"*

These are the certificates exchanged when establishing a secure communication between OPC-UA Client and Server



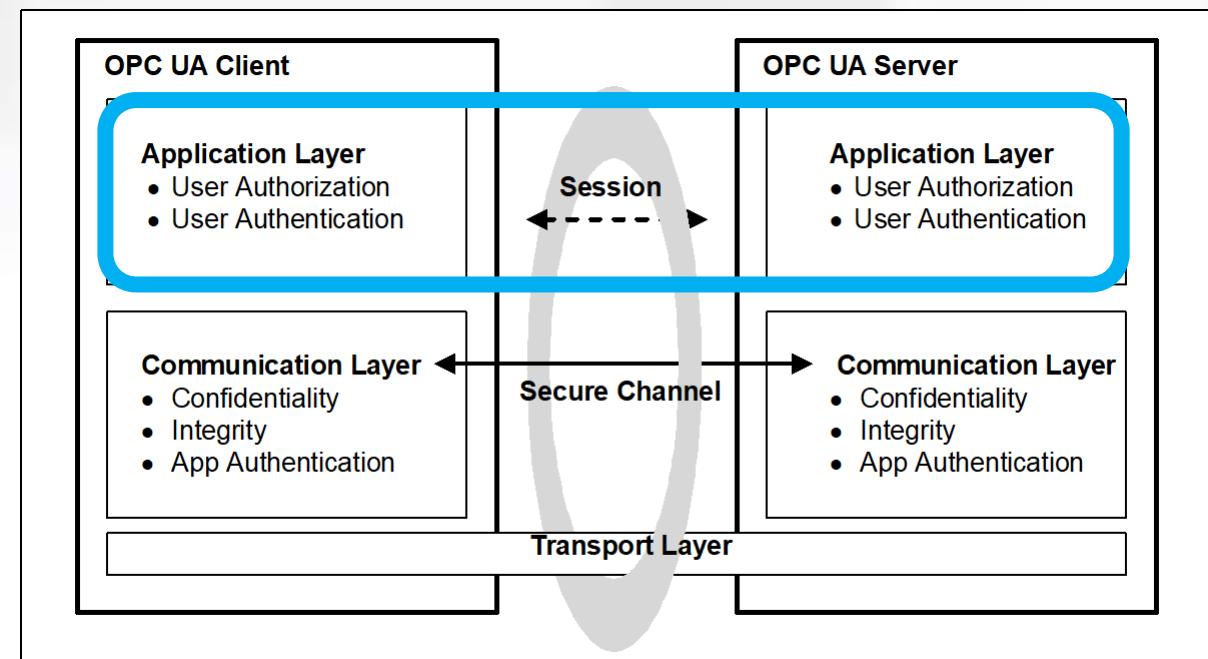
# How does encryption apply to OPC UA?

- Over the Secure Channel, a "**Session**" at "Application Layer" is defined.

Here is where the OPC-UA Client application and OPC-UA Server application transmit information, settings, and commands.

at Application Layer  
there is the User Authentication  
and User Authorization

- *anonymous*,
- *username/password*
- *X.509 Certificate*  
*(User Instance Certificate)*



# Secure channel security

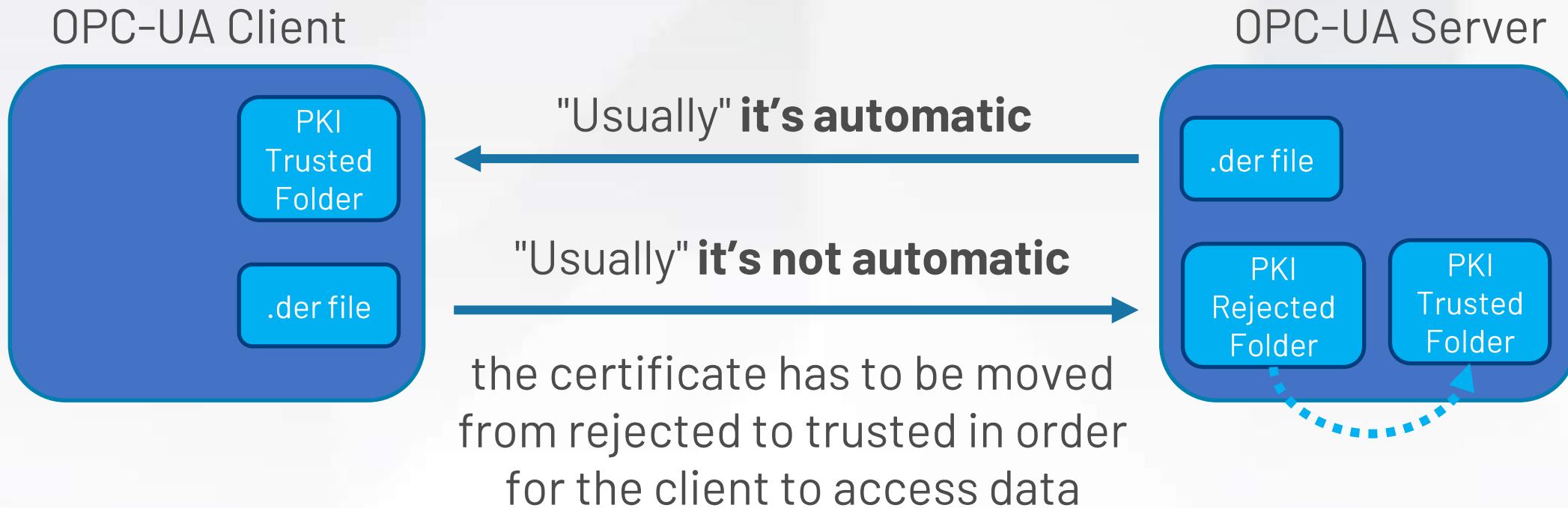
- Secure Channel security = Security Profile + Security Policy

Security Profiles	Description	Security Policies (length of the key and the algo)
None	No security	None
Sign	The sender uses <u>his private key</u> to digitally sign the data. The receiver verifies whether the data comes from the expected sender using the <u>sender's public key</u>	Basic256Sha256 Aes128-Sha256-RsaOaep Aes256-Sha256-RsaPss
Sign+Encrypt	In addition to Sign, the sender encrypt data using <u>receiver's public key</u>	

# PKI and X.509 Certificates

- Public Key Infrastructure (PKI)
  - is a system for the creation, storage, and distribution of digital certificates
- X.509
  - is a standard defining the format of public key certificates
- In OPC-UA:
  - ".der" is the certificate file containing **public key**,
  - ".pem" is the **private key** file

# OPC-UA certificate exchange



# Generating OPC-UA certificates



# Generating OPC-UA certificates

- Certificates are generated by authorities
  - Authorities are special entities, trusted by a number of users/clients that can **validate a certificate** and inform all users/clients that a certificate is valid and can be trusted
- Authorities (CA) can be public or local
  - Public authorities are well-known and trusted organizations that can sign and validate certificates (almost) worldwide. Examples include **DigiCert**, **Let's Encrypt**, **Comodo** and many more.
  - Private authorities are trusted services that can sign and validate certificates in a local (confined) environment. An example of a private authority can be the IT department of a company
- All certificates must be signed by an authority
  - After a certificate is generated, **an authority must sign it**
  - The certificate authority provides to all users/clients a notice with the list of valid and signed certificates
  - After a certificate is signed by an authority, all users/clients performing the handshake with the public key and the thumbprint of a signed certificate can be sure that nothing was tampered, and the connection can be established.
  - If the certificate is signed by an untrusted CA, it is rejected by default and the user is warned about potential risks.

# Creating a self-signed certificate from FactoryTalk Optix Studio

- From the dropdown of the **Settings** menu, select **Create certificate**
  - Fill-in all the details in the certificate creation form
  - Select the certificate name
  - Select the output folder
  - Click **Create** to generate the certificates
- 
- **Certificates created with FactoryTalk Optix Studio are self-signed and must be trusted manually by the clients**



# Creating a self-signed certificate from FactoryTalk Optix Studio

- From the dropdown of the **Settings** menu, select **Create certificate**
- Fill-in all the details in the certificate creation form
- Select the certificate name
- Select the output folder
- Click **Create** to generate the certificates

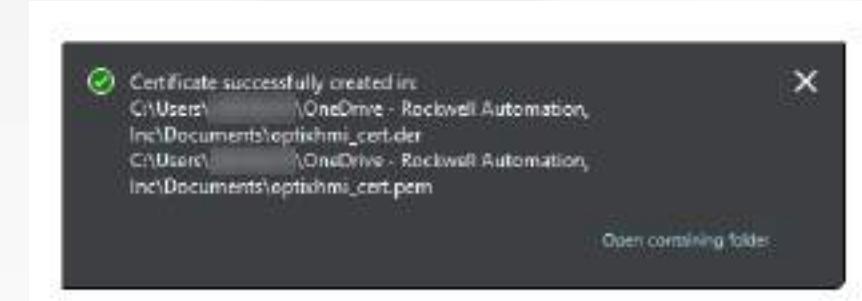
**Create certificate**

Subject		OPC UA information	
Common name*	FTOptixApplication	Application URL*	urn:FactoryTalkOptixHTML:FTOptix.Application
Organization*	Rockwell Automation	Domain name*	ftsoptixruntime.local localhost
Organization unit*	AGEM s.r.l. R&D team	IP address*	
Locality*	Astegna		
State*	Udine		
Country*	IT		
Certificate settings			
RSA key strength*	4096 bits	Signature algorithm*	SHA-512
		Expiration date*	
		05/01/2018 00:00:00	
Name	optikhtml_cert		
Location	C:\Users\ /OneDrive - Rockwell Automation, Inc\Documents		
*Required field			

**Cancel** **Create**

# Creating a self-signed certificate from FactoryTalk Optix Studio

- From the dropdown of the **Settings** menu, select **Create certificate**
- Fill-in all the details in the certificate creation form
- Select the certificate name
- Select the output folder
- Click **Create** to generate the certificates



# How to fill the self-signed certificate generation fields

Screenshot of the "Create certificate" dialog box. The "Subject" section contains fields for Common name, Organization, Organization unit, Locality, State, and Country. The "OPC UA information" section includes Application URI, Domain names, and IP addresses. The "Certificate settings" section allows setting RSA key strength (4096 bits), Signature algorithm (SHA-512), and Issuance date (05/22/2024 00:00:00). Below these are fields for Name (optikhmi\_cert) and Location (C:/Users/...). At the bottom, there are "Cancel" and "Create" buttons.

The "Common name" field is highlighted with a red arrow pointing to it. The "FTOptixApplication" value is selected in the dropdown menu.

**Common name:** name of the OPC-UA application. By default, in FactoryTalk Optix, this is always "FTOptixApplication"

# How to fill the self-signed certificate generation fields

Screenshot of the "Create certificate" dialog box. The "Organization" field is highlighted with a red arrow pointing to it. The "Organization" field contains "Rockwell Automation". The "Domain names" dropdown menu shows "ftoptikmni.local" and "localhost", with "ftoptikmni.local" selected.

**Create certificate**

**Subject**

- Common name\*: FTOptixApplication
- Organization\*: Rockwell Automation
- Organization unit\*: ASEM srl R&D team
- Locality\*: Artegno
- State\*: Udine
- Country\*: IT

**OPC UA information**

- Application URI\*: urn: FactoryTalkOptixHMI-FTO-application
- Domain names: +  
ftoptikmni.local  
localhost
- IP addresses: +

**Certificate settings**

- RSA key strength: 4096 bits
- Signature algorithm: SHA-512
- Expiration date: 05/22/2025 00:00:00

**Name:** optikmni\_cert

**Location:** C:/Users/ /OneDrive - Rockwell Automation, Inc/Documents

\*Required field

Cancel Generate

**Organization:** registration name of the company responsible for the certificate.  
Do not use abbreviations or non-alphanumeric characters.

# How to fill the self-signed certificate generation fields

Create certificate

Subject

Common name<sup>①</sup>: FTOptixApplication

Organization<sup>②</sup>: Rockwell Automation

Organization unit<sup>③</sup>: ASEMI s.r.l. R&D team

Locality<sup>④</sup>: Artegno

State<sup>⑤</sup>: Udine

Country<sup>⑥</sup>: IT

OPC UA information

Application URI<sup>⑦</sup>: urn: -FactoryTalkOptixHM:FTOptixApplication

Domain names<sup>⑧</sup>: replicantime.local, localhost

Certificate settings

RSA key strength<sup>⑨</sup>: 4096 bits

Signature algorithm<sup>⑩</sup>: SHA-512

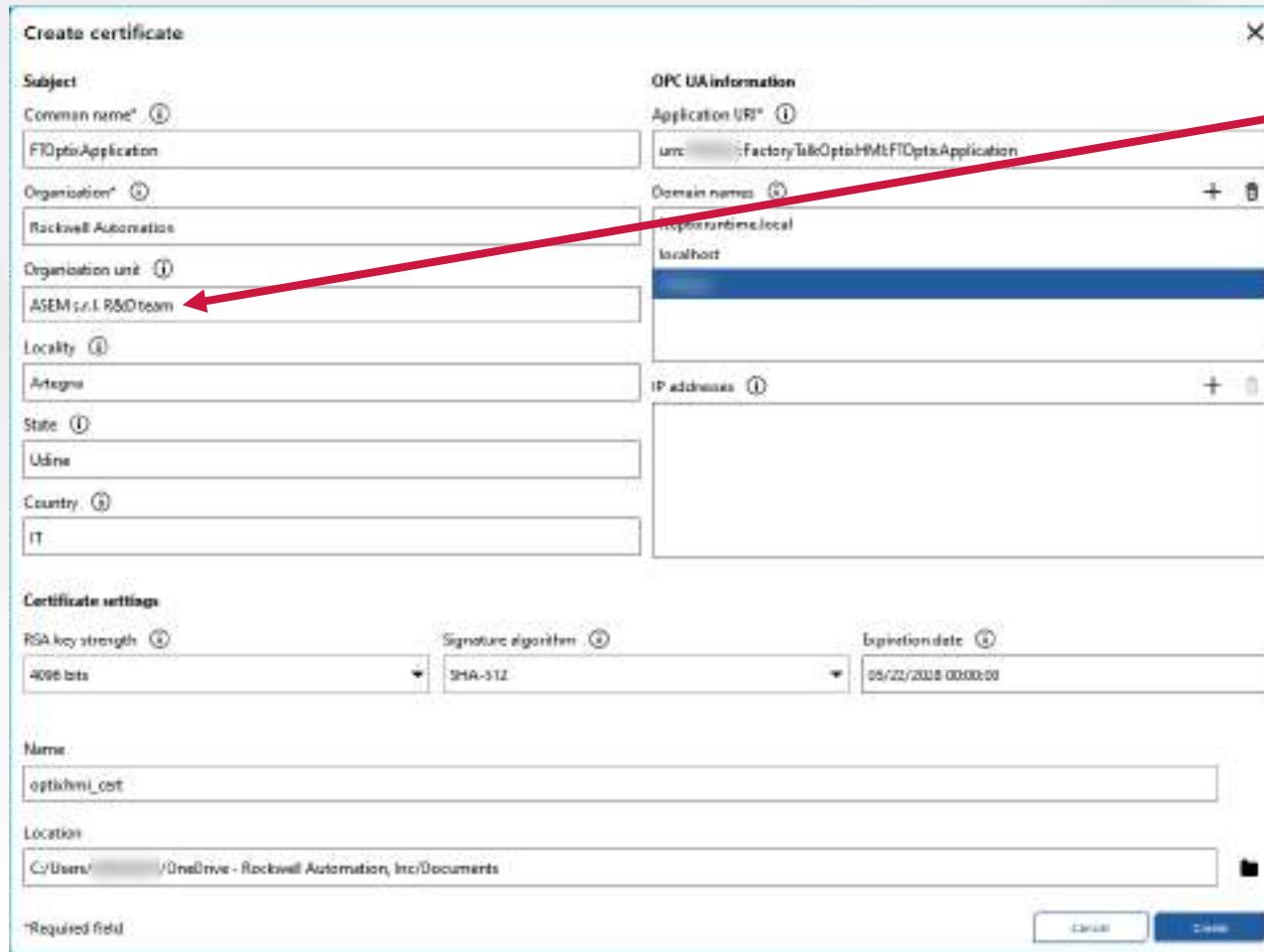
Expiration date<sup>⑪</sup>: 05/22/2028 00:00:00

Name: optikhni\_cert

Location: C:/Users/ /OneDrive - Rockwell Automation, Inc/Documents

\*Required field

Cancel Generate



**Organization unit:** local unit, office or group responsible for the certificate inside the Organization

# How to fill the self-signed certificate generation fields

Screenshot of the "Create certificate" dialog box. A red arrow points from the text "Locality: hometown of the Organization (or the Organization unit)." to the "Locality" input field, which contains the value "Artegno".

**Create certificate**

**Subject**

- Common name\*: FTOptixApplication
- Organization\*: Rockwell Automation
- Organization unit\*: ASEM s.r.l. R&D team
- Locality\*: Artegno (arrow points here)
- State\*: Udine
- Country\*: IT

**OPC UA information**

- Application URI\*: urn: -FactoryTalkOptixHM:FTOptixApplication
- Domain names:
  - ftoptixtime.local
  - localhost
- IP addresses:
  - 127.0.0.1

**Certificate settings**

- RSA key strength\*: 4096 bits
- Signature algorithm\*: SHA-512
- Expiration date\*: 05/22/2028 00:00:00

**Name:** optikhmi\_cert

**Location:** C:/Users/ /OneDrive - Rockwell Automation, Inc/Documents

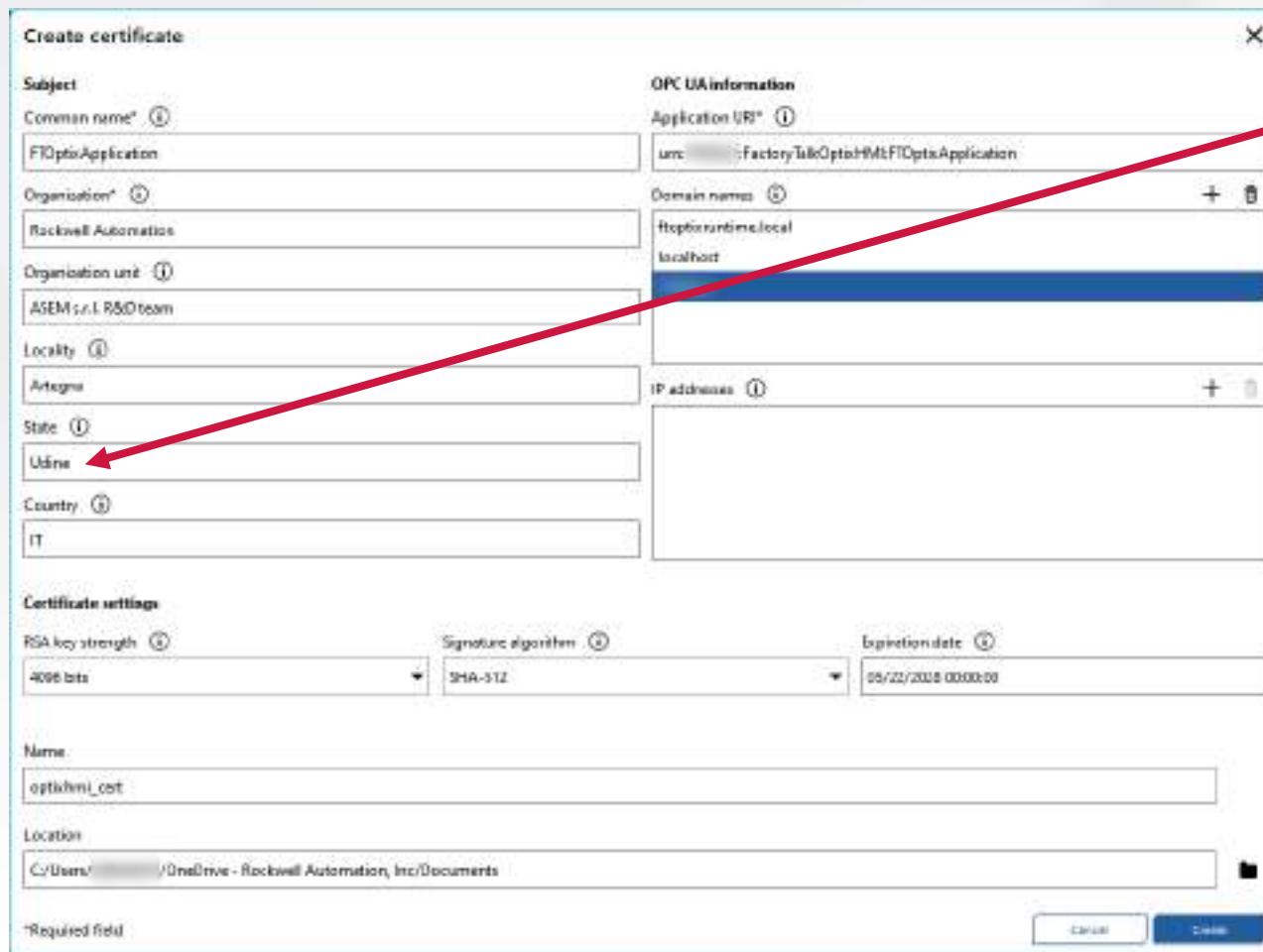
\*Required field

Cancel Generate

**Locality:** hometown of the Organization (or the Organization unit).

Must be the proper town name as specified in the postal address. Do not use abbreviations.

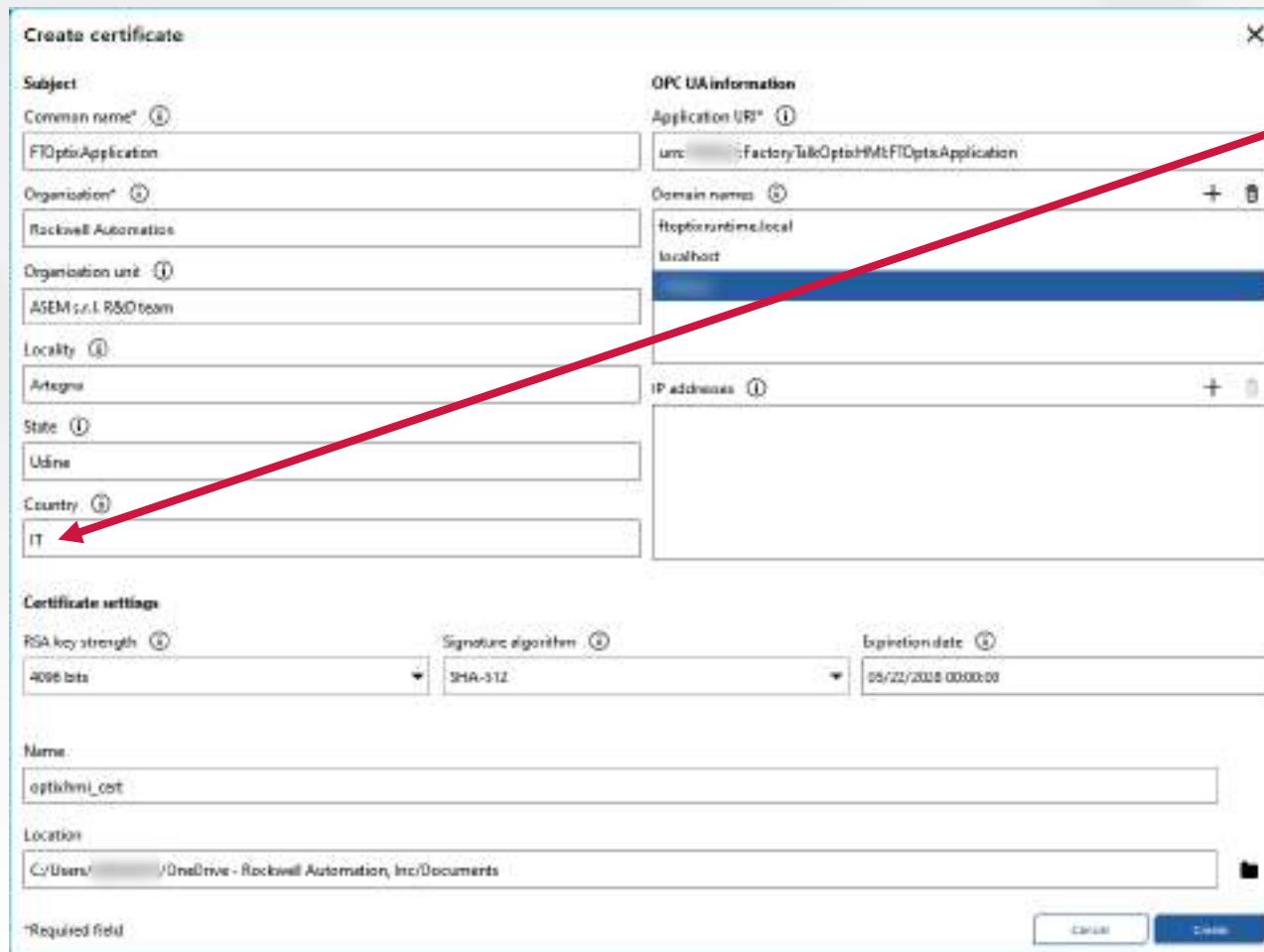
# How to fill the self-signed certificate generation fields



**State:** geographical area where the Organization (or Organization unit) is located.

This can be an US state or a province.

# How to fill the self-signed certificate generation fields



**Country:** Two-letters ISO-3166 country code where the Organization (or Organization unit) is located.

# How to fill the self-signed certificate generation fields

Create certificate

Subject

Common name<sup>①</sup>: FTOptixApplication

Organization<sup>②</sup>: Rockwell Automation

Organization unit<sup>③</sup>: ASEM srl R&D team

Locality<sup>④</sup>: Argegno

State<sup>⑤</sup>: Udine

Country<sup>⑥</sup>: IT

OPC UA information

Application URI<sup>①</sup>: urn:FactoryTalkOptixHMI:FTOptixApplication

Domain names<sup>③</sup>: ftoptimetime.local, localhost

IP addresses<sup>①</sup>

Certificate settings

RSA key strength<sup>⑦</sup>: 4096 bits

Signature algorithm<sup>⑧</sup>: SHA-512

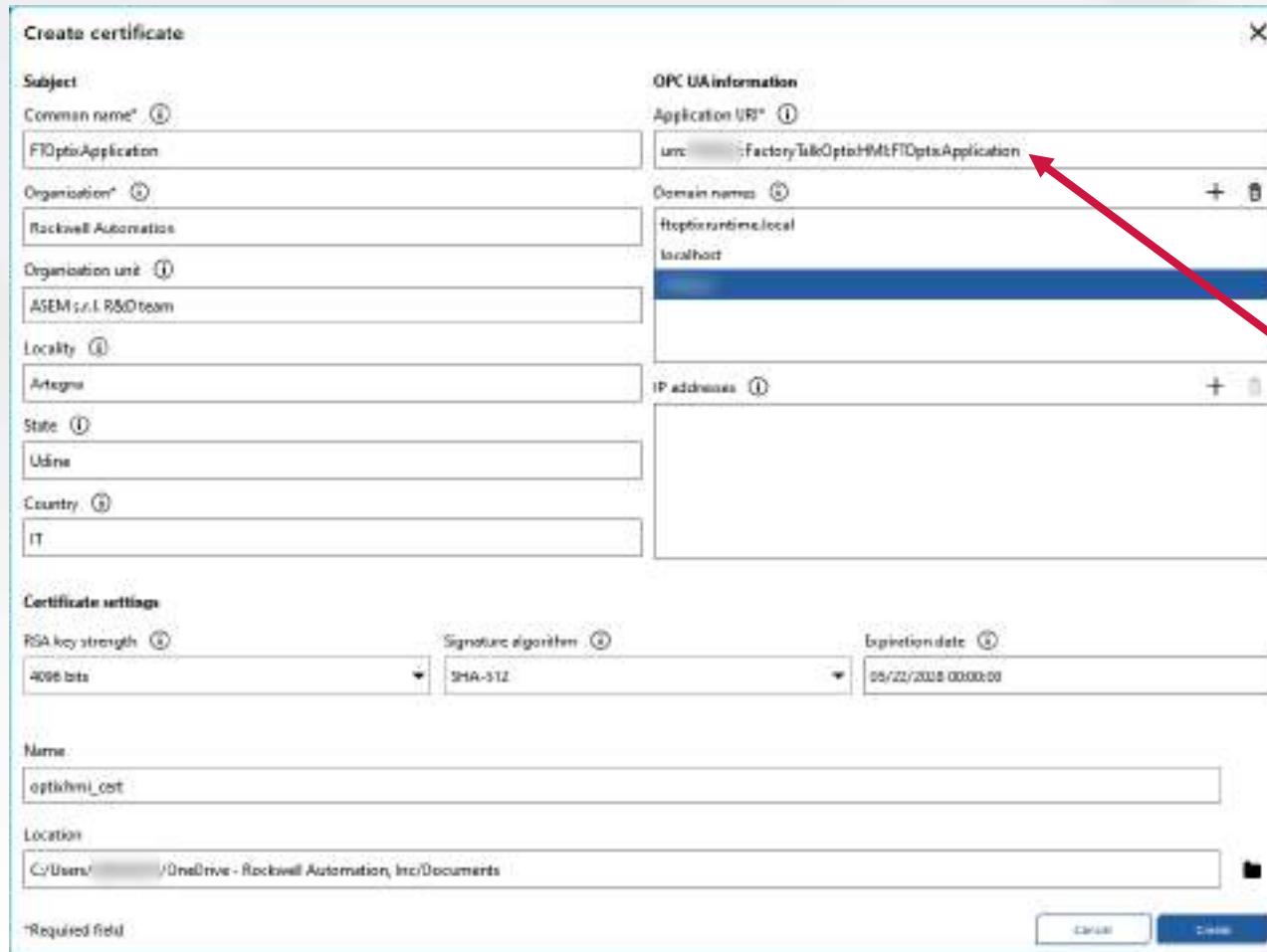
Expiration date<sup>⑨</sup>: 05/22/2028 00:00:00

Name: optikhmi\_cert

Location: C:/Users/.../OneDrive - Rockwell Automation, Inc/Documents

\*Required field

Cancel Done



Application URI: the unique identifier for the project on a specific machine.

In FactoryTalk Optix, this must be the same as the Product URI in the server configuration

Properties

Name: OPCUAserver

Type: OPC UA server

Server

Endpoint URL: opc.tcp://localhost:58100

Nodes to publish

Maximum number of connections: 1

Maximum number of subscriptions per client: 10

Use node path in NodeIds: False

Sampling interval: 0000-00-00 00:00

Max array length: 65536

Security

Minimum message security mode: Signature and cryptography

Minimum security policy: Basic256Sha256

Server certificate file: %PROJECTDIR%\PKI\Own\Certs\optikhmi\_cert.der

Server private key file: %PROJECTDIR%\PKI\Own\Certs\optikhmi\_cert.pvk

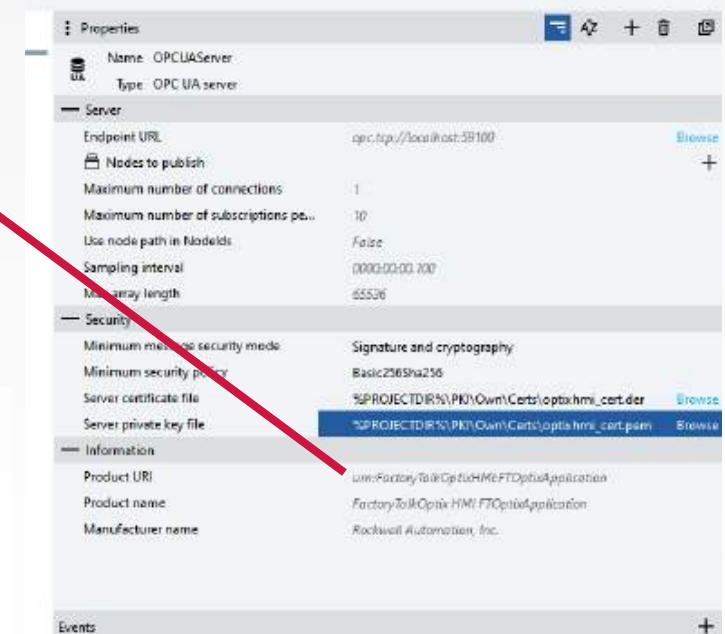
Information

Product URI: urn:FactoryTalkOptixHMI:FTOptixApplication

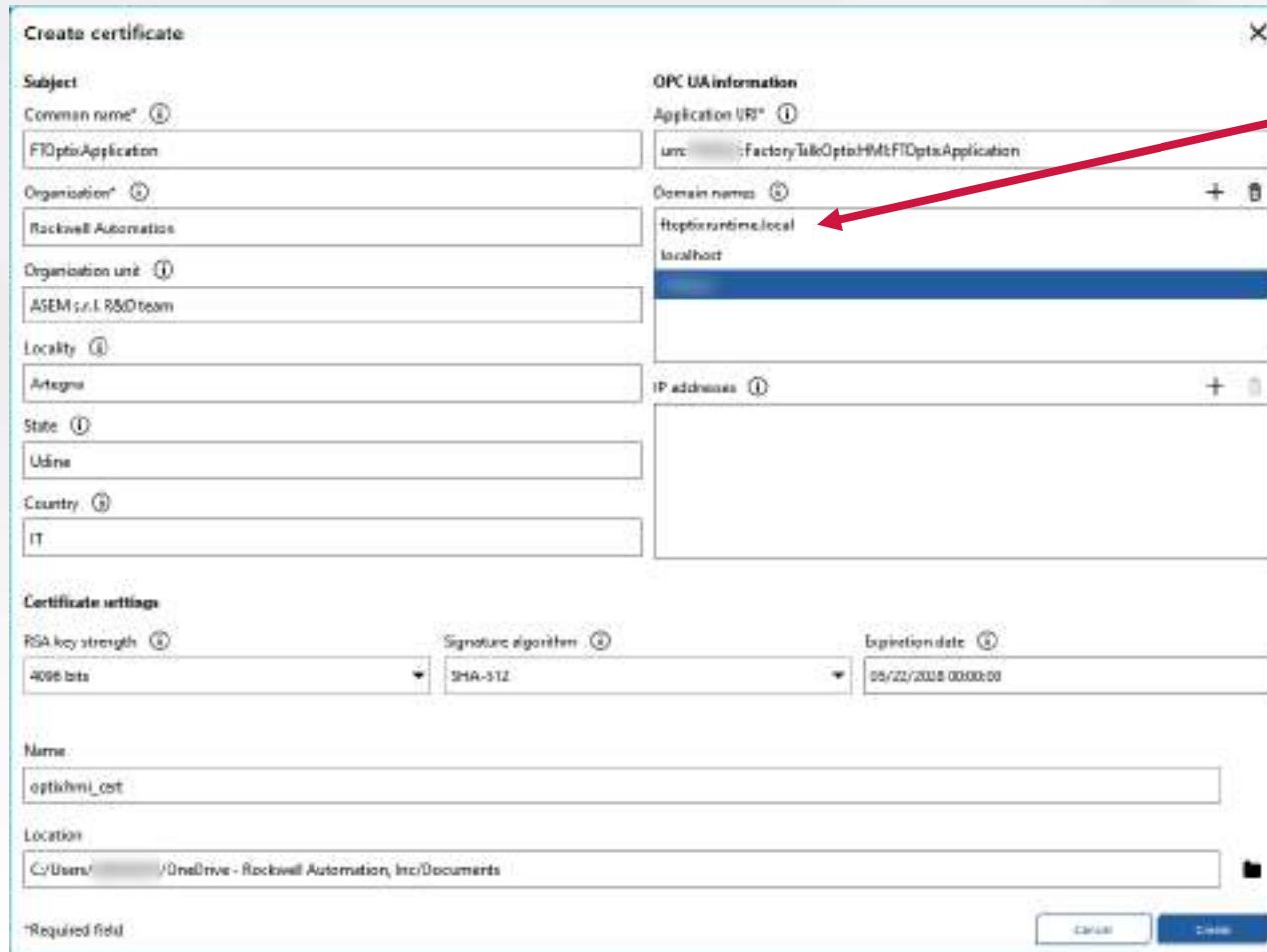
Product name: FactoryTalkOptix HMI:FTOptixApplication

Manufacturer name: Rockwell Automation, Inc.

Events



# How to fill the self-signed certificate generation fields

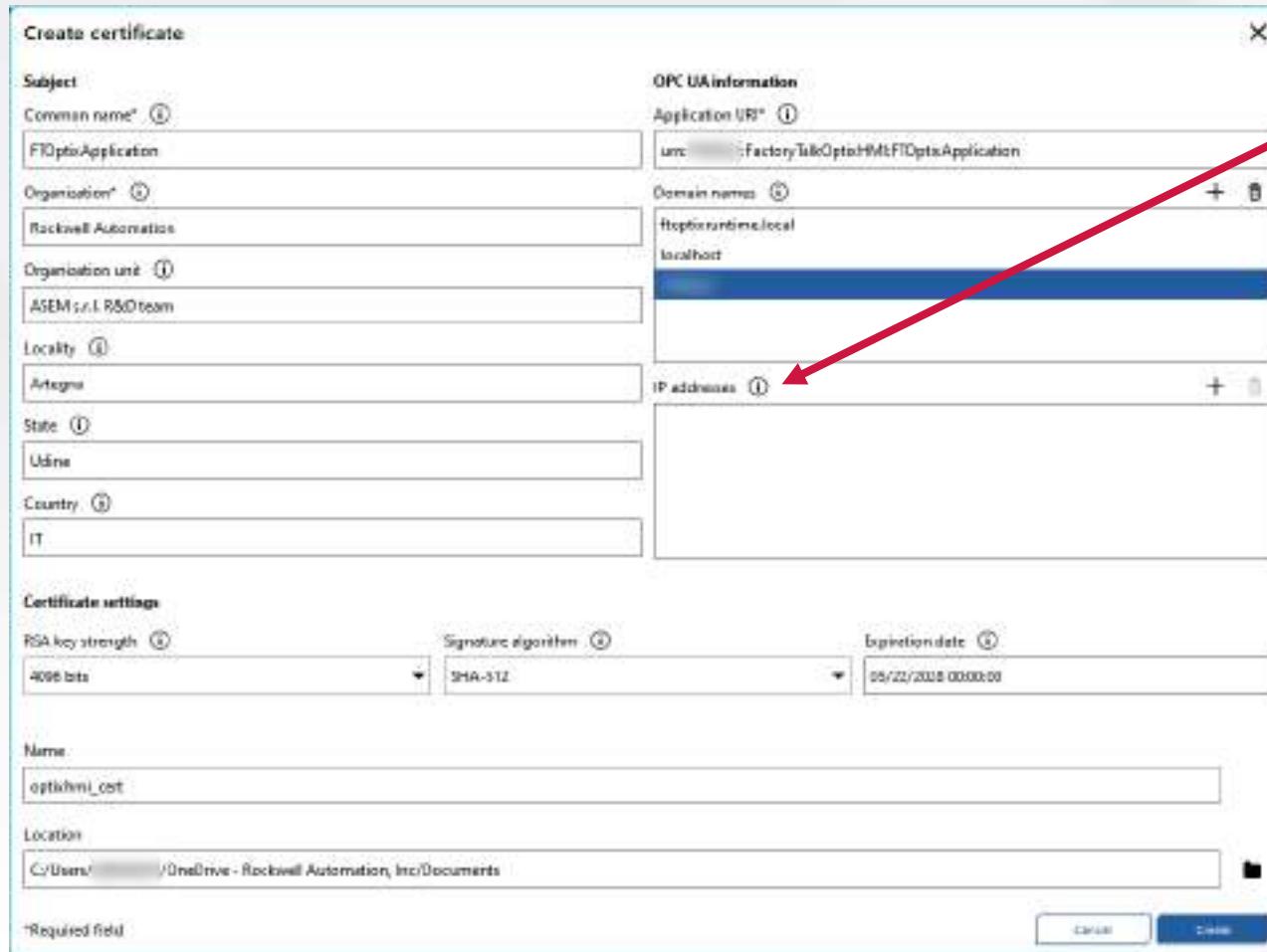


**Domain names:** a list of domain names that can be used to access the machine.

This field contains a list of URLs (like myapplication.example.com) or Active Directory name for the host machine.

This field should contain all the available domains used to access the host machine either locally or remotely.

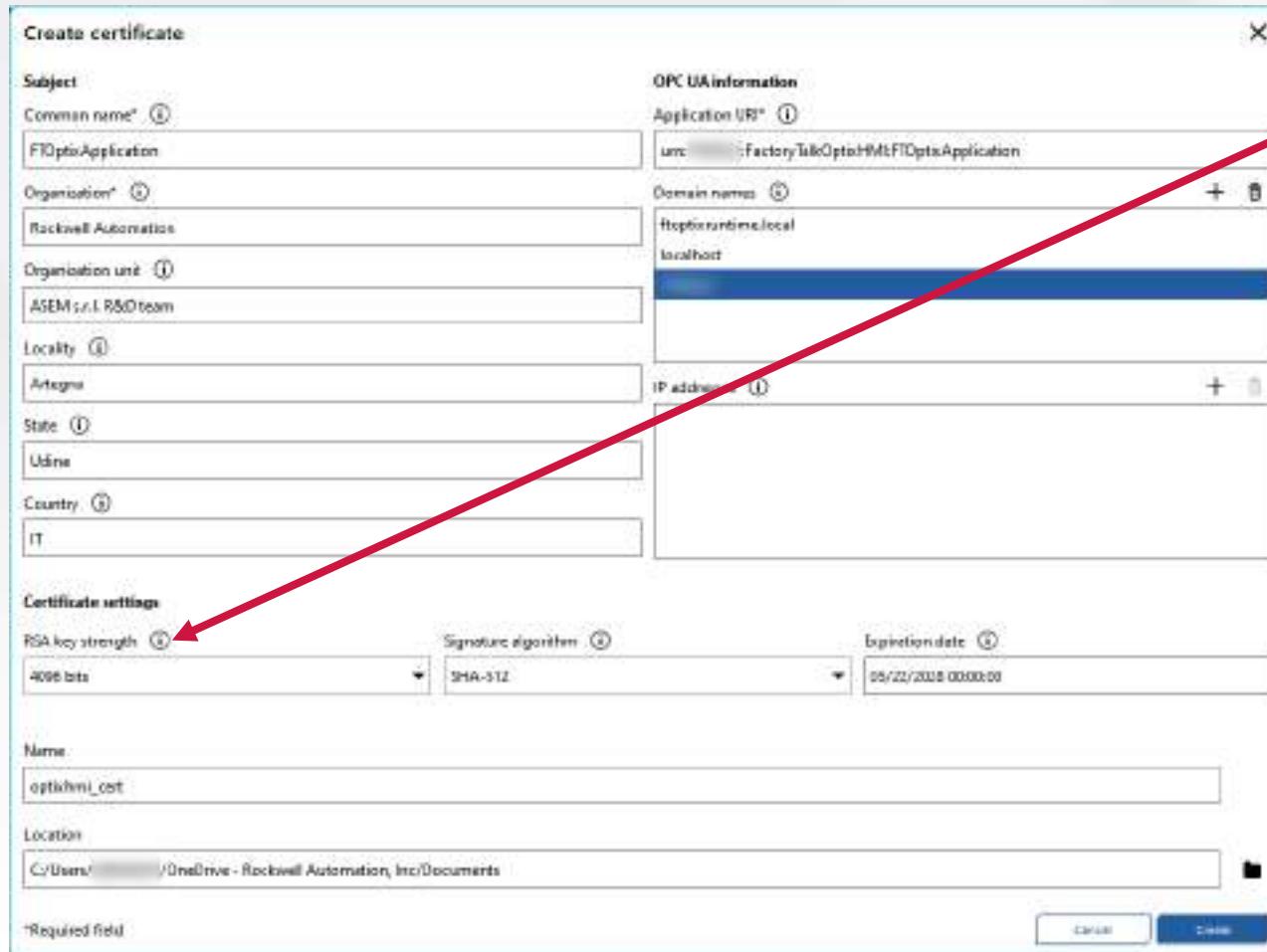
# How to fill the self-signed certificate generation fields



**IP address:** a list of IP addresses that can be used to access the machine.

This field is optional and should not be used in normal condition as the certificate would be invalidated if the IP address changes.

# How to fill the self-signed certificate generation fields



**RSA key strength:** length in bits of the RSA key.

Longer keys are more secure, but need more computing resources to be processed.

It is preferable to choose the highest value that does not impact on the host hardware.

# How to fill the self-signed certificate generation fields

Create certificate

Subject

Common name<sup>①</sup>: FTOptixApplication

Organization<sup>②</sup>: Rockwell Automation

Organization unit<sup>③</sup>: ASEM s.r.l. R&D team

Locality<sup>④</sup>: Artegno

State<sup>⑤</sup>: Udine

Country<sup>⑥</sup>: IT

OPC UA information

Application URI<sup>⑦</sup>: urn: -FactoryTalkOptixHM:FTOptixApplication

Domain names<sup>⑧</sup>: itoptixruntime.local  
localhost

IP addresses<sup>⑨</sup>

Certificate settings

RSA key strength<sup>⑩</sup>: 4096 bits

Signature algorithm<sup>⑪</sup>: SHA-512

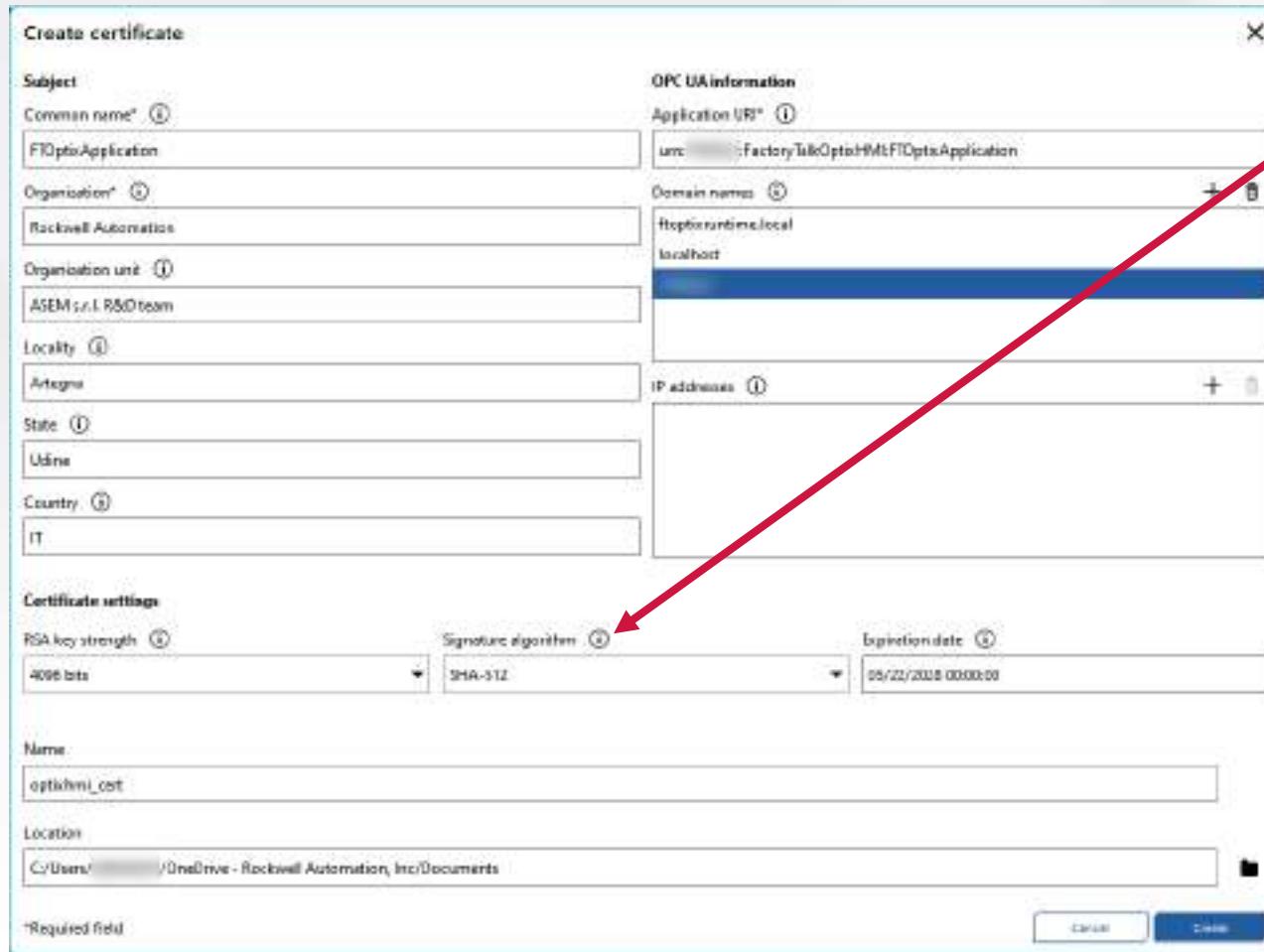
Expiration date<sup>⑫</sup>: 05/22/2028 00:00:00

Name: optikhmi\_cert

Location: C:/Users/ /OneDrive - Rockwell Automation, Inc/Documents

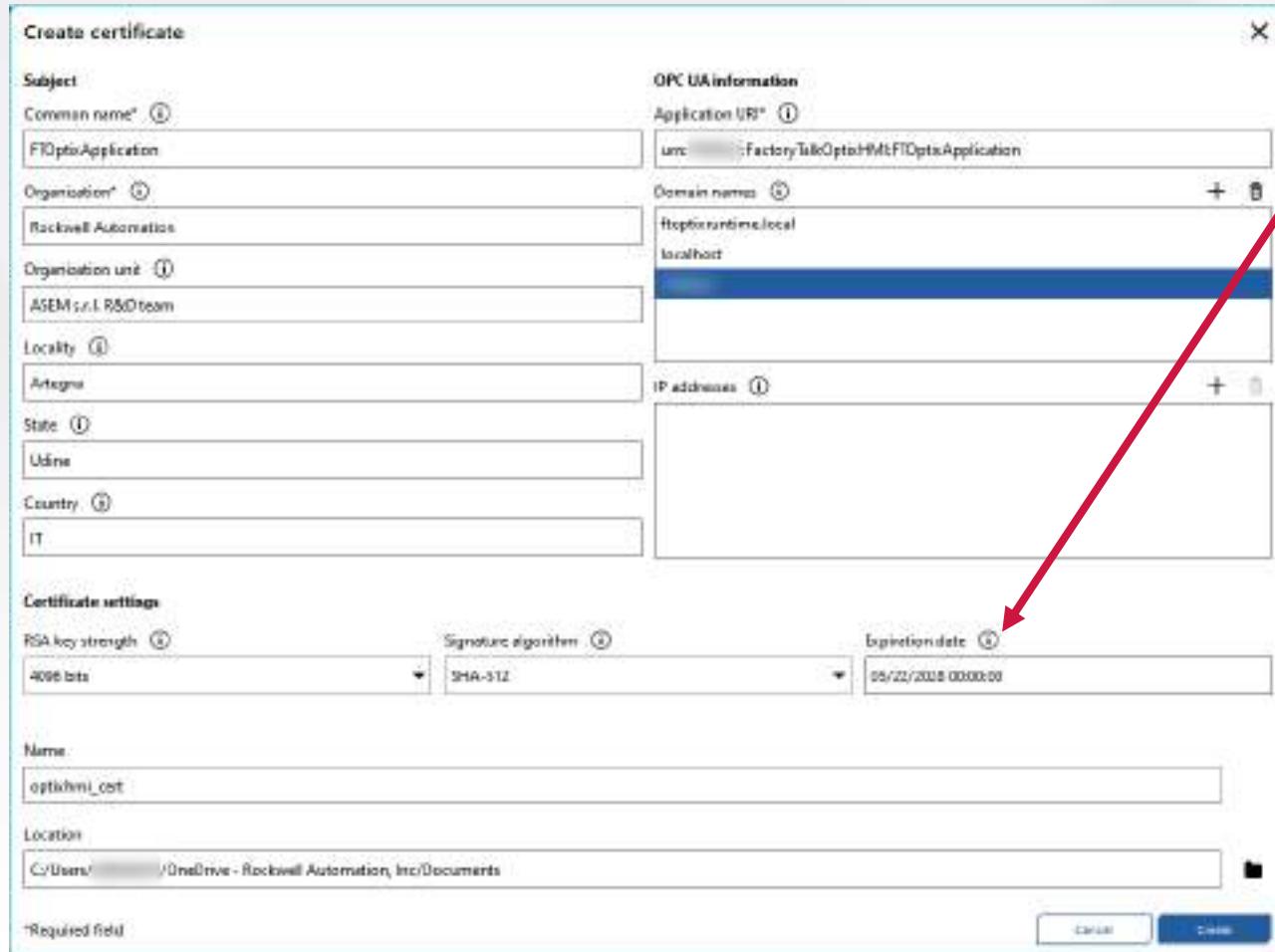
\*Required field

Cancel Create



**Signature algorithm:** algorithm used by the devices to sign the data using these certificates.

# How to fill the self-signed certificate generation fields



**Expiration date:** how long the certificate will be valid before a new certificate should be requested/generated.

Longer times means less maintenance but are more prone to tampering, hijacking or identity thefts (as the certificate remains the same for longer time, increasing attack surface).

# How to fill the self-signed certificate generation fields

Create certificate

Subject

Common name\* ⓘ  
FTOptixApplication

Organization\* ⓘ  
Rockwell Automation

Organization unit\* ⓘ  
ASEM s.r.l. R&D team

Locality\* ⓘ  
Artegno

State\* ⓘ  
Udine

Country\* ⓘ  
IT

OPC UA information

Application URI\* ⓘ  
urn: -FactoryTalkOptixHM:FTOptixApplication

Domain names\* ⓘ  
ftoptiontime.local  
localhost

IP addresses\* ⓘ  
+ ⌂

Certificate settings

RSA key strength\* ⓘ  
4096 bits

Signature algorithm\* ⓘ  
SHA-512

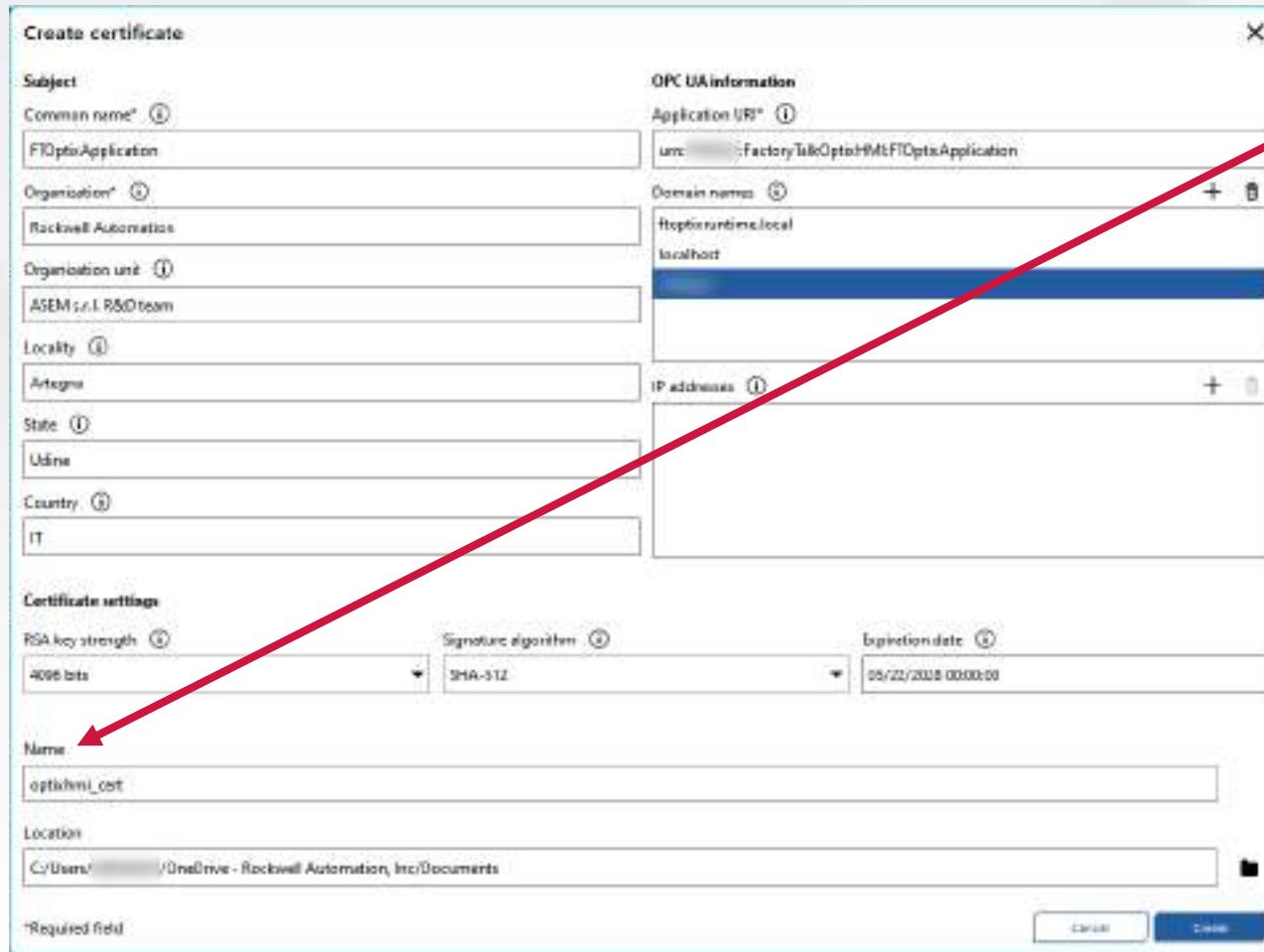
Expiration date\* ⓘ  
05/22/2025 00:00:00

Name:

Location:  ⌂

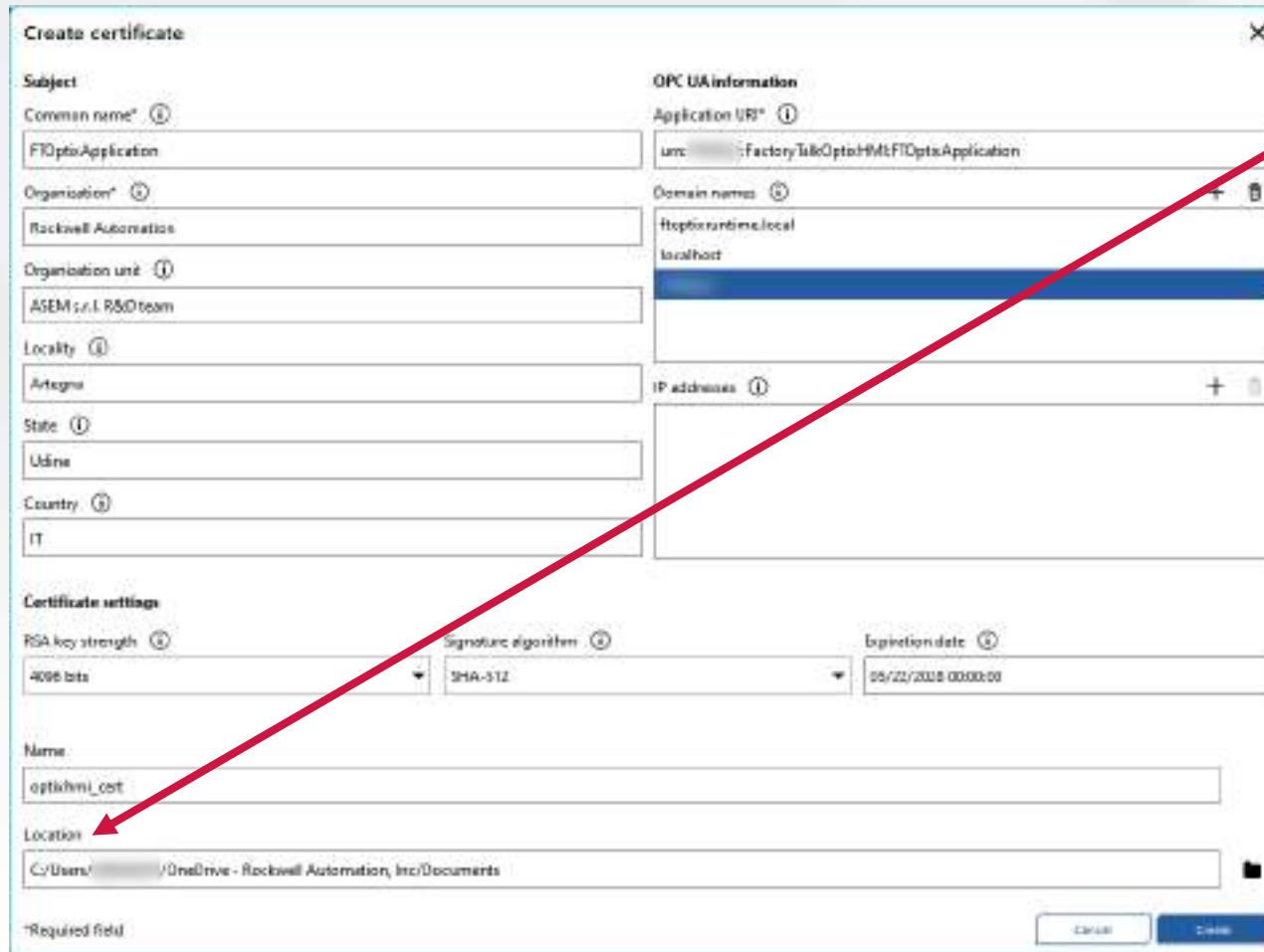
\*Required field

Cancel Generate



**Name:** name of the output certificate.  
Can be any valid file name.

# How to fill the self-signed certificate generation fields



**Location:** where to store the generated certificate.  
Can be any valid path.

# Creating a self-signed certificate with OpenSSL

- Generate a Private Key
- Create a Certificate Signing Request (CSR)
  - Fill in all the details in the command line (refer to the procedure using Factory Optix to fill the fields)
- Generate the self-signed certificate
- Convert private key to a “pem” file
- Extract the public key and convert it to a “der” file

```
# Generate the private key
openssl genrsa -out private.key 2048
# Create the certificate sign request
openssl req -new -key private.key -out request.csr
# Generate the self-signed certificate
openssl x509 -req -days 365 -in request.csr -signkey private.key -out certificate.crt
# Convert the private key to a .pem file
openssl rsa -in private.key -out private.pem
# Extract the public key from the generated key
openssl rsa -in private.key -pubout -out public.pem
# Convert the public key to a .der file
openssl rsa -pubin -in public.pem -outform der -out public.der
```

# Creating a CA-trusted certificate with OpenSSL

- Choose a Certificate Authority
  - Many online providers are available, some for free, some other require a fee
- Generate a Private Key
- Create a Certificate Signing Request (CSR)
  - Fill in all the details in the command line (refer to the procedure using Factory Optix to fill the fields)
- Send the CSR file to the Certificate Authority
  - Depending on the Certificate Authority, additional validations or checks might be needed
  - Depending on the Certificate Authority, the returned file might need to be converted to the "pem" and "der" files

```
# Generate the private key
openssl genrsa -out private.key 2048
# Create the certificate sign request
openssl req -new -key private.key -out request.csr
# Now send the generated certificate signing request file to the CA
# to proceed with the signing process.
```

# Using OPC-UA certificates

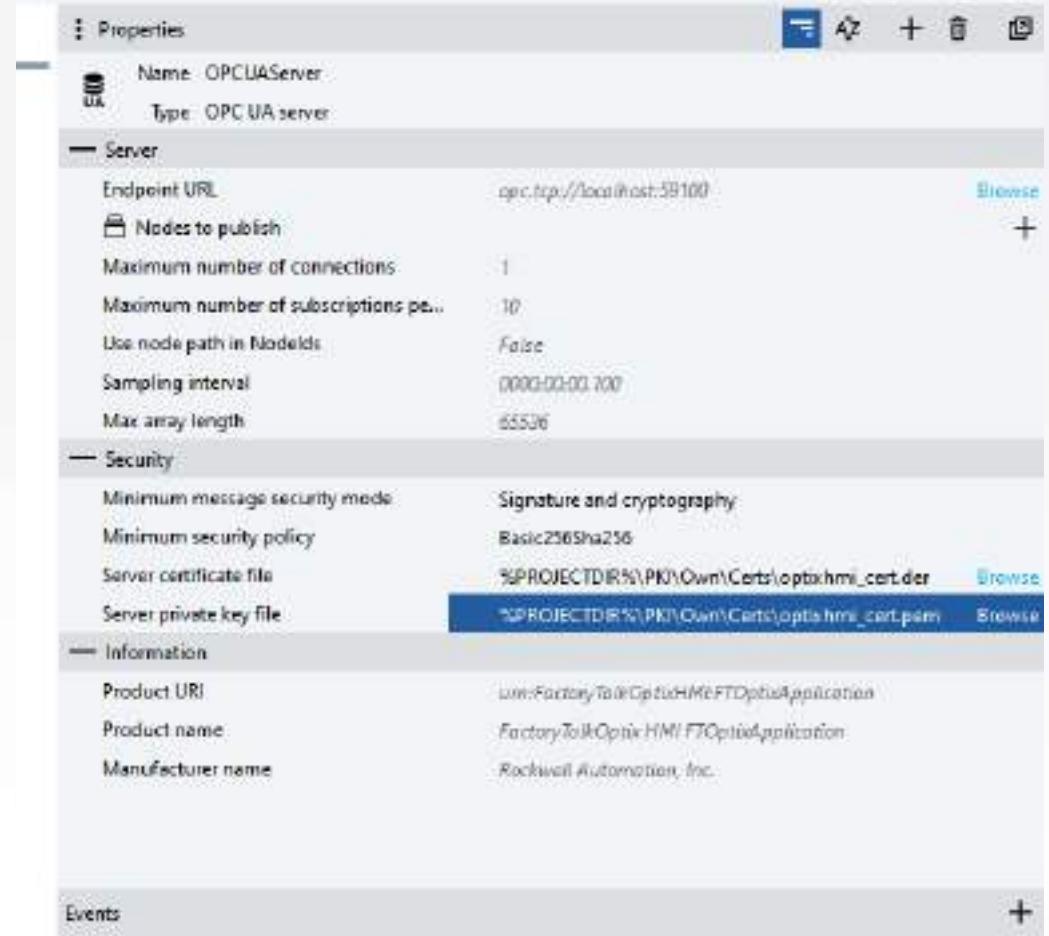


# Things to know about OPC-UA certificates

- A FactoryTalk Optix OPC-UA server always uses certificates to perform some actions, this depends on the server configuration on the “Minimum message security mode”:
  - None → Certificates are only used by the client to login to the server
  - Signature only → Certificates are used to sign messages, but content of messages is not encrypted
  - Signature and cryptography → Certificates are used to sign messages and to encrypt the content of the message
- Even if the “Minimum message security mode” is set to “None”, certificates are still required to complete the login process
- In addition to the security mode, the server can also allow only specific encryption algorithms to be used by the connected clients
- FactoryTalk Optix automatically creates a self-signed certificate for testing purposes, **the automatically created self-signed certificate should never be used in a production environment**
  - Certificates should be issued by a trusted authority
  - FactoryTalk Optix Studio includes a dedicated tool to generate custom self-signed certificates if needed

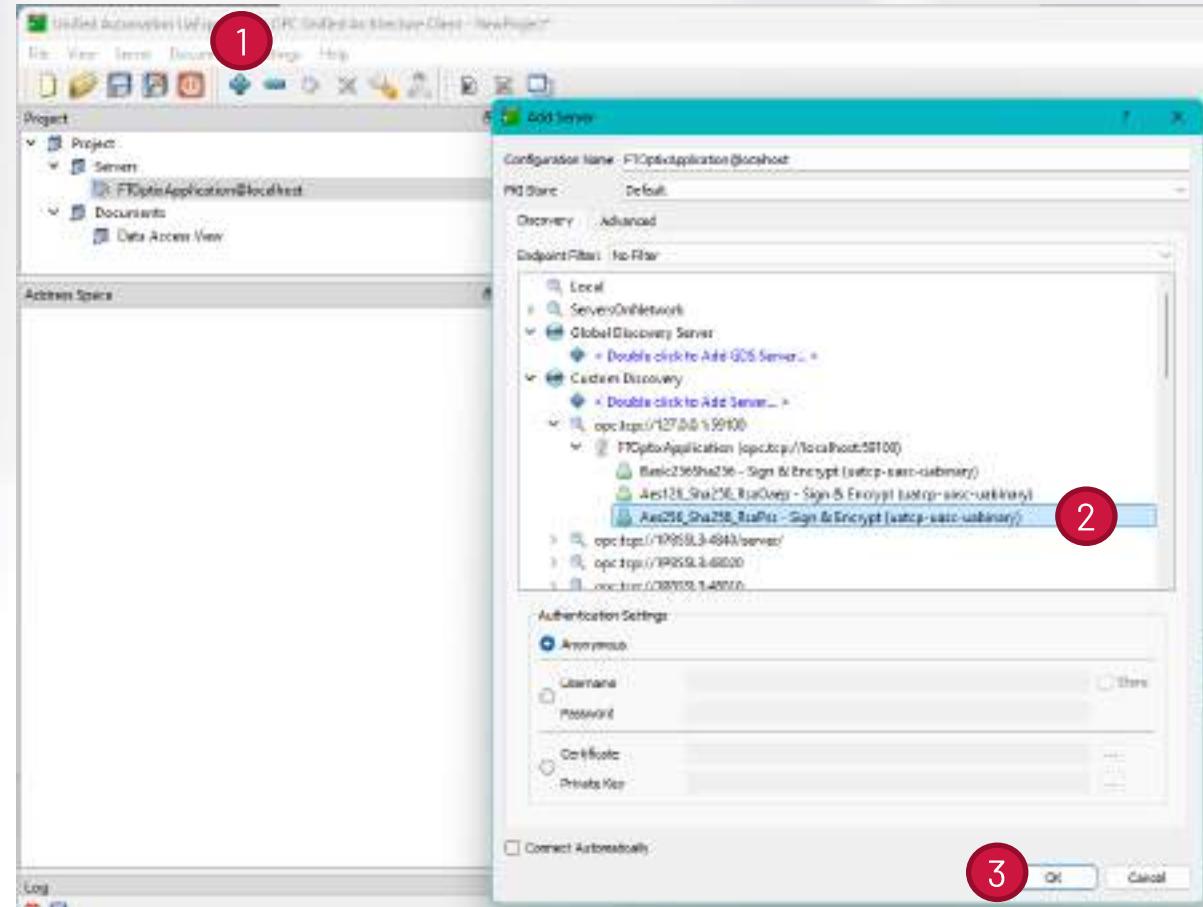
# Using a custom certificate in the OPC-UA server

- To apply custom certificates to an OPC-UA server, simply import them in the project and set the relevant properties in the server configuration:
  - Server certificate file: public key of the certificate ("der" format)
  - Server private key file: private key of the certificate ("pem" format)
- Additional settings can be used to enforce a minimum-security policy for connected clients
  - This allows forcing stronger encryption and signing algorithm for all connected clients



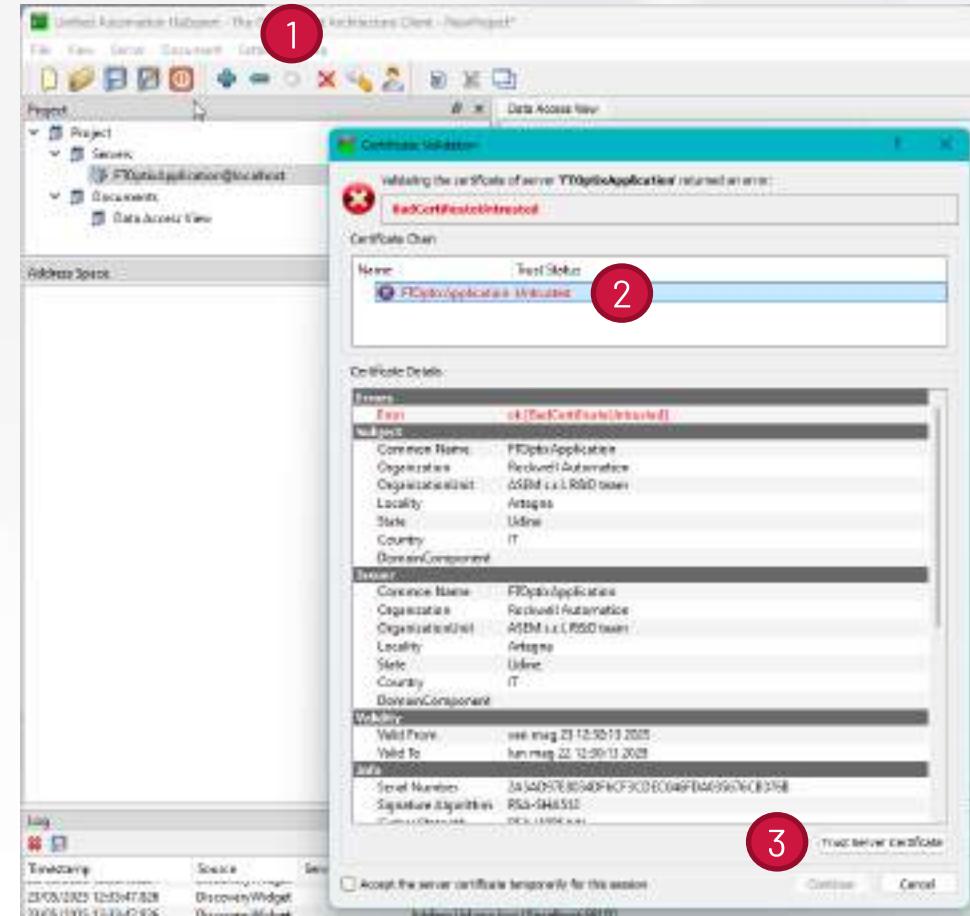
# Connecting to an OPC-UA server using custom certificates

- If the server is forcing clients to use certificates, when browsing the available servers, no unsecure connections are now listed
- Select the required sign and encryption protocol to be used
- If a self-signed certificate was used, the client will prompt the user to accept or reject the server certificate
- After trusting the certificate, the client can now attempt a connection
- FactoryTalk Optix will reject the client certificate by default, which needs to be trusted manually by moving it to the Trusted folder



# Connecting to an OPC-UA server using custom certificates

- If the server is forcing clients to use certificates, when browsing the available servers, no unsecure connections are now listed
- Select the required sign and encryption protocol to be used
- If a self-signed certificate was used, the client will prompt the user to accept or reject the server certificate
- After trusting the certificate, the client can now attempt a connection
- FactoryTalk Optix will reject the client certificate by default, which needs to be trusted manually by moving it to the Trusted folder



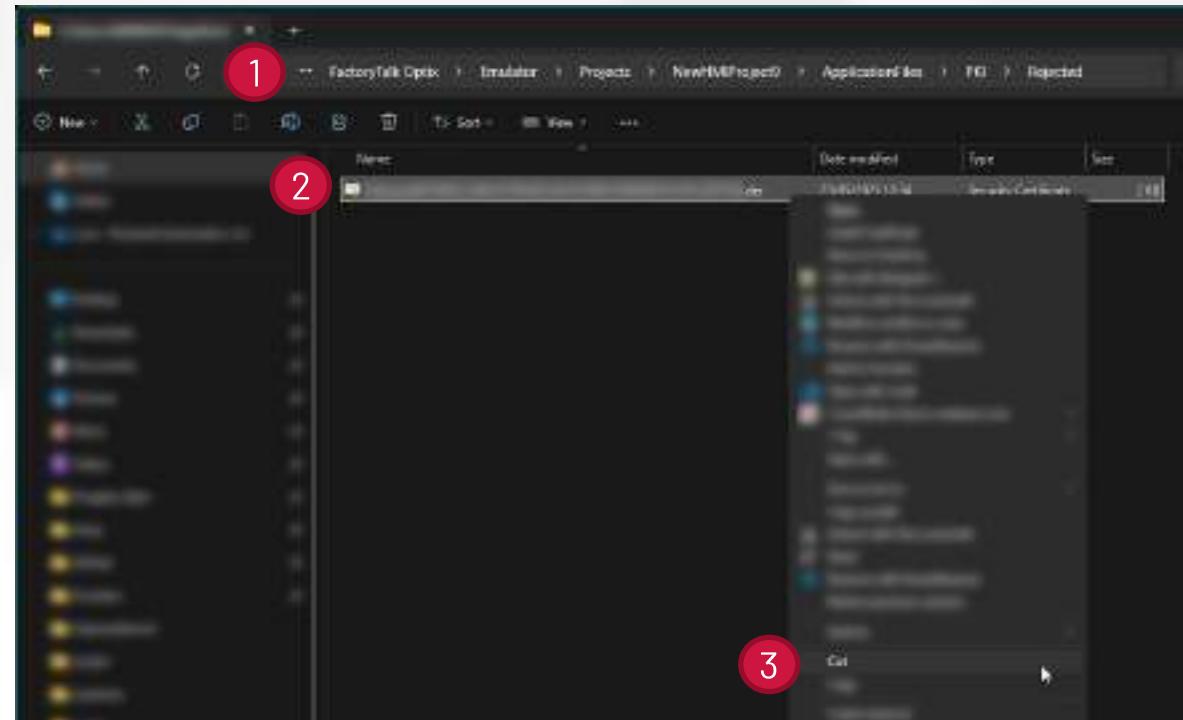
# Connecting to an OPC-UA server using custom certificates

- If the server is forcing clients to use certificates, when browsing the available servers, no unsecure connections are now listed
- Select the required sign and encryption protocol to be used
- If a self-signed certificate was used, the client will prompt the user to accept or reject the server certificate
- After trusting the certificate, the client can now attempt a connection
- FactoryTalk Optix will reject the client certificate by default, which needs to be trusted manually by moving it to the Trusted folder

Timestamp	Source	Server	Message
23/05/2025 12:33:47,826	DiscoveryWidget		Adding Url opc.tcp://localhost:59100
23/05/2025 12:34:06,580	Server Node	FTOptixApplicat..	Endpoint: 'opc.tcp://localhost:59100'
23/05/2025 12:34:09,580	Server Node	FTOptixApplicat..	Security policy: 'https://opcfoundation.org/UA/SecurityPolicy#Aes256_Sha256_RsaPss'
23/05/2025 12:34:09,580	Server Node	FTOptixApplicat..	ApplicationId: urn: [REDACTED]FactoryTalkOptixHmtFTOptixApplication'
23/05/2025 12:34:09,580	Server Node	FTOptixApplicat..	Used UserTokenType: Anonymous
23/05/2025 12:34:27,940	Server Node	FTOptixApplicat..	Error 'BadSecurityCheckFailed' was returned during OpenSecureChannel
23/05/2025 12:34:27,949	Server Node	FTOptixApplicat..	Connection status of server 'FTOptixApplication@localhost' changed to 'Disconnected'.

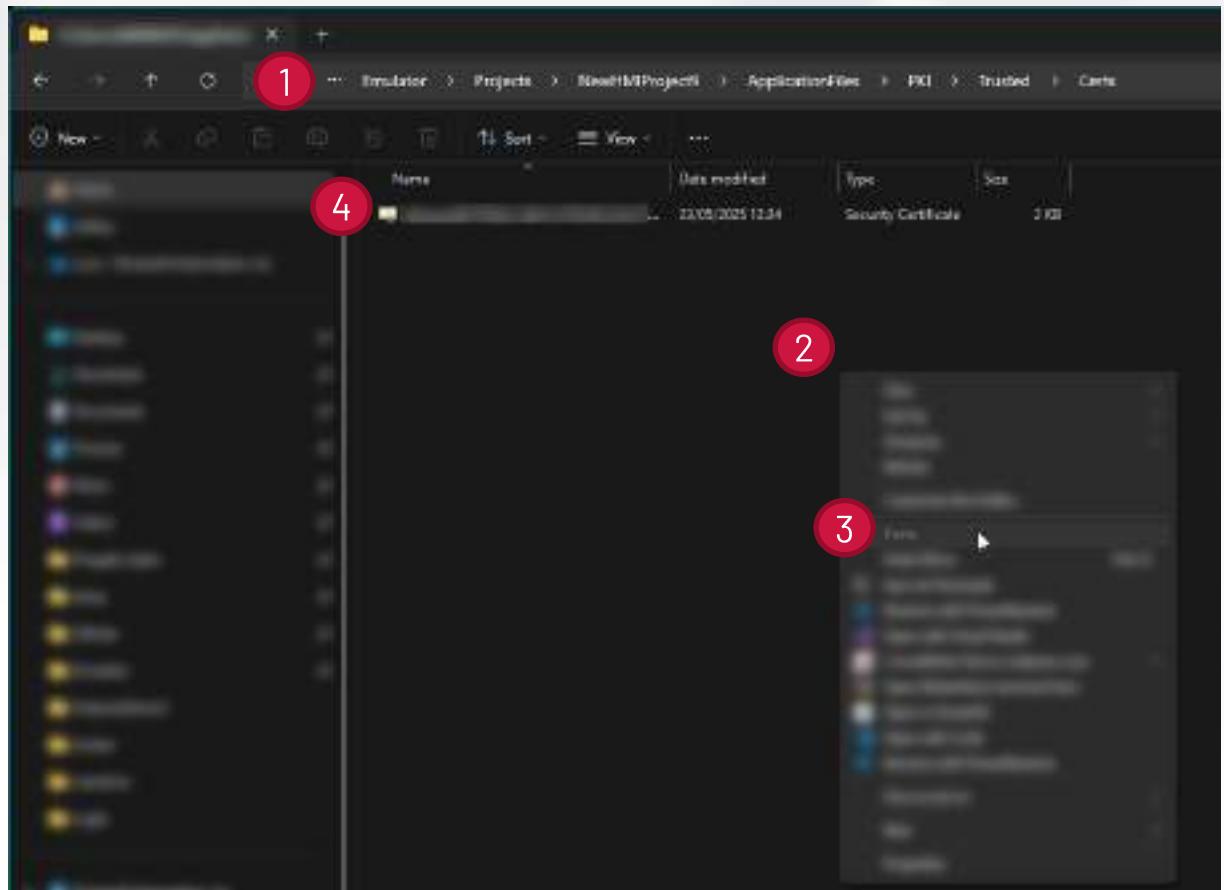
# Connecting to an OPC-UA server using custom certificates

- If the server is forcing clients to use certificates, when browsing the available servers, no unsecure connections are now listed
- Select the required sign and encryption protocol to be used
- If a self-signed certificate was used, the client will prompt the user to accept or reject the server certificate
- After trusting the certificate, the client can now attempt a connection
- FactoryTalk Optix will reject the client certificate by default, which needs to be trusted manually by moving it to the Trusted folder



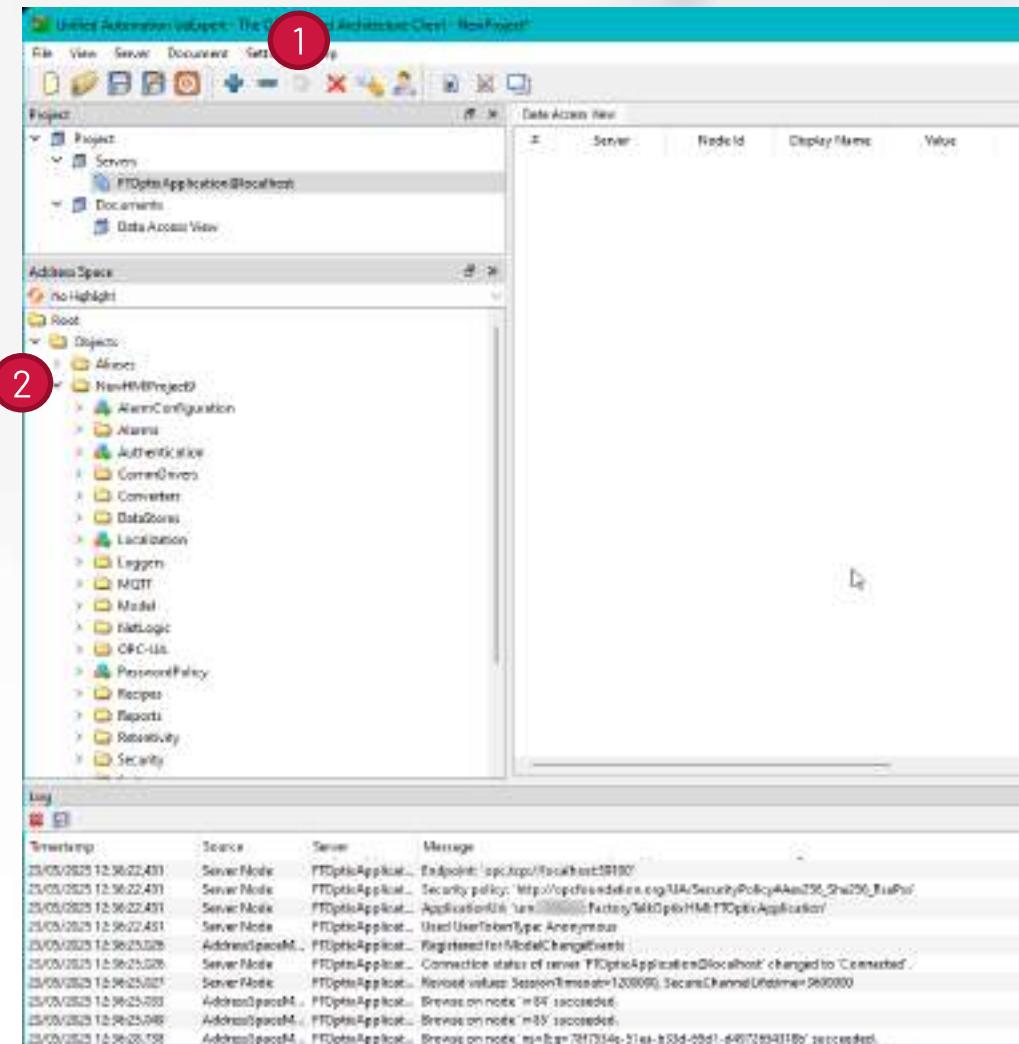
# Connecting to an OPC-UA server using custom certificates

- If the server is forcing clients to use certificates, when browsing the available servers, no unsecure connections are now listed
- Select the required sign and encryption protocol to be used
- If a self-signed certificate was used, the client will prompt the user to accept or reject the server certificate
- After trusting the certificate, the client can now attempt a connection
- FactoryTalk Optix will reject the client certificate by default, which needs to be trusted manually by moving it to the Trusted folder



# Connecting to an OPC-UA server using custom certificates

- If the server is forcing clients to use certificates, when browsing the available servers, no unsecure connections are now listed
- Select the required sign and encryption protocol to be used
- If a self-signed certificate was used, the client will prompt the user to accept or reject the server certificate
- After trusting the certificate, the client can now attempt a connection
- FactoryTalk Optix will reject the client certificate by default, which needs to be trusted manually by moving it to the Trusted folder



# Return parameters from NetLogic methods



# Return parameters from NetLogic methods

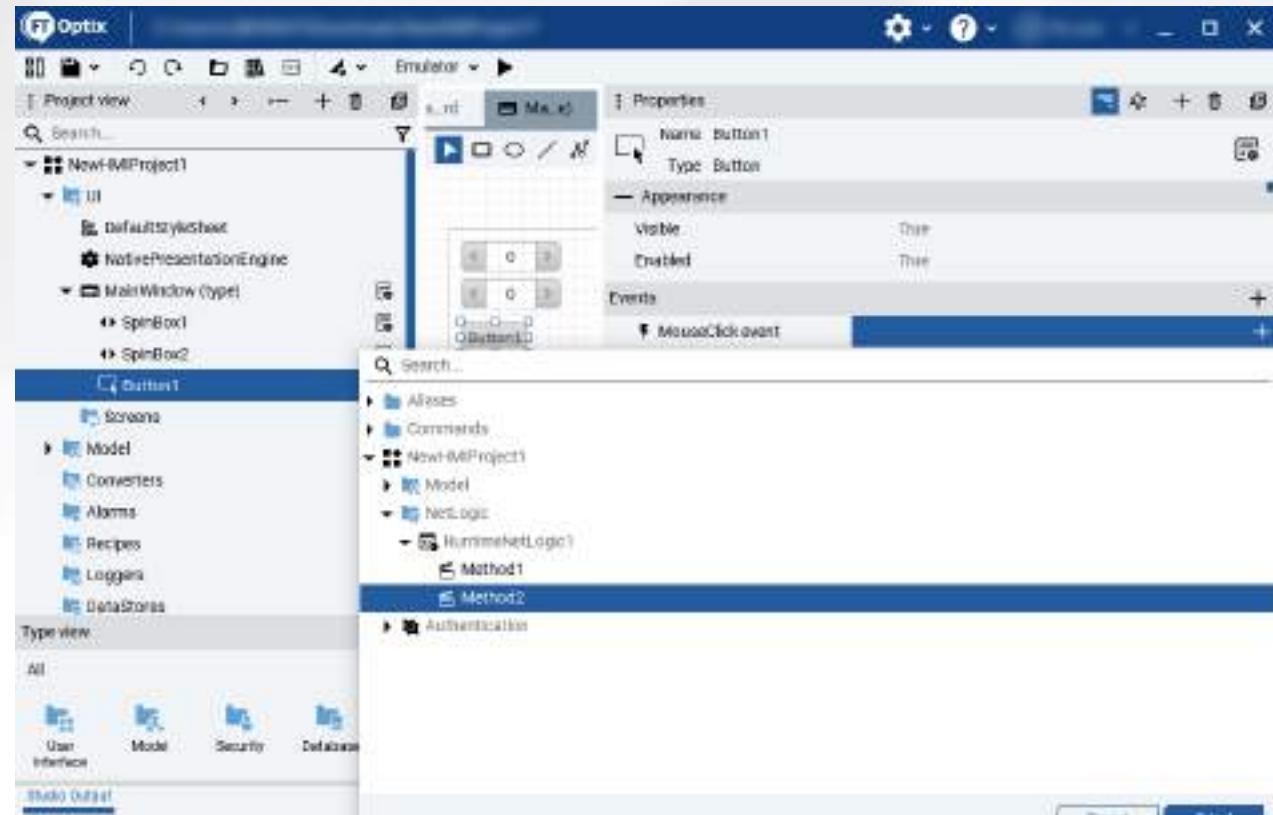
- All methods to be used in FactoryTalk Optix must be void (returning nothing)
- Parameters can be returned by using the «out» modifier in the arguments list

```
// This one will not work, returned value is discarded
[ExportMethod]
public int Method1(int a, int b) {
    return a + b;
}
```

```
// This one will work, returned value can be used
[ExportMethod]
public void Method2(int a, int b, out int c) {
    c = a + b;
}
```

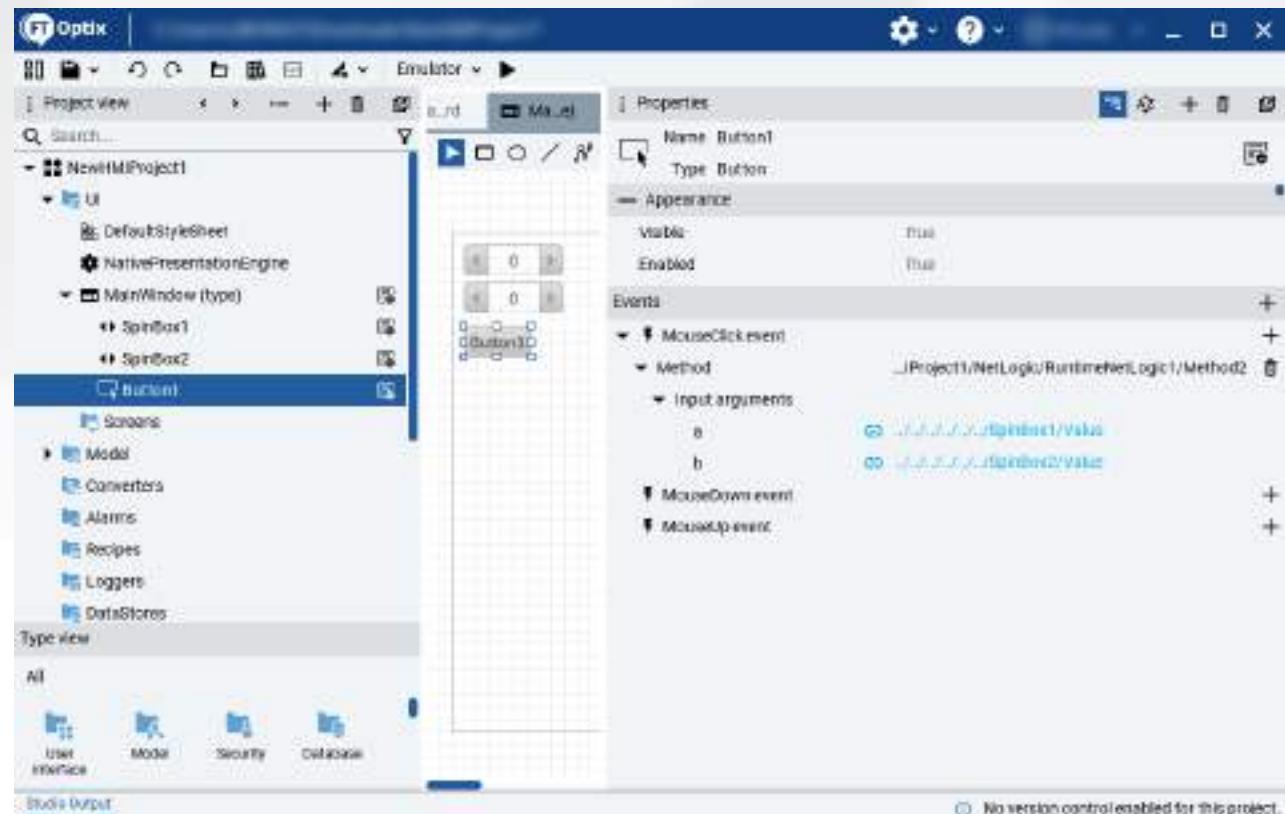
# Return parameters from NetLogic methods

- On any method that can trigger an event, call the method exposed by the NetLogic
- Configure the input parameters (if any)
- Add a second method invocation
- Use a SetVariableValue to get the output value from the MethodContainer1 and set it to a different variable (or property)



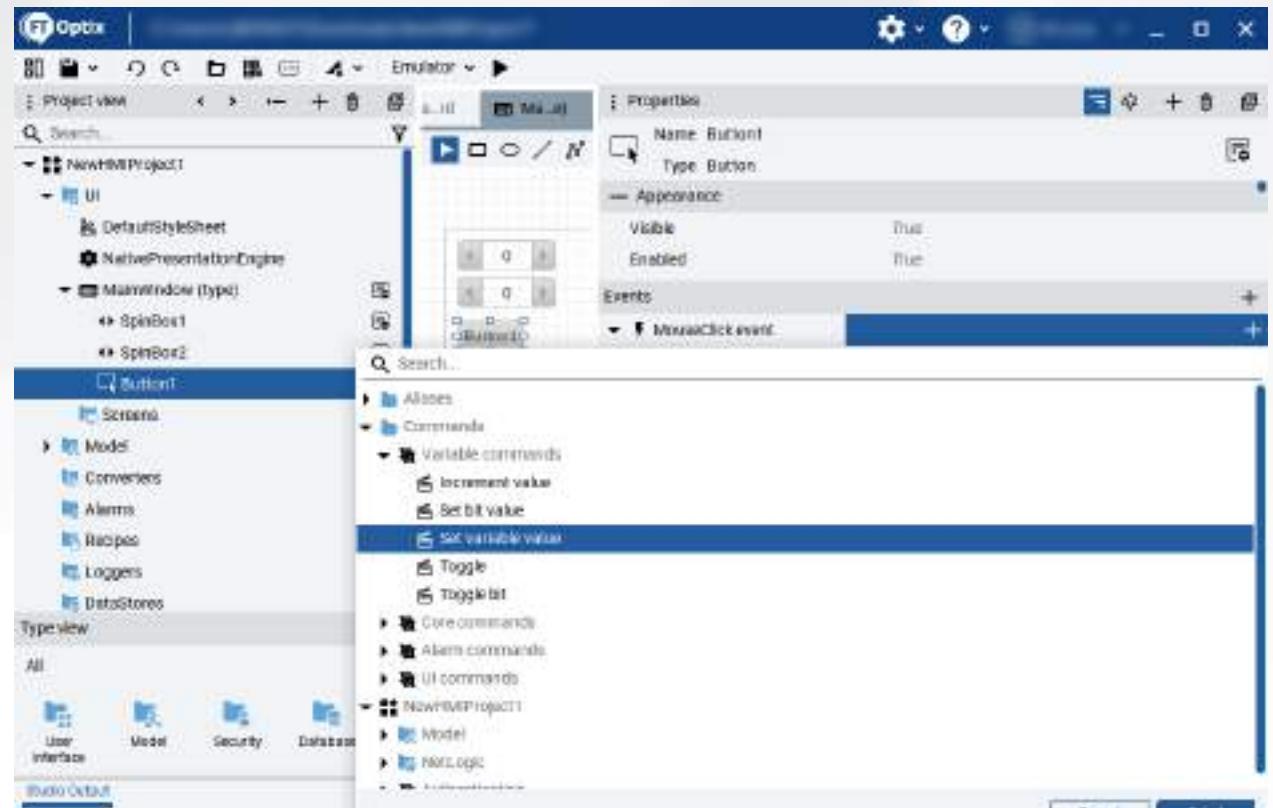
# Return parameters from NetLogic methods

- On any method that can trigger an event, call the method exposed by the NetLogic
- Configure the input parameters (if any)
- Add a second method invocation
- Use a SetVariableValue to get the output value from the MethodContainer1 and set it to a different variable (or property)



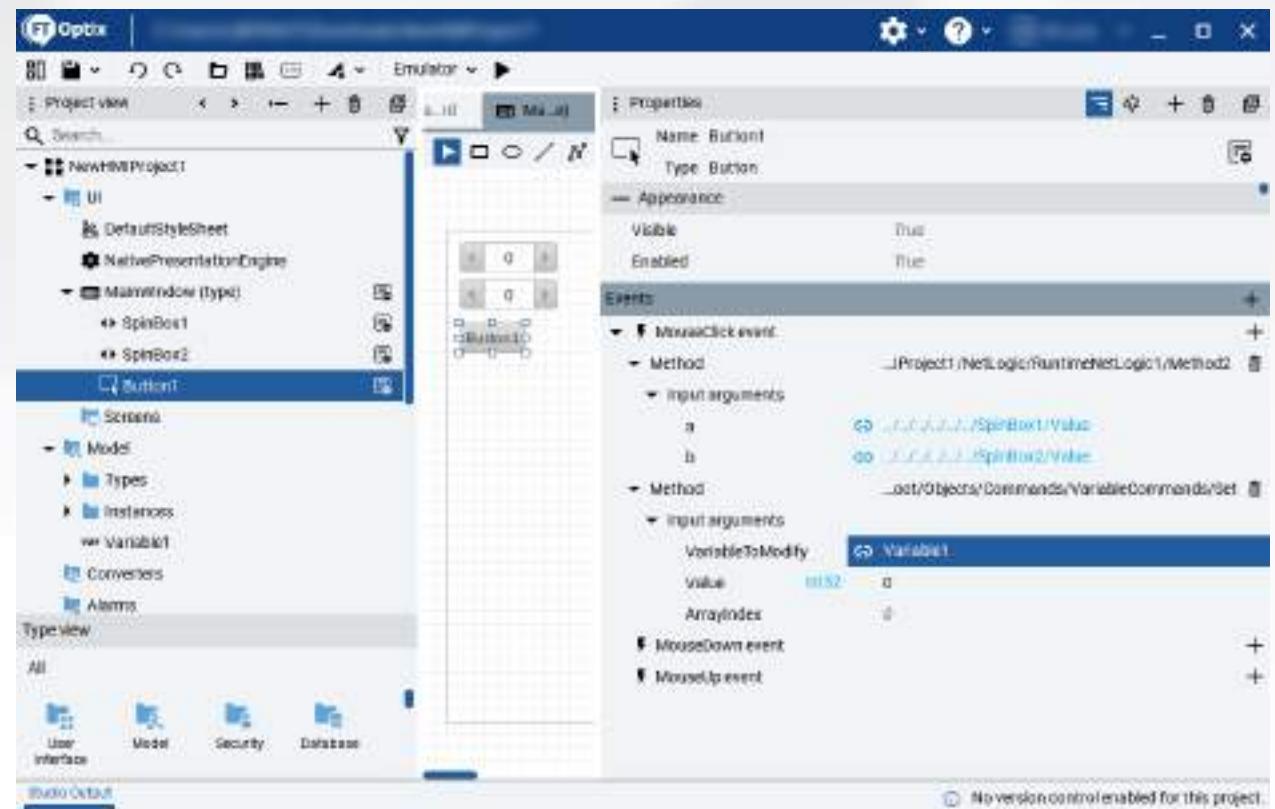
# Return parameters from NetLogic methods

- On any method that can trigger an event, call the method exposed by the NetLogic
- Configure the input parameters (if any)
- Add a second method invocation
- Use a SetVariableValue to get the output value from the MethodContainer1 and set it to a different variable (or property)



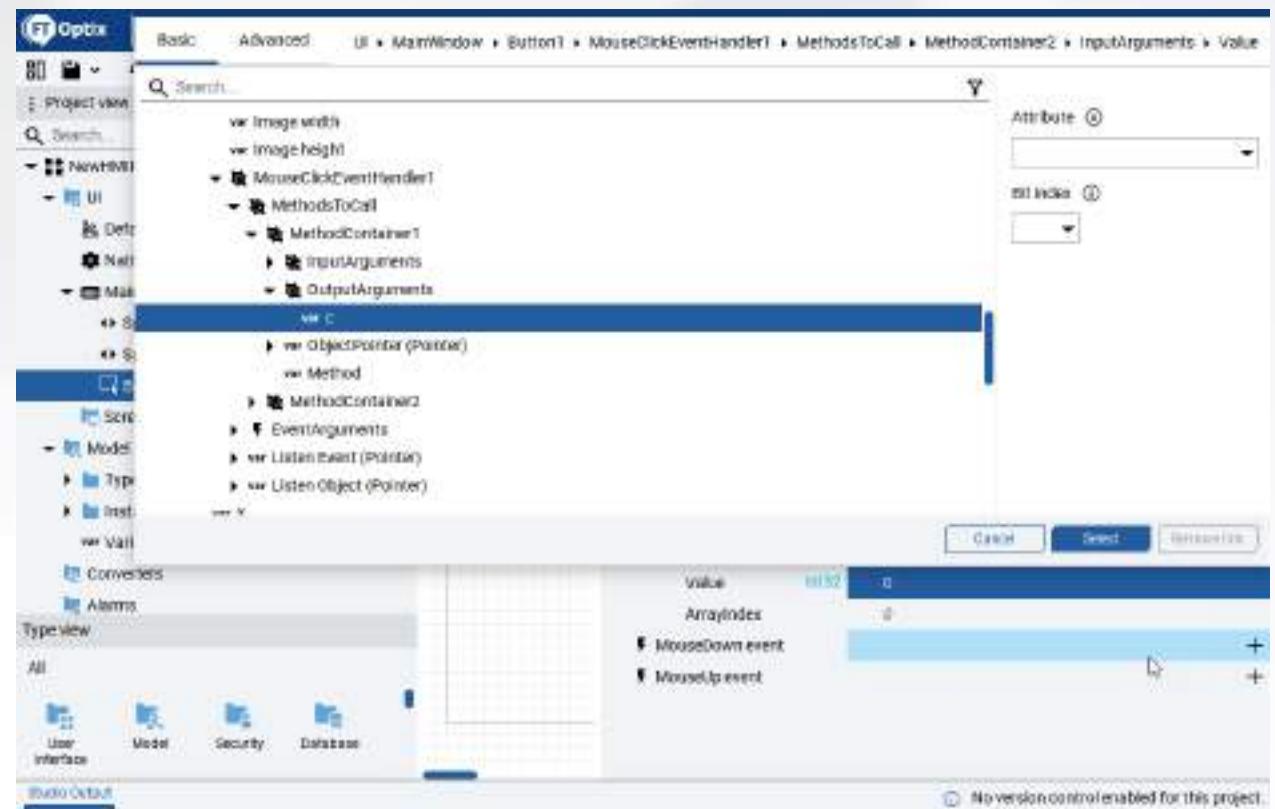
# Return parameters from NetLogic methods

- On any method that can trigger an event, call the method exposed by the NetLogic
- Configure the input parameters (if any)
- Add a second method invocation
- Use a SetVariableValue to get the output value from the MethodContainer1 and set it to a different variable (or property)



# Return parameters from NetLogic methods

- On any method that can trigger an event, call the method exposed by the NetLogic
- Configure the input parameters (if any)
- Add a second method invocation
- Use a SetVariableValue to get the output value from the MethodContainer1 and set it to a different variable (or property)



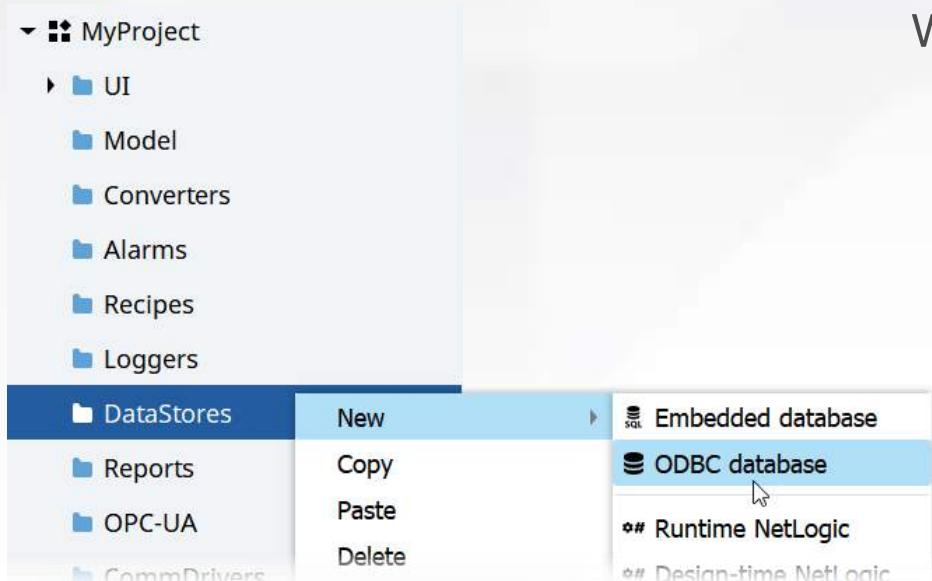
# Connect to an External Database



# ODBC Database

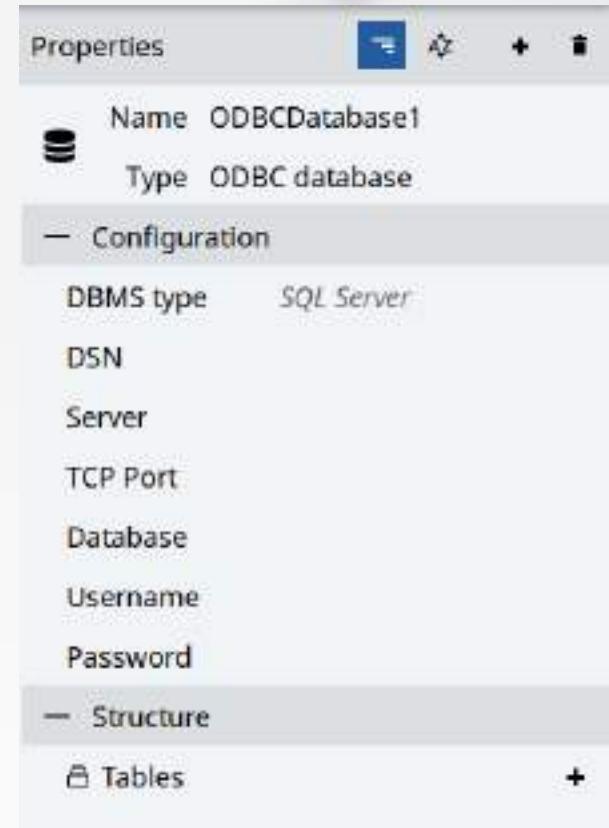
- It's used to query/populate a relational database external to the FTOptixApplication
- Supported DBMS:
  - SQL Server or SQL Server Express 2012 (or higher)
  - MySQL Server

- Objects that can exchange data with an ODBC database:
  - *Data grid, List box, Combo box and Trend objects*  
query a table to display data at runtime
  - *Data logger and Event logger objects*  
store data in a table
  - *Recipe object*  
stores data in a table and execute a query to display data at runtime



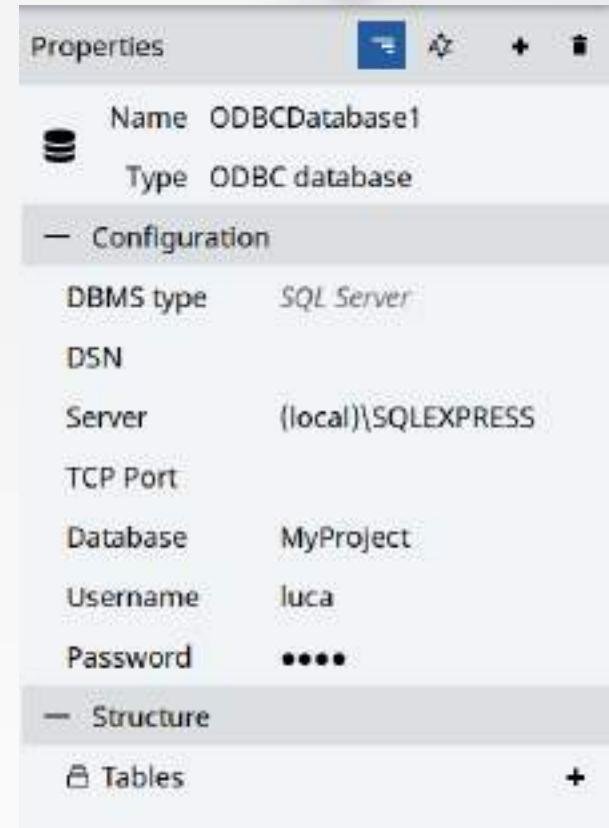
# ODBC Database

- *DBMS type*: SQL Express or MySQL
- *DSN (optional)*: only in case ODBC is used
- *Server*: can be IP Address, Hostname or "Hostname\Instance Name"
- *TCP Port (optional)*: to be used if the port is different from the default
- *Database*: Name of the database  
**NOTE: database must be created manually**
- *Username/Password*: credential of the user, used to read/write into database tables

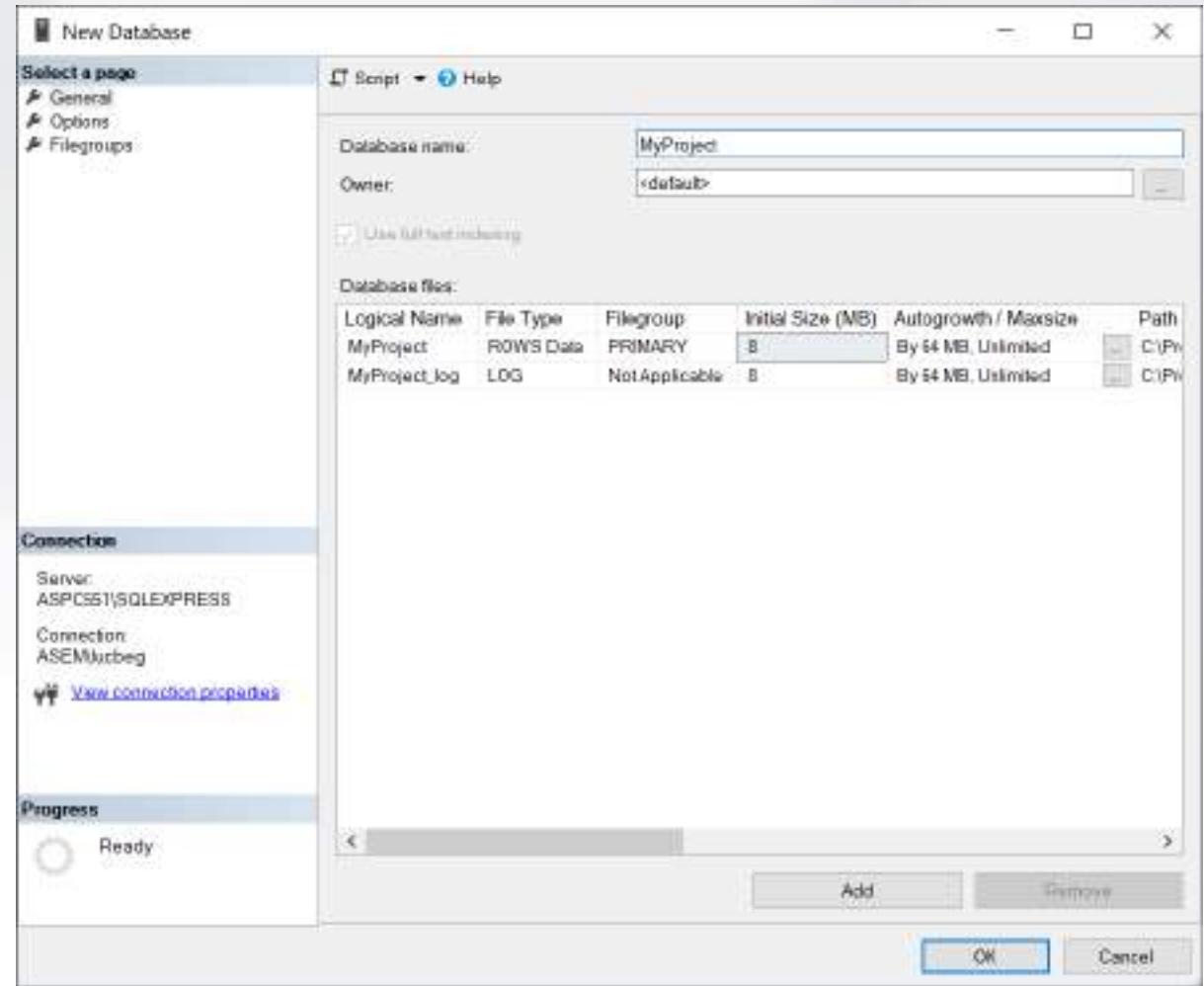
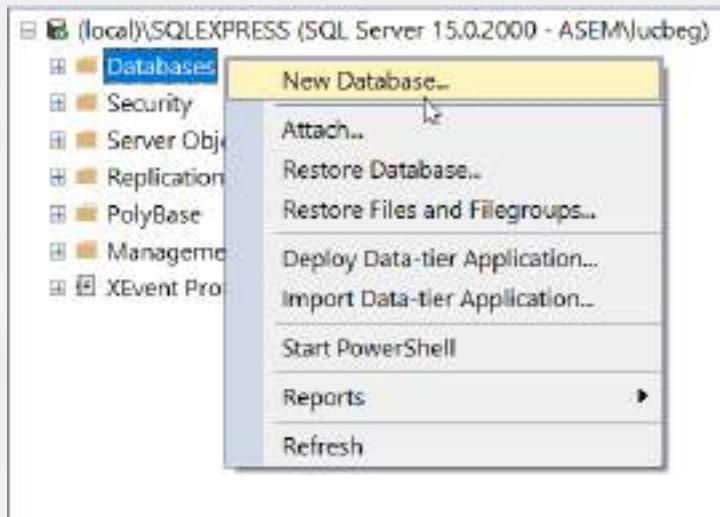


# ODBC Database

- *DBMS type*: SQL Express or MySQL
- *DSN (optional)*: only in case ODBC is used
- *Server*: can be IP Address, Hostname or "Hostname\Instance Name"
- *TCP Port (optional)*: to be used if the port is different from the default
- *Database*: Name of the database  
**NOTE: database must be created manually**
- *Username/Password*: credential of the user, used to read/write into database tables



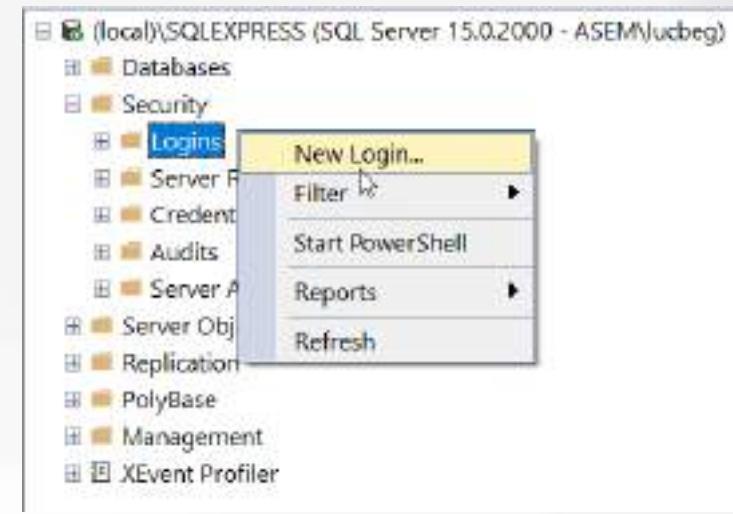
# SQL Server: create database via SQL Server Management Studio



- Is suggested to create the database before the creation of the user
- To create a database  
Databases > New Database...  
and is enough to provide the name

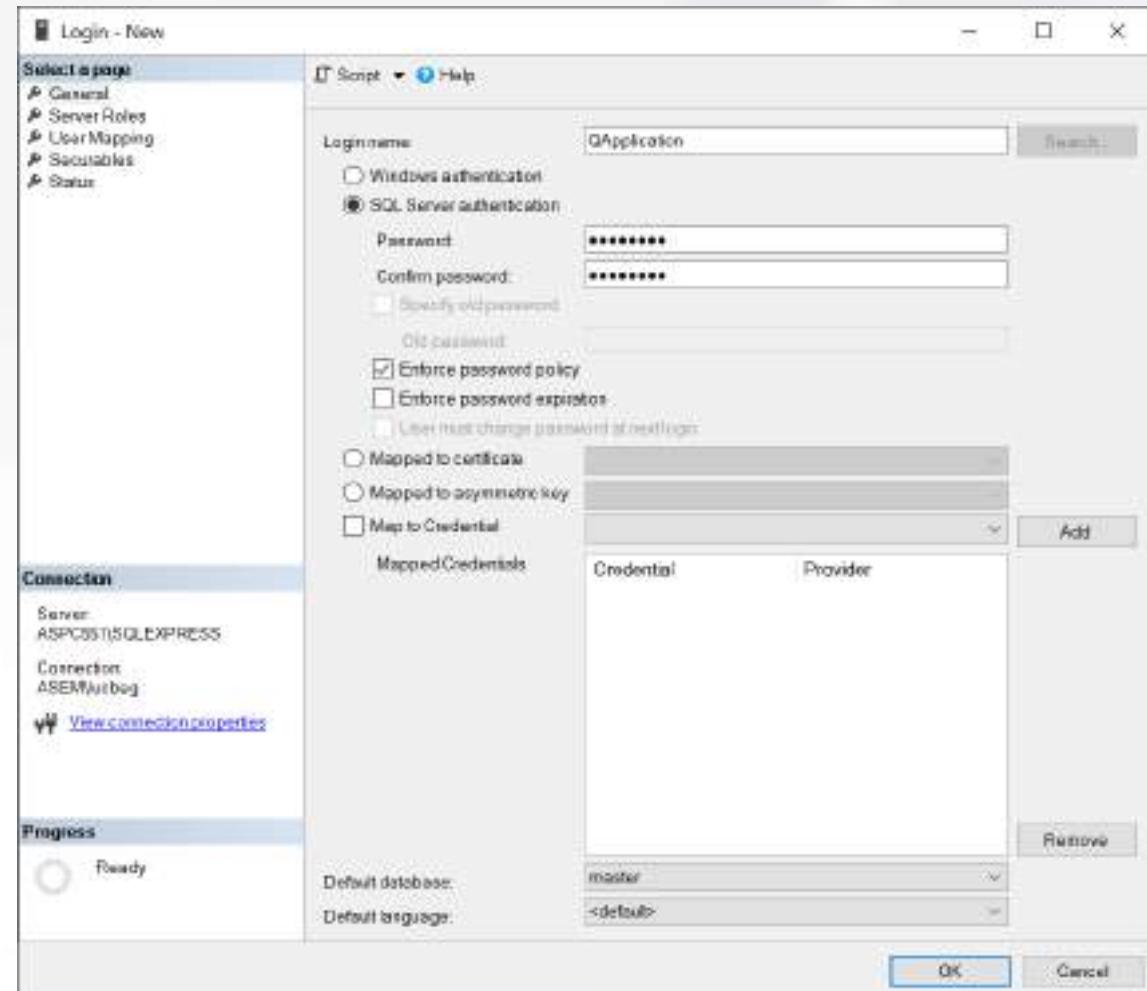
# SQL Server: create user via SQL Server Management Studio

1. Security > Logins > New Login...
2. Once created a User, it's necessary to assign a role for the needed database
3. User Mapping > Database > db\_owner



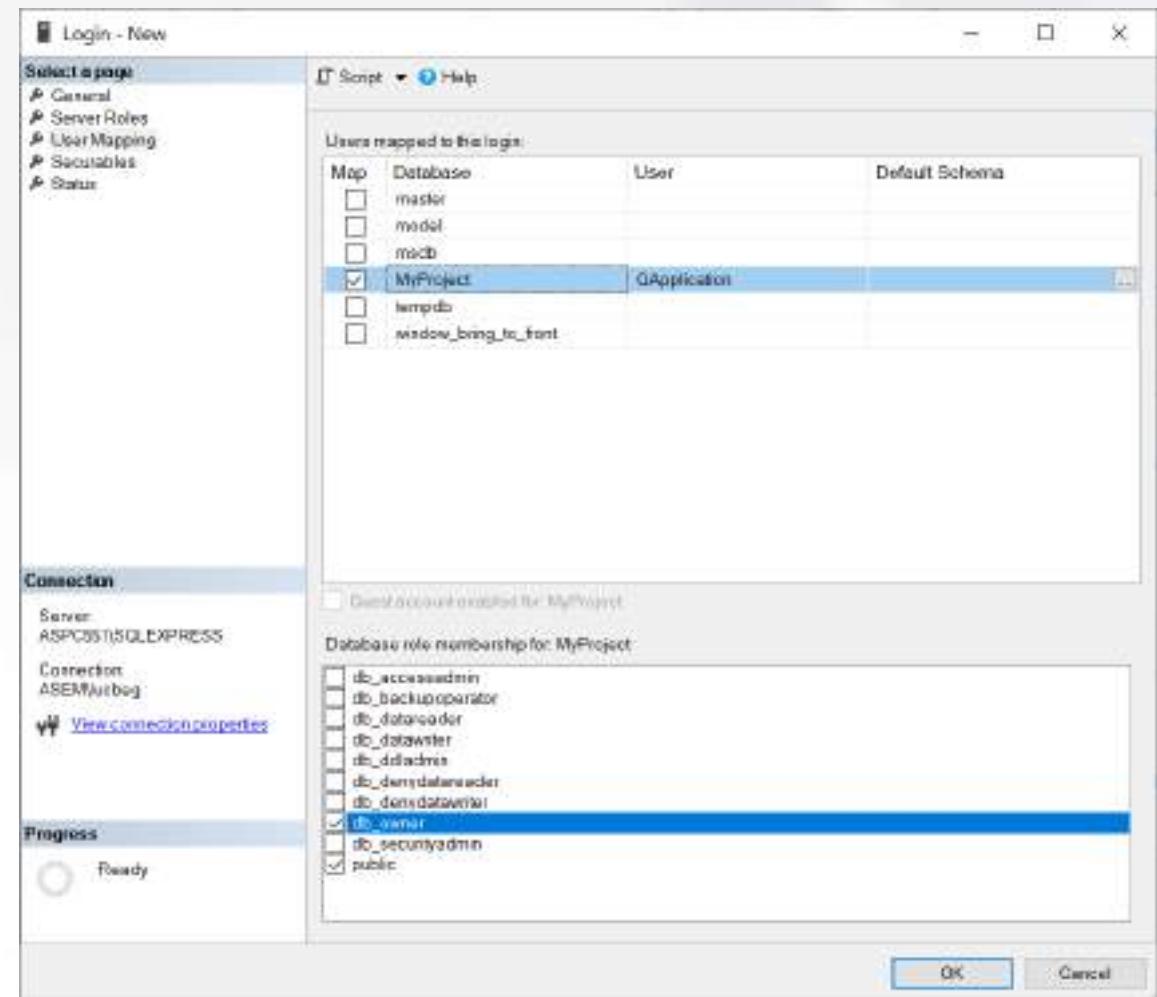
# SQL Server: create user via SQL Server Management Studio

1. Security > Logins > New Login...
2. Once created a User, it's necessary to assign a role for the needed database
3. User Mapping > Database > db\_owner



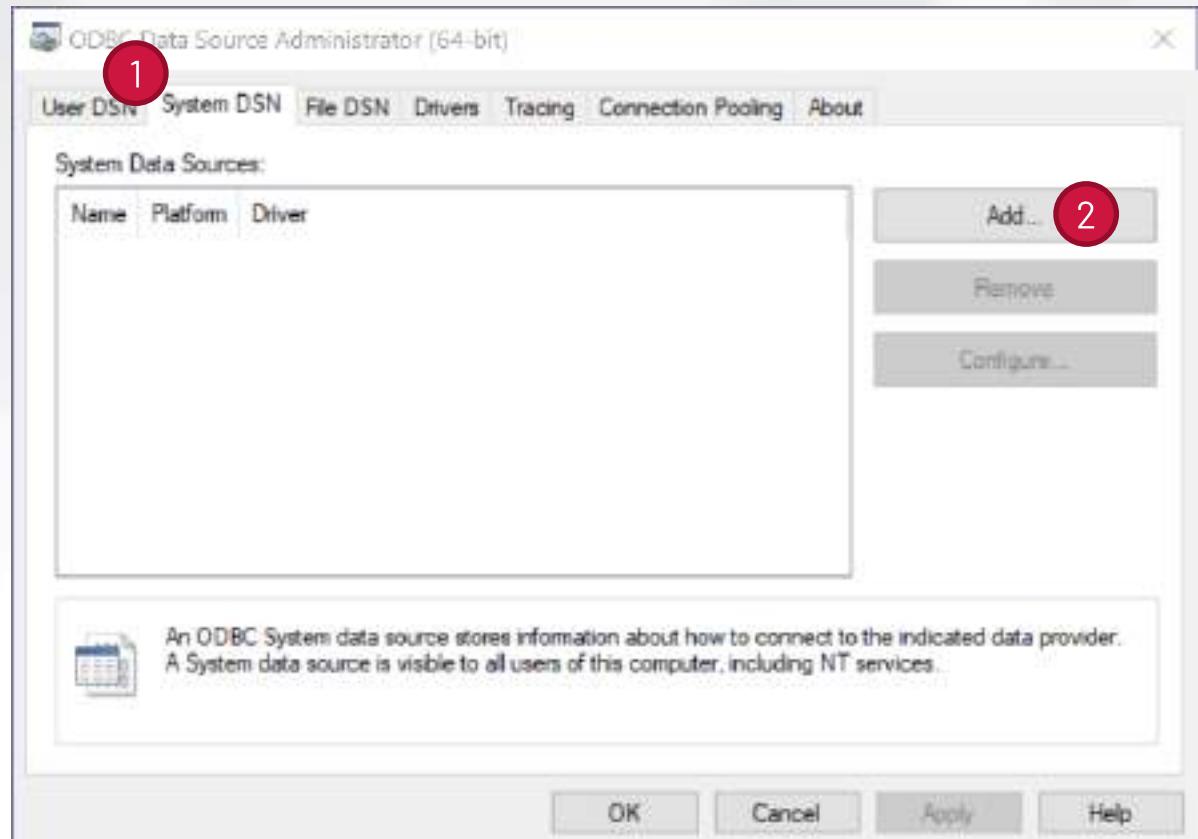
# SQL Server: create user via SQL Server Management Studio

1. Security > Logins > New Login...
2. Once created a User, it's necessary to assign a role for the needed database
3. User Mapping > Database > db\_owner



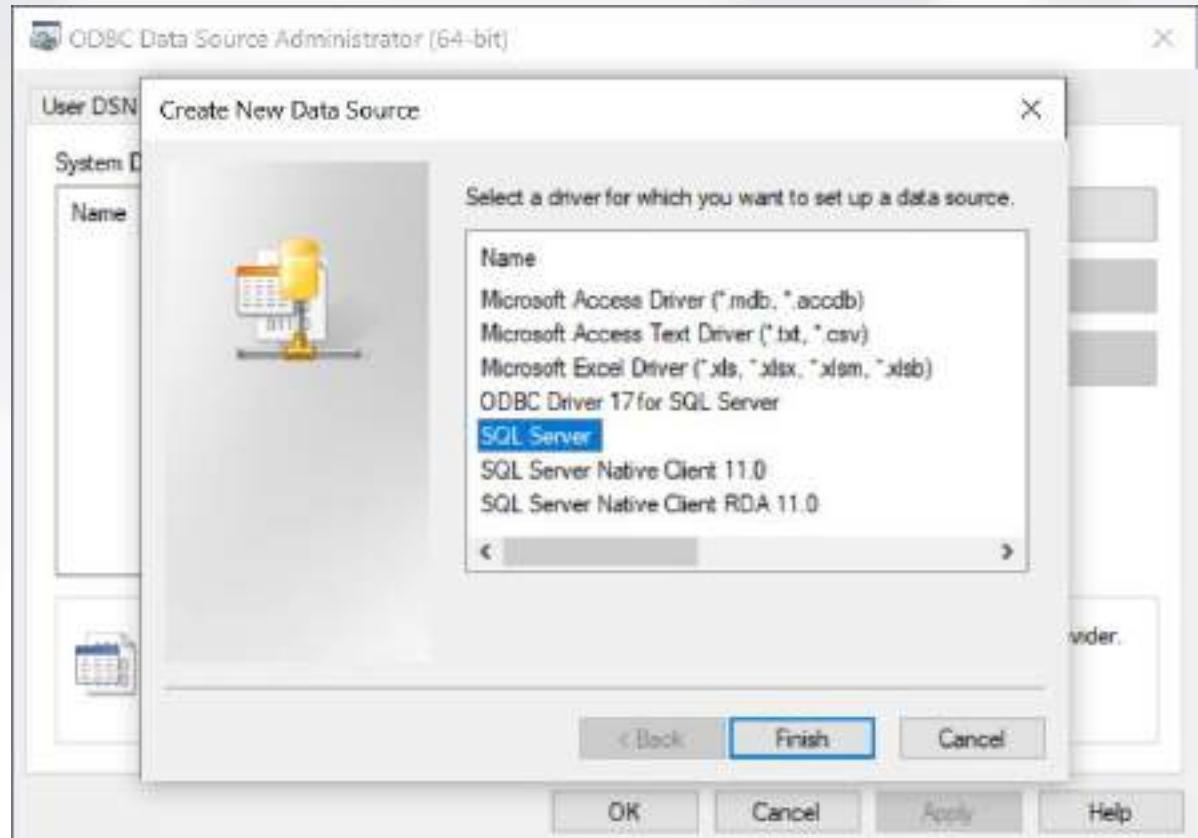
# Usage of ODBC Data Source (optional)

- To configure a DSN  
use "ODBC Data Sources (64-bit)"
- ODBC Driver supported:
  - SQL Server
  - MySQL ANSI
  - MySQL Unicode
- Key settings:
  - DSN Name: defined into ODBC DSN, must be the same in FactoryTalk Optix Studio
  - Default Database: is the Database created for the FTOptixApplication
  - Username/Password: valid credential to access the Database



# Usage of ODBC Data Source (optional)

- To configure a DSN  
use "ODBC Data Sources (64-bit)"
- ODBC Driver supported:
  - SQL Server
  - MySQL ANSI
  - MySQL Unicode
- Key settings:
  - DSN Name: defined into ODBC DSN,  
must be the same in FactoryTalk Optix Studio
  - Default Database: is the Database created  
for the FTOptixApplication
  - Username/Password: valid credential to access the Database



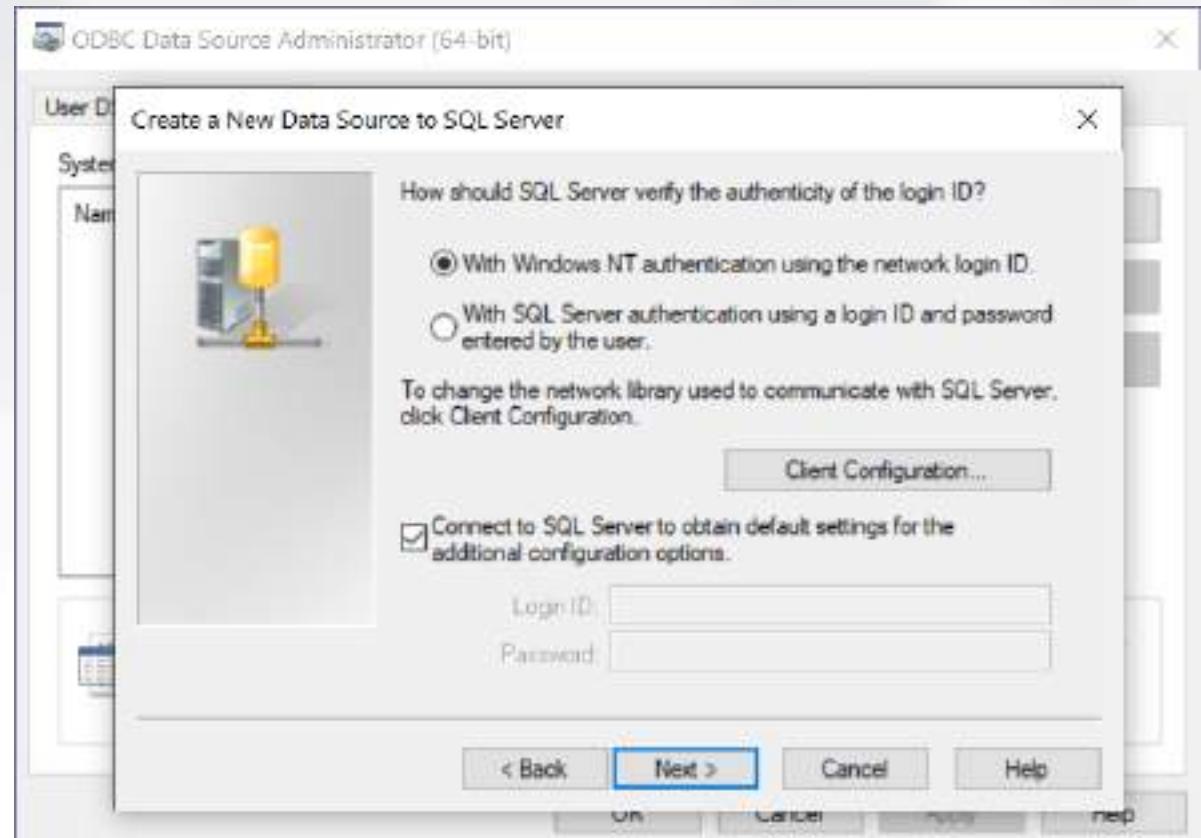
# Usage of ODBC Data Source (optional)

- To configure a DSN  
use "ODBC Data Sources (64-bit)"
- ODBC Driver supported:
  - SQL Server
  - MySQL ANSI
  - MySQL Unicode
- Key settings:
  - DSN Name: defined into ODBC DSN,  
must be the same in FactoryTalk Optix Studio
  - Default Database: is the Database created  
for the FTOptixApplication
  - Username/Password: valid credential to access the Database



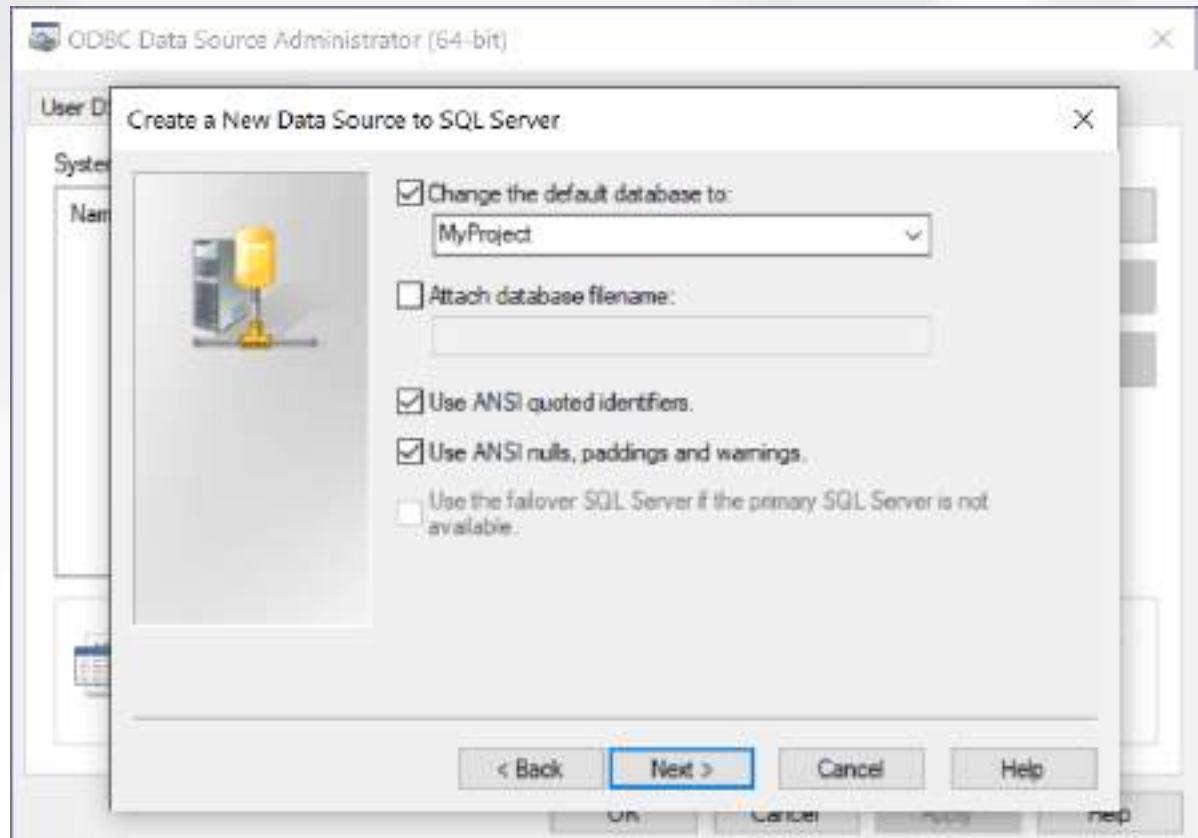
# Usage of ODBC Data Source (optional)

- To configure a DSN  
use "ODBC Data Sources (64-bit)"
- ODBC Driver supported:
  - SQL Server
  - MySQL ANSI
  - MySQL Unicode
- Key settings:
  - DSN Name: defined into ODBC DSN,  
must be the same in FactoryTalk Optix Studio
  - Default Database: is the Database created  
for the FTOptixApplication
  - Username/Password: valid credential to access the Database



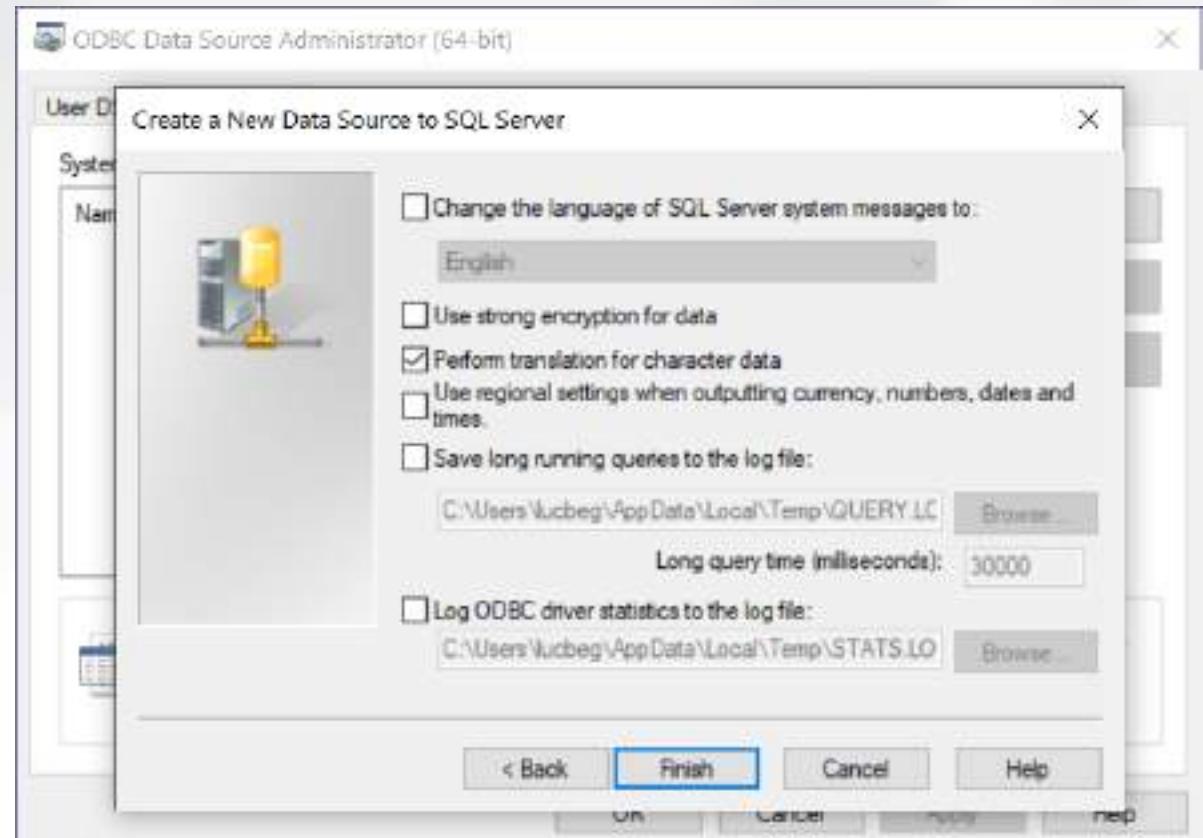
# Usage of ODBC Data Source (optional)

- To configure a DSN  
use "ODBC Data Sources (64-bit)"
- ODBC Driver supported:
  - SQL Server
  - MySQL ANSI
  - MySQL Unicode
- Key settings:
  - DSN Name: defined into ODBC DSN,  
must be the same in FactoryTalk Optix Studio
  - Default Database: is the Database created  
for the FTOptixApplication
  - Username/Password: valid credential to access the Database



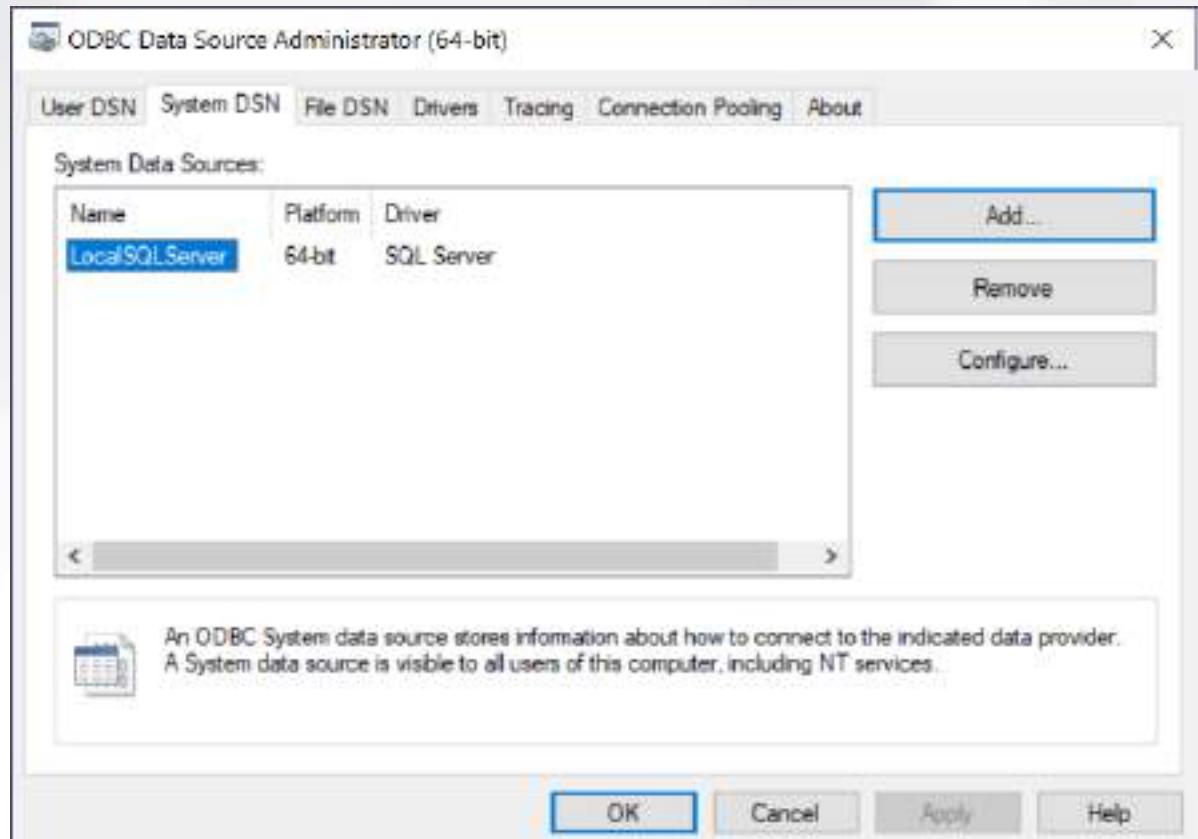
# Usage of ODBC Data Source (optional)

- To configure a DSN  
use "ODBC Data Sources (64-bit)"
- ODBC Driver supported:
  - SQL Server
  - MySQL ANSI
  - MySQL Unicode
- Key settings:
  - DSN Name: defined into ODBC DSN,  
must be the same in FactoryTalk Optix Studio
  - Default Database: is the Database created  
for the FTOptixApplication
  - Username/Password: valid credential to access the Database



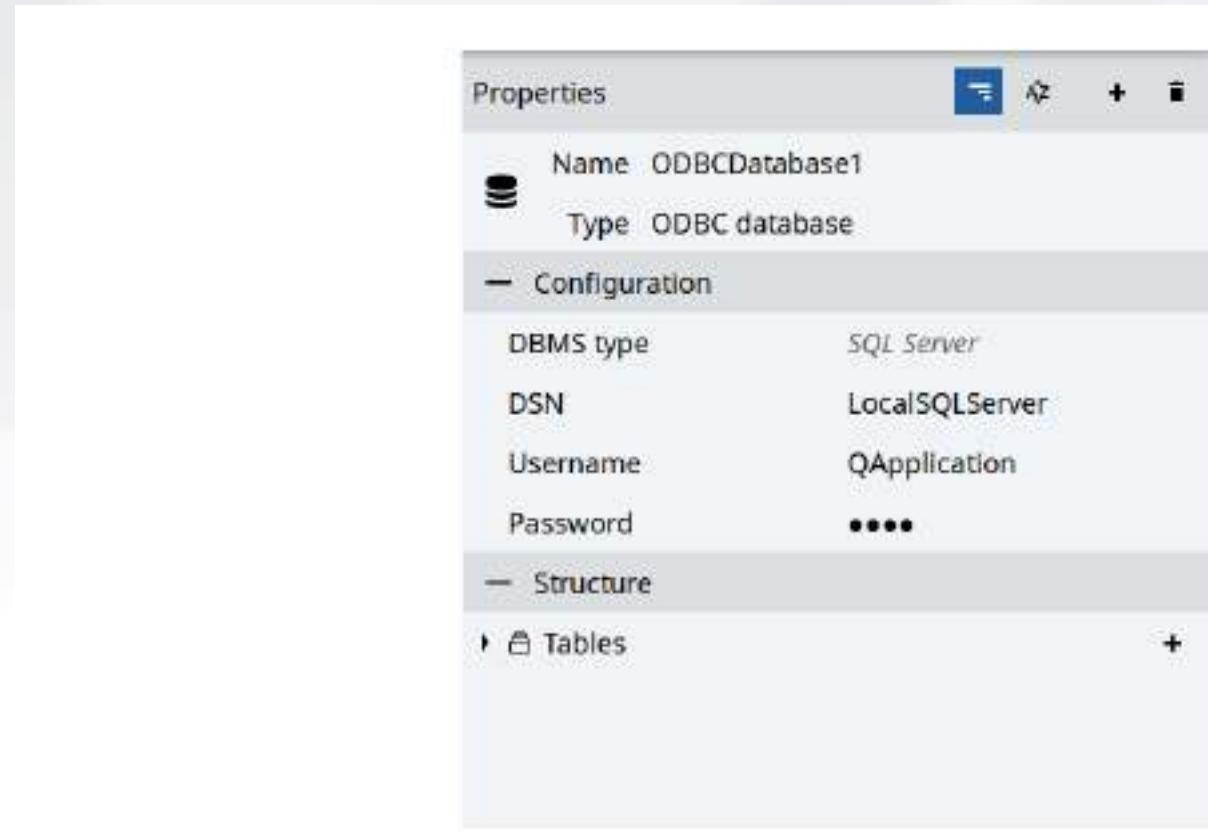
# Usage of ODBC Data Source (optional)

- To configure a DSN  
use "ODBC Data Sources (64-bit)"
- ODBC Driver supported:
  - SQL Server
  - MySQL ANSI
  - MySQL Unicode
- Key settings:
  - DSN Name: defined into ODBC DSN, must be the same in FactoryTalk Optix Studio
  - Default Database: is the Database created for the FTOptixApplication
  - Username/Password: valid credential to access the Database



# Usage of ODBC Data Source (optional)

- To configure a DSN  
use "ODBC Data Sources (64-bit)"
- ODBC Driver supported:
  - SQL Server
  - MySQL ANSI
  - MySQL Unicode
- Key settings:
  - DSN Name: defined into ODBC DSN,  
must be the same in FactoryTalk Optix Studio
  - Default Database: is the Database created  
for the FTOptixApplication
  - Username/Password: valid credential to access the Database



A large, abstract graphic consisting of a grid of blue dots arranged in a wave-like, undulating pattern across the entire background of the slide.

# Connect to an External InfluxDB instance



# Sample InfluxDB server using Docker

services:

influxdb:

```
  image: influxdb:latest
  container_name: influxdb
  ports:
    - '8086:8086' # Port to set in the Optix project
```

volumes:

```
  - influxdb-storage:/var/lib/influxdb2
```

restart: unless-stopped

environment:

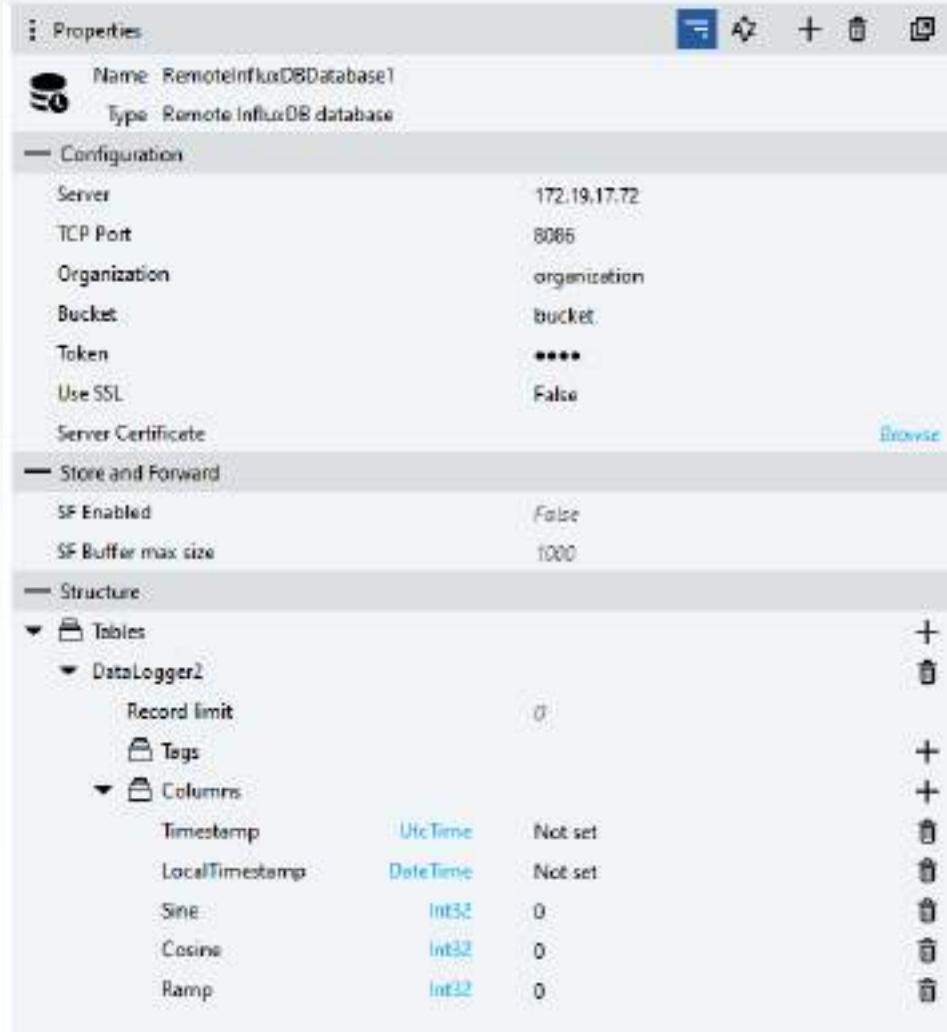
```
- DOCKER_INFLUXDB_INIT_MODE=setup
- DOCKER_INFLUXDB_INIT_USERNAME=admin
- DOCKER_INFLUXDB_INIT_PASSWORD=admin_pass
- DOCKER_INFLUXDB_INIT_ORG=organization # Name of the organization to set in the Optix project
- DOCKER_INFLUXDB_INIT_BUCKET=bucket # Name of the bucket to set in the Optix project
- DOCKER_INFLUXDB_INIT_RETENTION=4w # How long to retain data
- DOCKER_INFLUXDB_INIT_ADMIN_TOKEN=MyVeryLongAndS3cr3tTok3nToWriteD4t4 # Access token to set in the Optix project
```

volumes:

influxdb-storage:

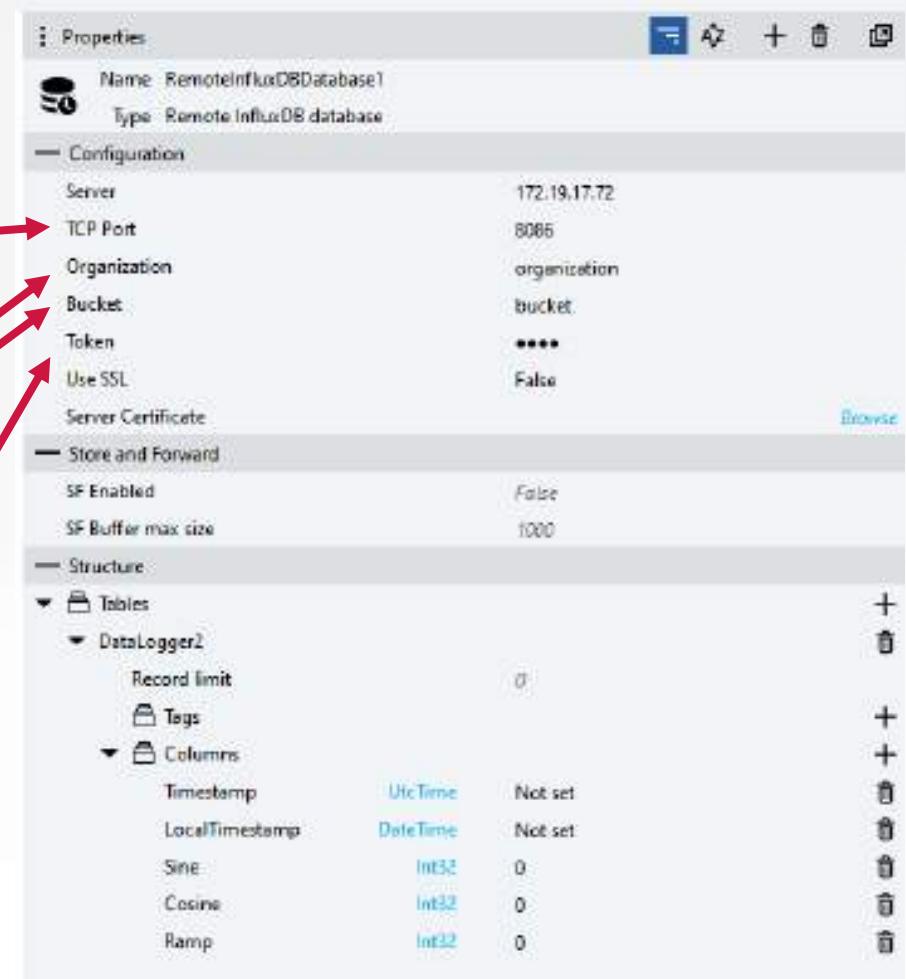
# Remote InfluxDB configuration

- Configure the connection parameters to reach the InfluxDB instance
- Usage of SSL is recommended
  - Certificate can be exported from the InfluxDB server and loaded in the Optix application
- Tables and columns are only required if inserting new data (same as ODBC)

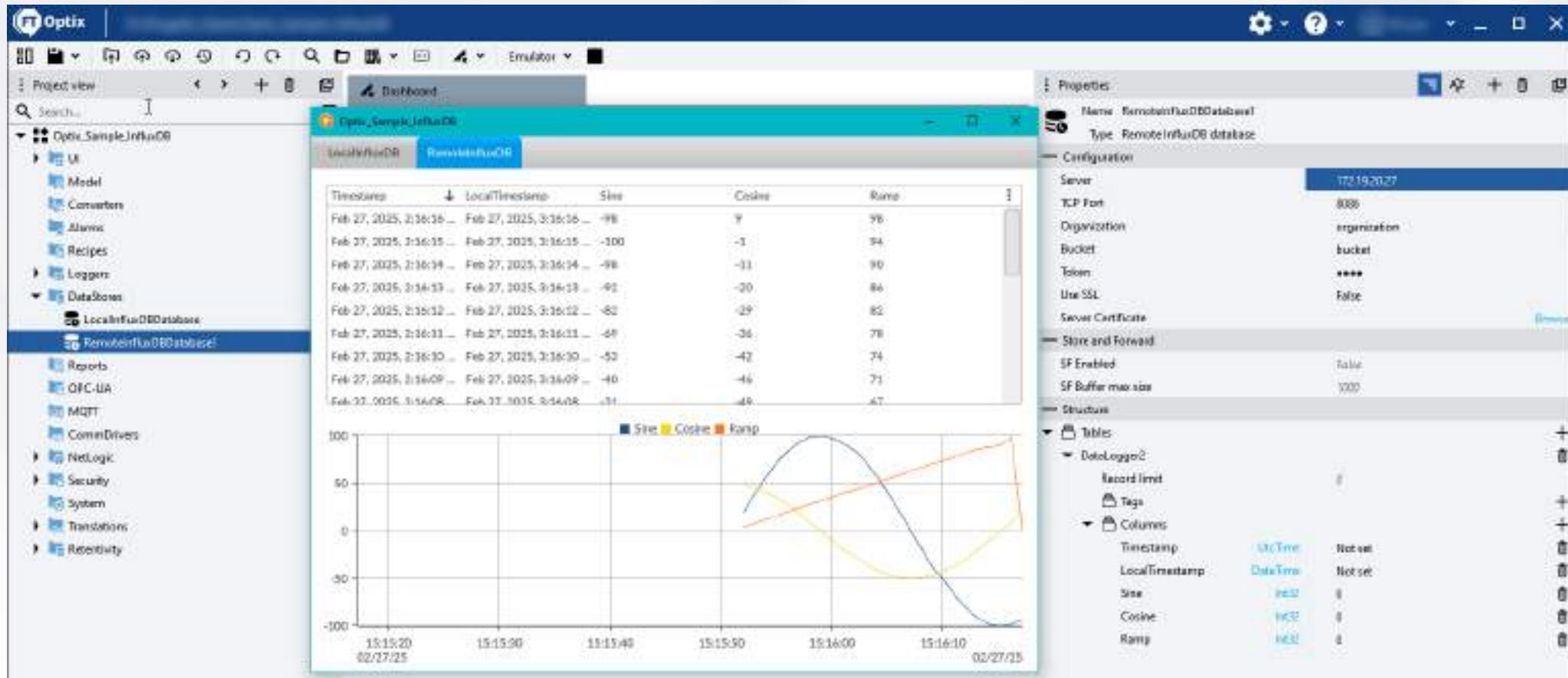


# Configuration example

```
services:  
  influxdb:  
    image: influxdb:latest  
    container_name: influxdb  
    ports:  
      - '8086:8086'  
    volumes:  
      - influxdb-storage:/var/lib/influxdb2  
    restart: unless-stopped  
    environment:  
      - DOCKER_INFLUXDB_INIT_MODE=setup  
      - DOCKER_INFLUXDB_INIT_USERNAME=admin  
      - DOCKER_INFLUXDB_INIT_PASSWORD=admin_pass  
      - DOCKER_INFLUXDB_INIT_ORG=organization  
      - DOCKER_INFLUXDB_INIT_BUCKET=bucket  
      - DOCKER_INFLUXDB_INIT_RETENTION=4w  
      - DOCKER_INFLUXDB_INIT_ADMIN_TOKEN=MyVeryLongAndS3cr3tTok3nToWriteD4t4  
volumes:  
  influxdb-storage:
```



# Configuration example

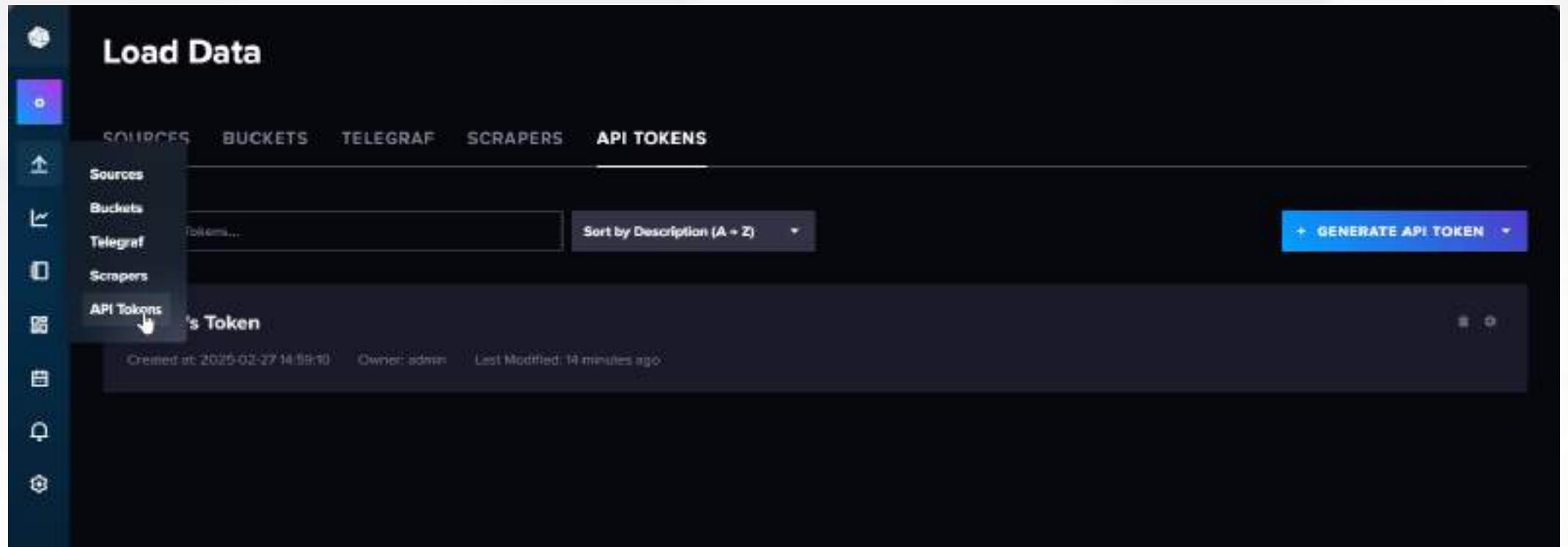


# InfluxDB limitations

- No support for Flux query language
- Limited set of queries (see InfluxDB documentation in Optix help)
- Recipes cannot be used with an InfluxDB store

# Creating a new token

- Access tokens are created from the InfluxDB web interface



# Creating or configuring a bucket

- Buckets, retention and user access are configured using the InfluxDB web interface

The screenshot shows the InfluxDB web interface with the 'BUCKETS' tab selected in the top navigation bar. The main area displays three buckets: 'bucket', '\_monitoring', and '\_tasks'. Each bucket entry includes its name, type (e.g., 'System Bucket'), retention policy (e.g., 'Retention: 30 days'), ID, and a small icon. To the right of the bucket list is a 'CREATE BUCKET' button and a sidebar titled 'What is a Bucket?'. The sidebar defines a bucket as a named location for time-series data and explains retention policies.

Bucket	Type	Retention	ID
bucket	System Bucket	Retention: 30 days	ID: influxdb@0000000000000000
_monitoring	System Bucket	Retention: 7 days	ID: 0000000000000000
_tasks	System Bucket	Retention: 3 days	ID: 07ab6975e68c2970

# Additional resources



# FactoryTalk Optix GitHub organization

- Contains a big list of projects with a lot of features that can be studied and reused in custom projects
- Contains a lot of sample projects
- Contains a NetLogic CheatSheet with a lot of useful snippets

The screenshot shows the GitHub organization page for 'FactoryTalk-Optix'. The page includes the organization's logo, a brief description ('expanding human possibility™'), and links to their website and email. Below this, the README.md file is displayed, featuring a large heading 'Welcome to FactoryTalk® Optix™ organization!' and several paragraphs of descriptive text about the platform's features and benefits.

README.md

## Welcome to FactoryTalk® Optix™ organization!

Welcome to [FactoryTalk® Optix™](#), a new visualization platform that accelerates value delivery with modern technologies, innovative designs, and scalable deployment options.

[FactoryTalk® Optix™](#) can help improve your process, efficiency, and deliverables – in one easy-to-access tool. Take advantage of new levels of collaboration, scalability, and interoperability to achieve your HMI vision.

New SaaS-enabled workflows will enable your team to collaborate any time, from anywhere, thanks to built-in change tracking and versioning that automatically keeps track of who did what and when.



# FactoryTalk Optix landing page

- Contains all the FactoryTalk Optix related resources (help, install guide, release notes, etc)
- Constantly updated

The screenshot shows the homepage of the FactoryTalk Optix Portfolio Technical Documentation. At the top, there's a navigation bar with the Rockwell Automation logo and links for Products, Services, Solutions & Industries, Support, and Sales & Partners. Below the navigation, the title "FactoryTalk Optix Portfolio Technical Documentation" is displayed. A large, abstract graphic of overlapping red and blue triangles is on the right. Below the title, there are several menu items: FactoryTalk Optix Portfolio, Home > Review FactoryTalk Optix Portfolio, System Architectures, and How to Create a Project. A "SIGN-IN/CREATE FREE" button is located at the bottom right of the main content area. The main content area contains text about the FactoryTalk Optix portfolio and two bulleted lists under the heading "New to FactoryTalk Optix?".

The FactoryTalk® Optix™ portfolio combines cloud-based software and hardware devices to offer a flexible solution built for openness and extensibility:

- Cloud-based software enables you to create innovative designs and scale through seamless access to physically distant sites.
- Flexible target devices include ControlLogix® Embedded Edge Compute modules, OptixPanel™ graphic terminals, and ABEM™ 8300 industrial PCs.

New to FactoryTalk Optix?

Start with these resources to help you select what you need:

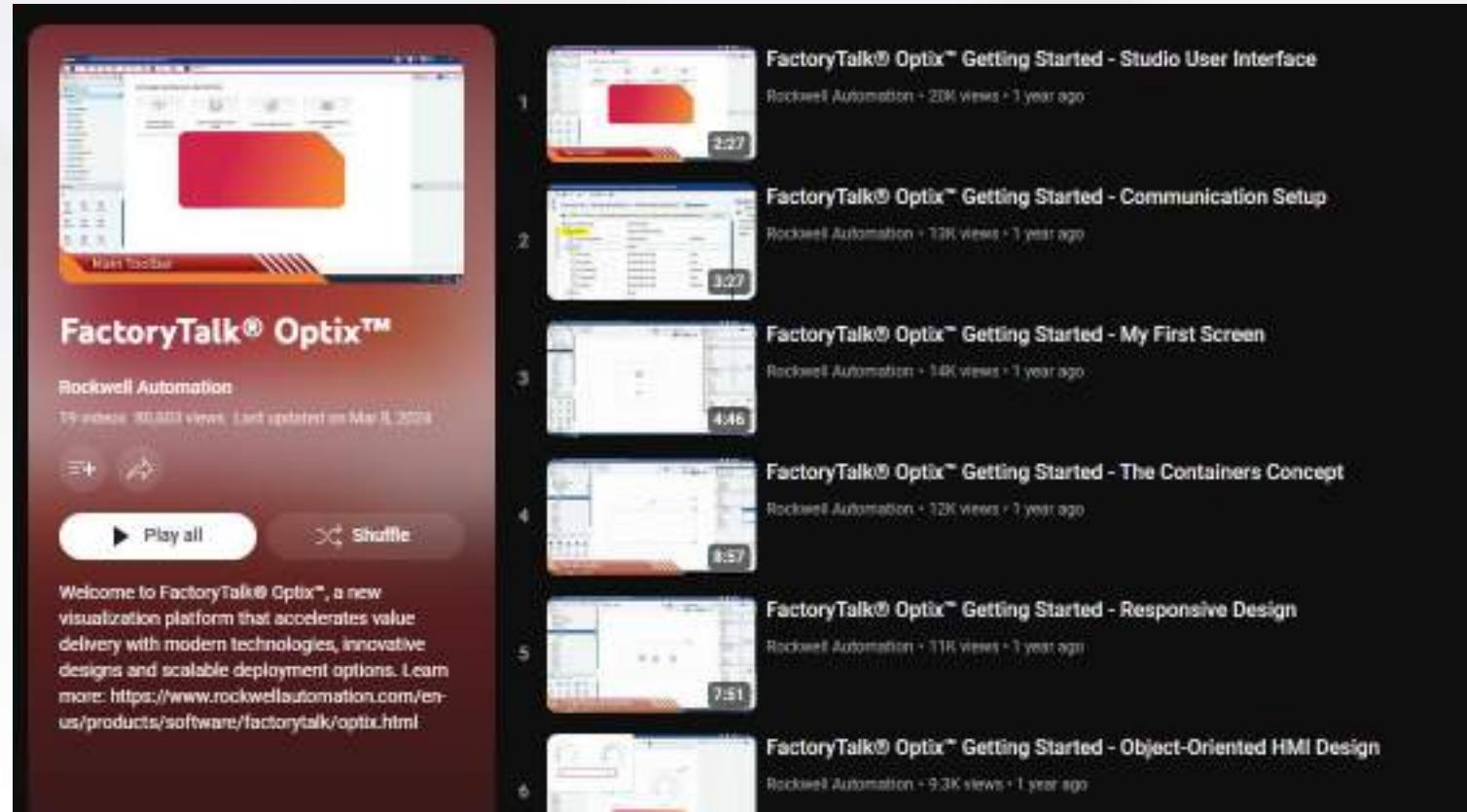
- FactoryTalk Optix Installation Guide
- FactoryTalk Optix Demos: Control valves
- FactoryTalk Optix Solutions Application Techniques Update 9 or 10.0.3



[FactoryTalk Optix Portfolio Technical Documentation | Rockwell Automation](#)

# FactoryTalk Optix YouTube channel

- Contains videos on how to build your first project (check the playlists tab)



[Rockwell Automation - YouTube](#)

# FactoryTalk Optix forum

- Used by customers and colleagues to exchange informations

The screenshot shows the search results for 'FactoryTalk' on the Engage community platform. The search bar at the top contains the query 'FactoryTalk'. Below the search bar, there are several filters: 'All Content' (selected), 'Communities' (1 result), and 'Discussion Threads' (707 results). The results are listed in a grid format. The first result is titled 'ETOptix - String Tag not logged in RemoteInfluxDB' and has three replies. The second result is titled 'ET Optix not logging PLC events from TwinCAT OPC server - Is this a bug or missing feature?' and also has three replies. The third result is titled 'ETOptix - AuditSigning - Change WorkflowType in NetLogic script(CR)' and has four replies. Each result includes a small profile picture of the poster, the number of replies, and a 'View Details' link.



[Search - Engage, A Rockwell Automation Community](#)

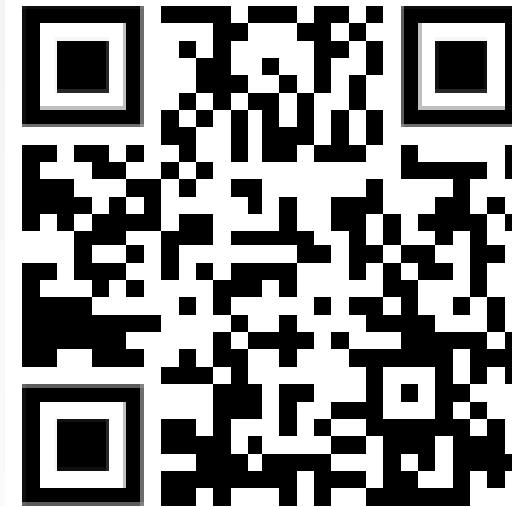
# FactoryTalk Hub

- Contains all the releases of FactoryTalk Optix
  - Contains additional resources like RuntimeTools
  - Used to access the WebIDE
- Contains some interactive demos

The screenshot shows the FactoryTalk Hub interface. On the left, there's a sidebar with 'Resources', 'Skills', and 'Machine Tools'. Below it, a section titled 'FactoryTalk Optix Release' lists several releases with download links. On the right, a larger area is titled 'Resources' and shows a grid of cards with titles like 'Optix 10.0.0.0', 'Optix 10.0.0.1', 'Optix 10.0.0.2', and 'Optix 10.0.0.3'. Each card has a preview image, a title, a date, a description, and two buttons: 'Open in Browser' and 'Download'.

This screenshot shows a detailed view of the 'Resources' section in the FactoryTalk Hub. It displays a table with columns for 'Thumbnail', 'Title', 'Date', and 'Description'. The table contains four rows corresponding to the releases shown in the previous screenshot. Each row includes a 'View in Browser' button and a 'Download' button.

Thumbnail	Title	Date	Description
	Optix 10.0.0.0	2023-06-01	This initial release for the latest integrated version of the FactoryTalk Optix application. It includes the core functionality of the application.
	Optix 10.0.0.1	2023-06-01	This update contains most of the features required for a full Optix code.
	Optix 10.0.0.2	2023-06-01	Some improvements have been made to the user interface and performance.
	Optix 10.0.0.3	2023-06-01	Some improvements have been made to the user interface and performance.



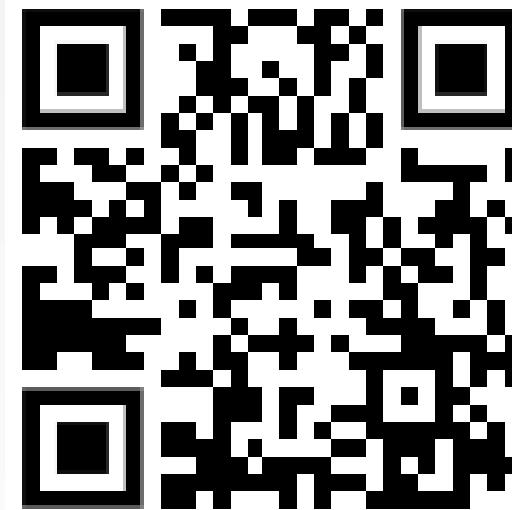
[FactoryTalk Hub](#)

# FactoryTalk Hub

- Used to view, manage and activate FactoryTalk Optix licenses
- Contains some useful tools
  - Runtime License size calculator (which license is needed to run the project)
  - Device sizing calculator (which device is recommended to run the project)

The screenshot shows the 'Calculate runtime size' interface. At the top, it says 'Your current entitlement is at 7 tokens. Your recommended package is Standard.' Below this, there's a summary table with columns for 'Entitlement', 'Quantity', and 'Tokens'. The table includes rows for 'Standard', 'Advanced', and 'Enterprise'. A large button labeled 'Calculate' is at the bottom right.

The screenshot shows the 'Calculate your device size' interface. It has a similar layout to the first one, with a summary table at the top and a 'Calculate' button at the bottom right.



[FactoryTalk Hub](#)



THANK YOU!