Smart People. Expert Solutions.

SIEMENS OPEN LIBRARY

3 – Example Object Configuration

NOVEMBER 3, 2017

Contents

1. Purpose

The purpose of this document is to walk through the steps required to setup an object in the Siemens Open Library. Each object has a different interface, as defined in the detailed block description for that object, however, all of the objects are setup and configured using the methods outlined in this document.

2. Intended Use

This document is intended to be used by anyone utilizing the Open Library for PLC and HMI Development. This document should be used after reviewing the following documents:

1. 1- Siemens Open Library – Library Overview and Architecture
2. 2- Siemens Open Library – Initial Setup

3. Revision History

Version	Date	Author	Comments
1.0	2016-05-23	DMC	Initial Release
1.1	2016-06-20	DMC	No Changes
1.2	2016-08-23	DMC	No Changes
1.3	2016-10-11	DMC	No Changes
2.0	2017-11-03	DMC	Added reference to SiVArc use for HMI Development.

4. Open Library License

Copyright (c) 2016 DMC, Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

5. Hardware and Software Compatibility

This library was developed in TIA Portal V13 SP1. It has been tested on the S7-1200 and S7-1500 platforms, and untested modifications have been made for compatibility with S7-300 and S7-400. The PLC objects can be used with any HMI, however, the configuration of the faceplates is only available using a Comfort Panel or WinCC Advanced.

6. Example Setup

This tutorial walks through the process of using one of the blocks from the Open Library. The example object is the Solenoid Valve, but the same method is utilized for any other object block, with special consideration and separate document for use of the PID Interface block. For detailed information for each individual object, see the documentation pertaining to that object in '4- Siemens Open Library – Detailed Block Overview.'

6.1. Initial Setup

Before starting this document please make sure you have set up your project following the steps in '2- Siemens Open Library - Initial Setup.' Each library object block is dependent on global constants and clock memory bits and will not compile without correctly completing the initial setup. The following steps need to be performed:

1. Enable system and clock memory bytes on the CPU.
2. Retrieve the Open Library.
3. Pull the Open Library PLC tags into the project.
4. Setup Mode Control, or understand how Open Library Modes function. Additionally information can be found in '1- Siemens Open Library - Library Overview and Architecture'

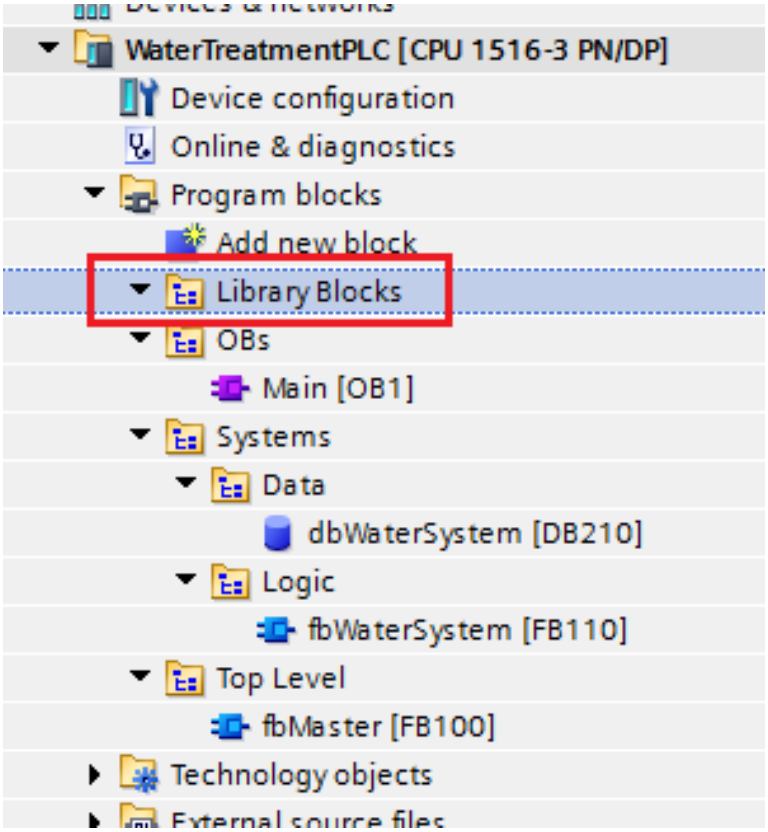
6.2. PLC

This section covers the steps required to configure the PLC project. It utilizes the best practices standard with this library, but different methods can be used to store the Function Block Instance memory and the User Defined Types.

6.2.1. Adding PLC Library Objects to Project

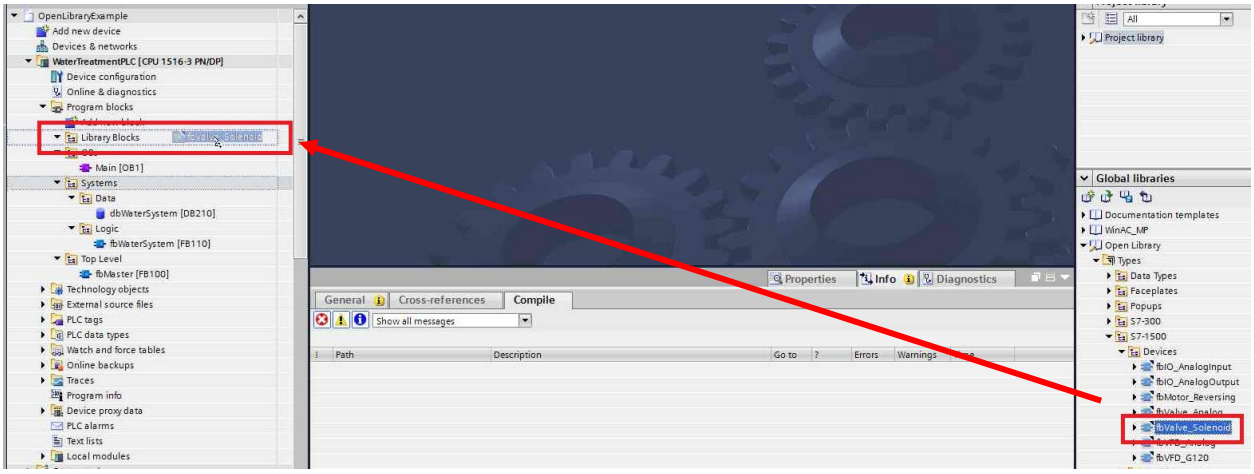
This section walks through the required steps to add an individual library object to a project. The entire PLC library can be added to the project by dragging and dropping the entire folder into the project Program Blocks. The following steps will demonstrate adding fbValve_Solenoid to the project.

1. In the controller's Program blocks, we have created example groups and blocks used in this tutorial, as seen below. A similar structure is suggested for your project. Notice the Library Blocks group is empty now because this is where the library blocks will be placed.

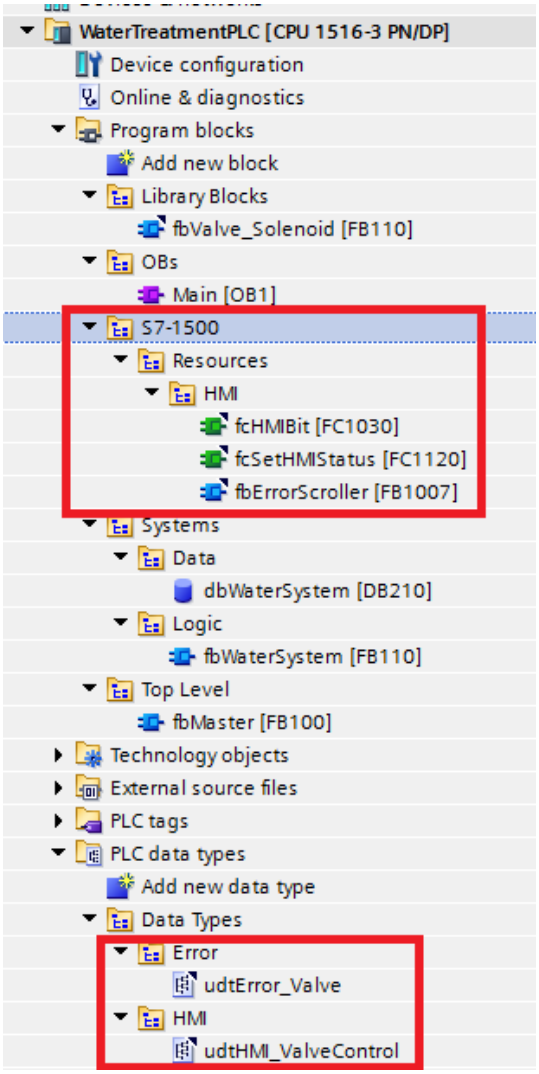


2. Copy the desired Function Block from the Open Library Types group of the library into the Library Blocks group created in our project. This is done by dragging and dropping from

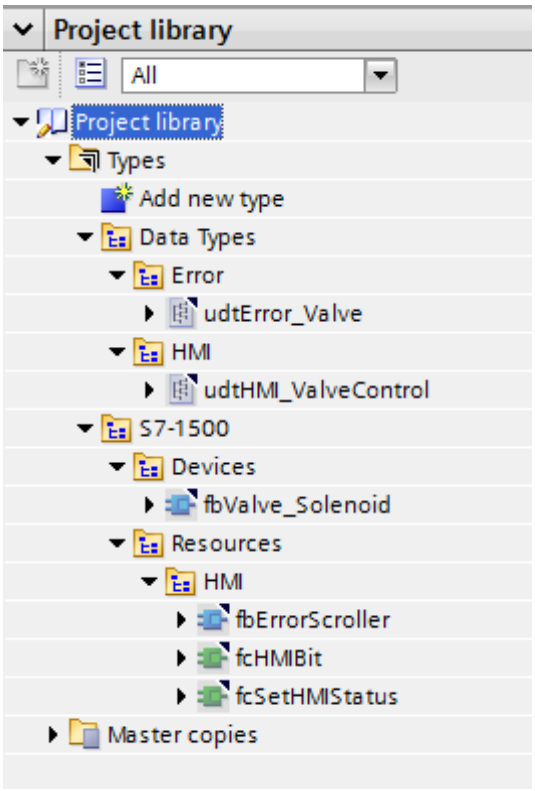
the library into the Program Blocks of your PLC. In this case we will demonstrate fbValve_Solenoid. This will need to be done for each PLC in the project that uses this block.



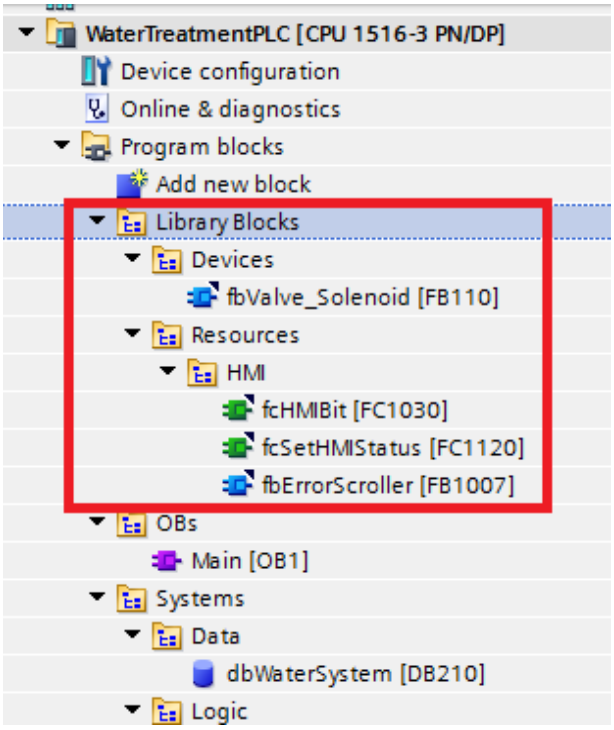
3. Notice that several other function calls and function blocks were added automatically into our project. This happens because fbValve_Solenoid is dependent on fcHMIBit, fcSetHMIStatus, fbErrorScroller, udtError_Valve, and udtHMI_ValveControl. When pulling in a library block, it will automatically pull in all required block dependencies and user defined types. The auxiliary blocks will also be automatically added to the project based on the folder structure of the Open Library.



4. The library blocks are added to the Project library. Any library blocks that need to be changed for a particular project can be modified here. Additionally, new versions can be created here, depending on the required application.



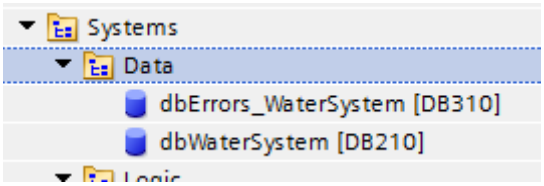
5. Reorganize the Program blocks after the library blocks have been automatically added so that navigation is easier. This is not a required step and should be organized based on personal preference or company standards. The organization of the blocks will not affect performance.



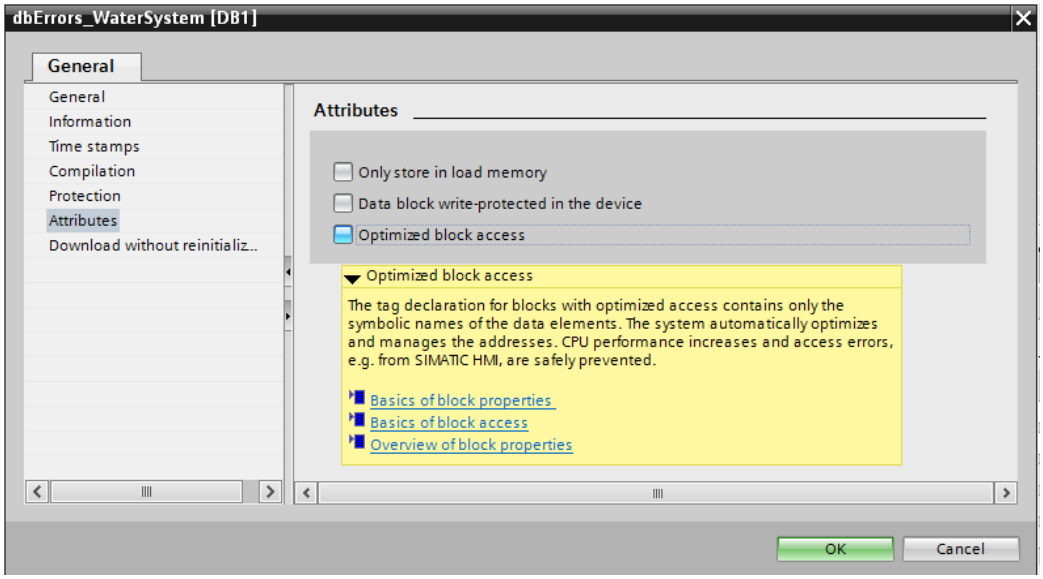
6.2.2. Create Required Data Blocks and User Defined Types

The following section contains the information pertaining to the addition of the required data blocks to contain the User Defined Types in the Siemens Open Library. Blocks can be named or organized in whatever way is best for the individual project, but this section will outline the creation of an HMI Data Block to contain the User Defined Types that map to the Faceplates and an Error Data block to contain the User Defined Types that contain alarms.

1. Create global Data Blocks for housing of the User Defined Types for the HMI and the Errors. In this example, the HMI data will be contained in dbWaterSystem and the error data will be contained in dbErrors_WaterSystem.



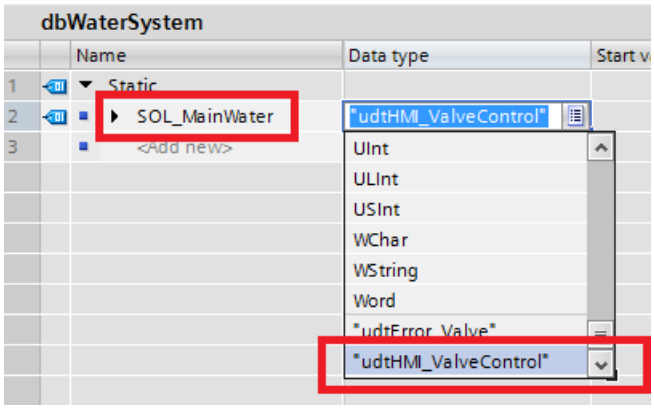
2. The Error Data block needs to be non-optimized to utilize the Excel Macro to automatically generate alarms. To setup non-optimized access, right click on the data block and select 'Properties.' Under the 'Attribute' tab, verify the 'Optimized block access' is not checked. For additional details see '5- Siemens Open Library - Siemens HMI Alarm Generation.'



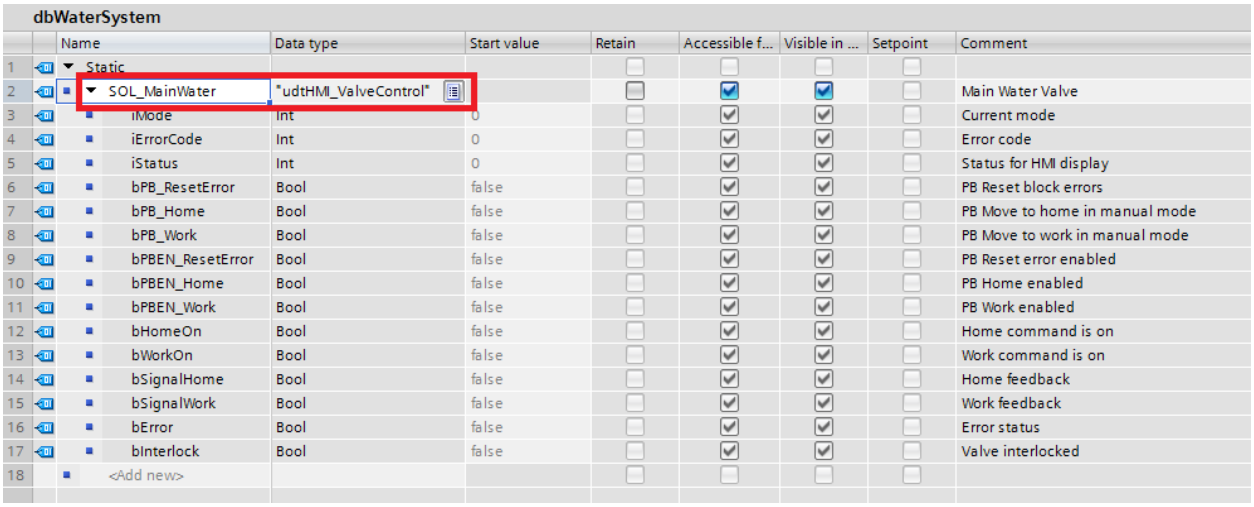
3. Repeat step 2 for the HMI data block if using an S7-300 or S7-400.

4. In the Data Block that will contain the HMI interface, dbWaterSystem in this example, add the HMI control structure for the device by naming the structure and assigning it the

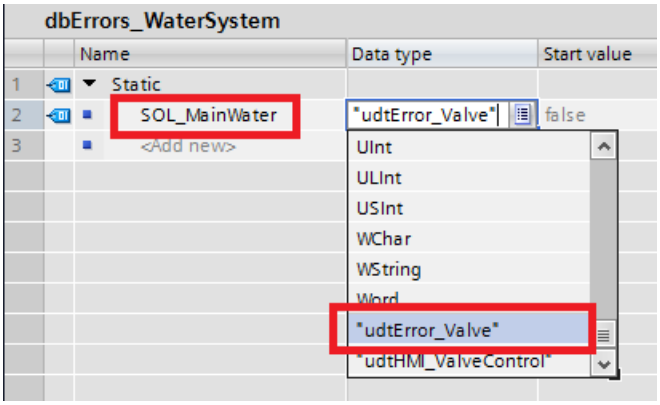
proper Data type. In this example, the UDT is named SOL_MainWater (SOL derives from Solenoid and is a DMC naming standard for Solenoid devices) but any name can be used.

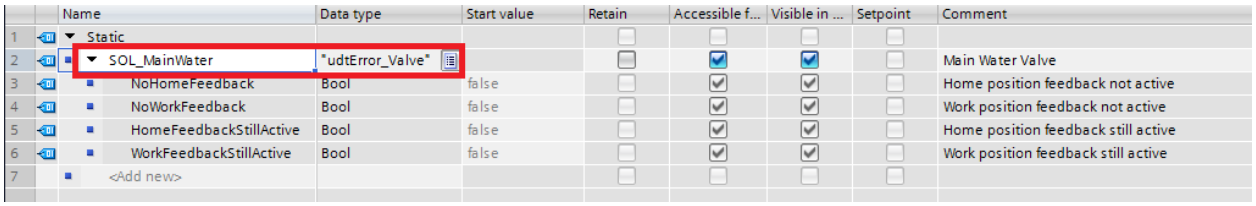


5. The User Defined Type has a large number of variables, with full details provided in the documentation for the specific Library Object that is being used. It can be expanded to see the variables available inside the UDT.



6. Add the error structure to error data block. In this example it is added to dbErrors_WaterSystem.

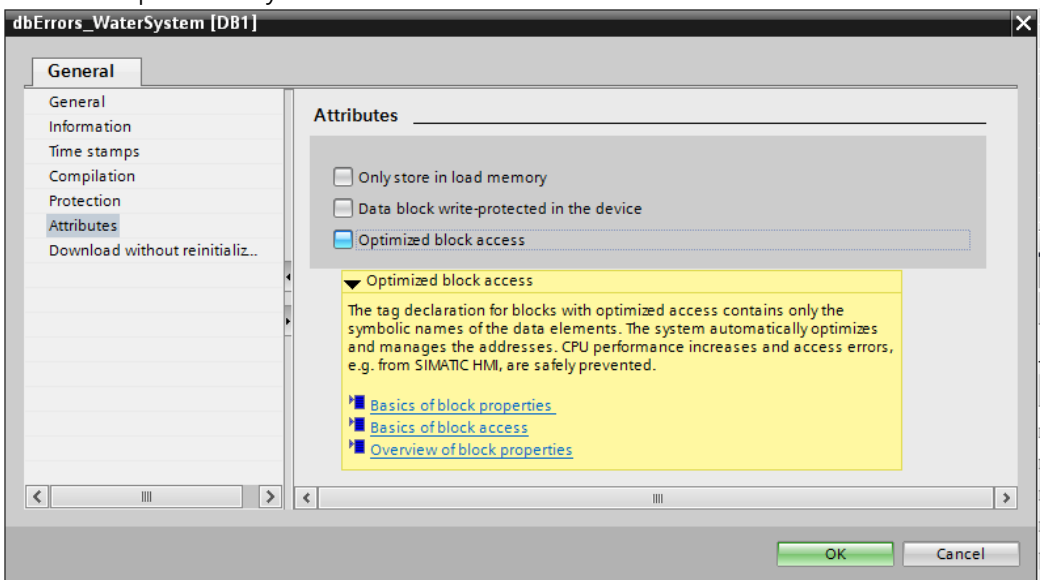




	Name	Data type	Start value	Retain	Accessible f...	Visible in ...	Setpoint	Comment
1	Static							
2	SOL_MainWater	"udtError_Valve"						Main Water Valve
3	NoHomeFeedback	Bool	false					Home position feedback not active
4	NoWorkFeedback	Bool	false					Work position feedback not active
5	HomeFeedbackStillActive	Bool	false					Home position feedback still active
6	WorkFeedbackStillActive	Bool	false					Work position feedback still active
7	<Add new>							

The Error Data Block needs to be non-optimized, for which the setting is available via the Data Block Properties by right clicking on the data block. For additional details see '5-

Siemens Open Library - Siemens HMI Alarm Generation.'



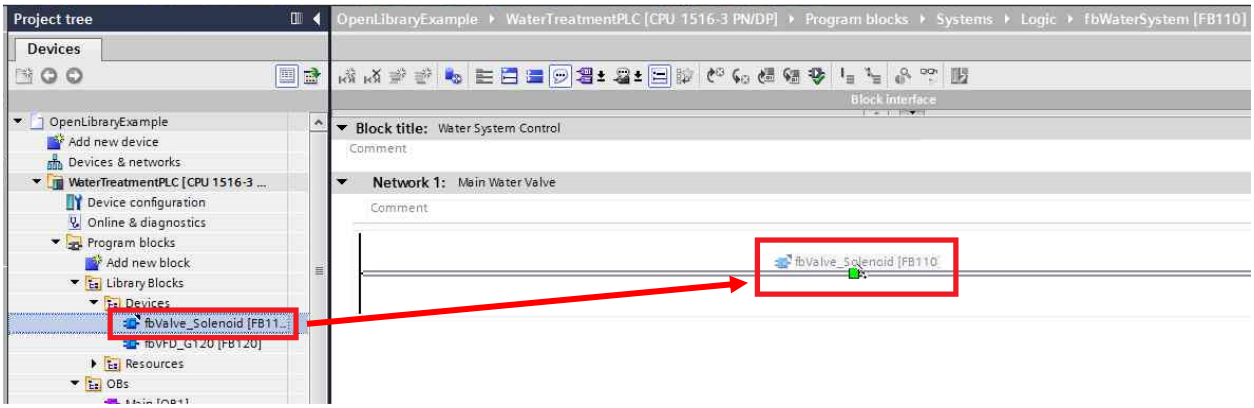
6.2.3. Create PLC Code

1. Create the function block that will contain the logic for the Solenoid valve. In this case we created a Function Block called fbWaterSystem. Add the inputs and outputs that correspond to the required logic for the water system.
2. Inside the function block that will contain the logic, add a Static Variable to contain the instance memory for the Library Object. In this example, 'SOL_MainWater" is added with data type 'fbValve_Solenoid'.

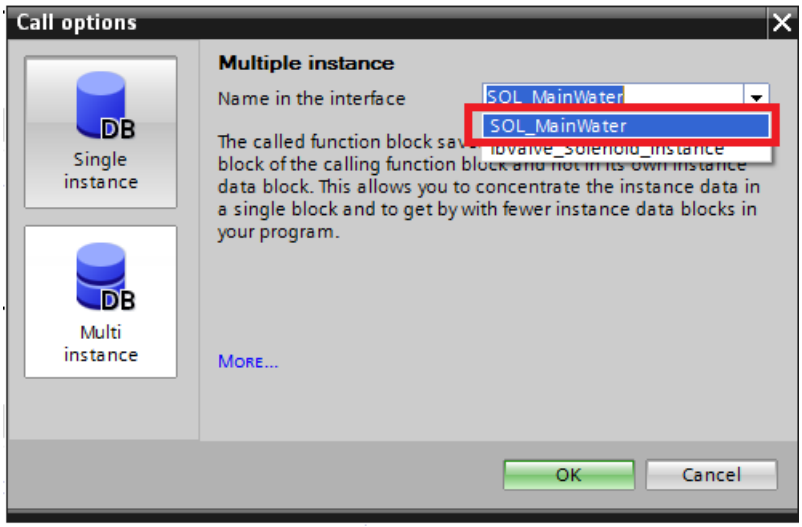
fbWaterSystem

	Name	Data type	Default value	Retain	Accessible f...	Visible in ...	Setpoint	Comment
1	Input				☐	☐	☐	
2	iInMode	Int	0	Non-retain	☑	☑	☐	Water System Mode
3	bInEstop	Bool	false	Non-retain	☑	☑	☐	E-Stop Occurred
4	bInResetError	Bool	false	Non-retain	☑	☑	☐	Reset Errors
5	Output				☐	☐	☐	
6	bOutAuto	Bool	false	Non-retain	☑	☑	☐	All Actuators in Auto
7	bOutError	Bool	false	Non-retain	☑	☑	☐	Block Errors Exists
8	InOut				☐	☐	☐	
9	<Add new>				☐	☐	☐	
10	Static				☐	☐	☐	
11	SOL_MainWater	"fbValve_Solenoid"			☑	☑	☐	Main Water Valve

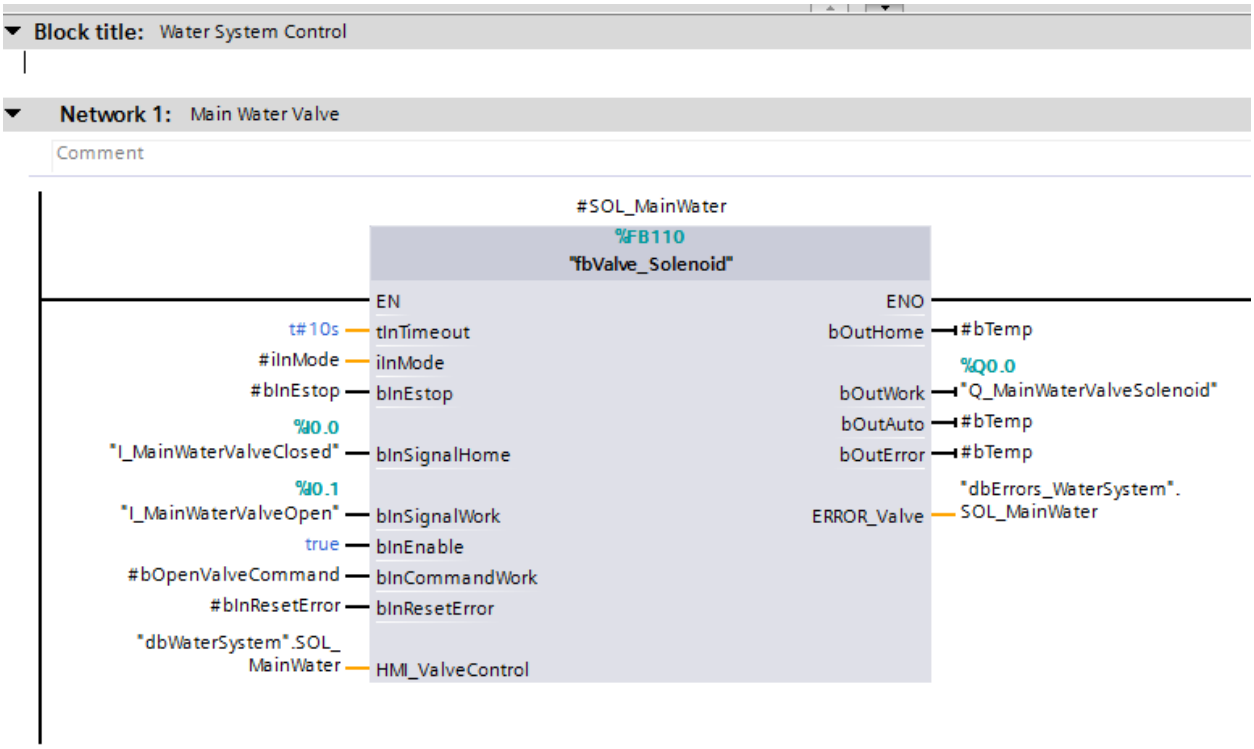
3. Drag and drop the Object from the project tree Program blocks into the ladder logic where it will be used.



4. Choose to use this block as a multiple instance call and select the instance created in the static memory of fbWaterSystem. Alternatively, you can provide a name for the instance memory and TIA Portal will automatically add it to the static variables section of the block.



5. Assign tags to the inputs and outputs of the object. Detailed information for the meanings of each input and output can be found in the '4- Siemens Open Library - Detailed Block Overview' document.



6.3. HMI Development

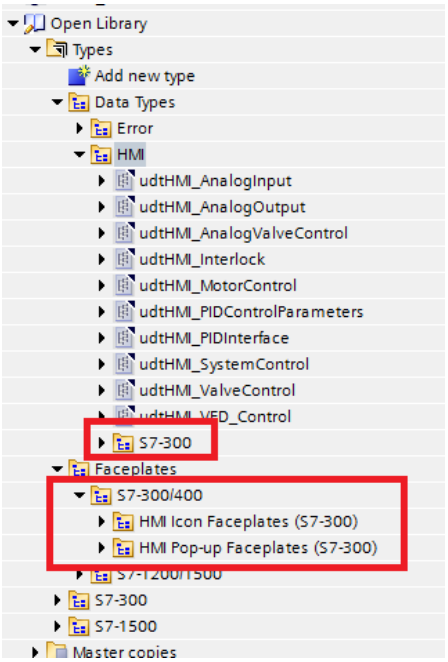
This section covers the use of the HMI Icon and the HMI Pop-up screen. The HMI library can be used as a fully functioning library, or it can be used as a building block for custom HMI Icons and Pop-ups.

6.3.1. Using SIMANTIC Visualization Architect (SiVArc) for HMI Development

One option for HMI development is to use SiVArc, a plug-in for TIA portal that allows for the automatic generation of screens and screen objects based off the controller program for the project. SiVArc is compatible with TIA Portal V14 and later, and is compatible with S7-1200 and S7-1500 PLC's, but is not compatible S7-300 or S7-400 PLC's. To implement SiVArc in a project, refer to "8 – SIMANTIC Visualization Architect" for a description of the process, and "4 – Detailed Block Overview" for the specifics of using it with a particular library object. If not using SiVArc, follow the directions below.

6.3.2. Special considerations for S7-300/S7-400 PLCs

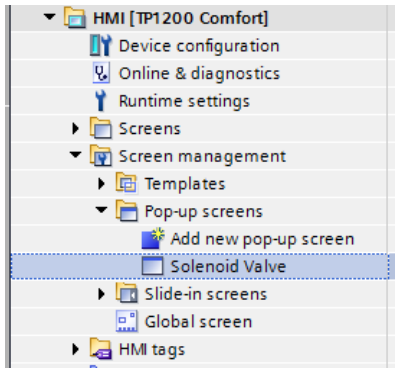
If an S7-300 or S7-400 PLC is used in the project, the faceplates and UDTs created specifically for the S7-300 series PLC need to be used. The below instructions can still be followed, however, the separate faceplates need to used where applicable.



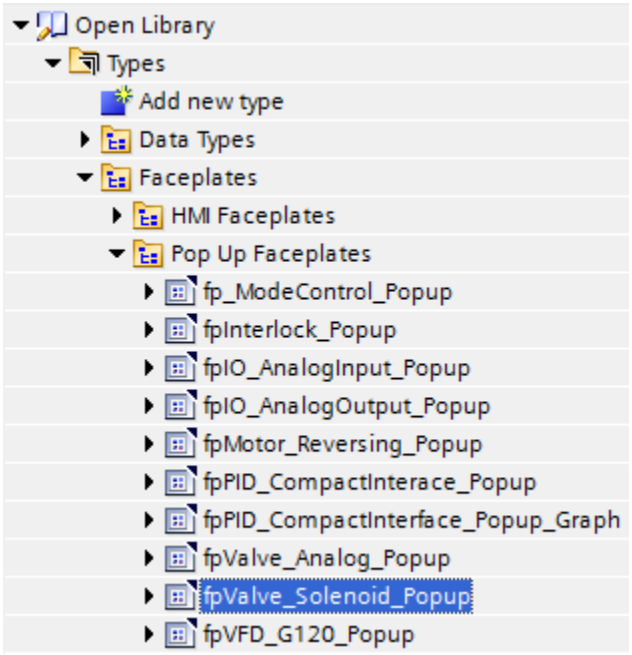
6.3.3. HMI Pop-up Creation and Configuration (no SiVArc)

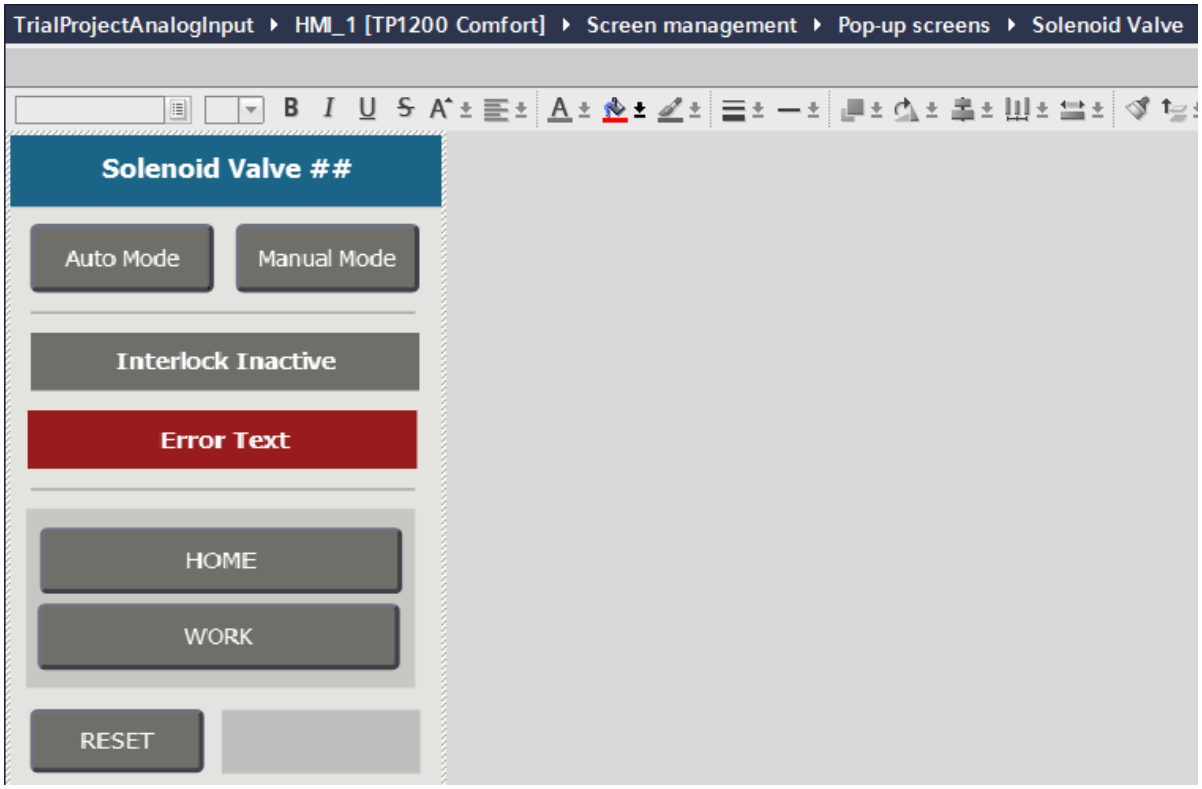
This section covers how to use the Pop-up faceplates on a pop-up screen. Each object will have a separate pop-up screen.

1. Create a new Pop-up screen for the Solenoid Valve. This is done by clicking the 'Add new pop-up screen' button in the Pop-up screens section. Give a name that identifies the specific object being used.

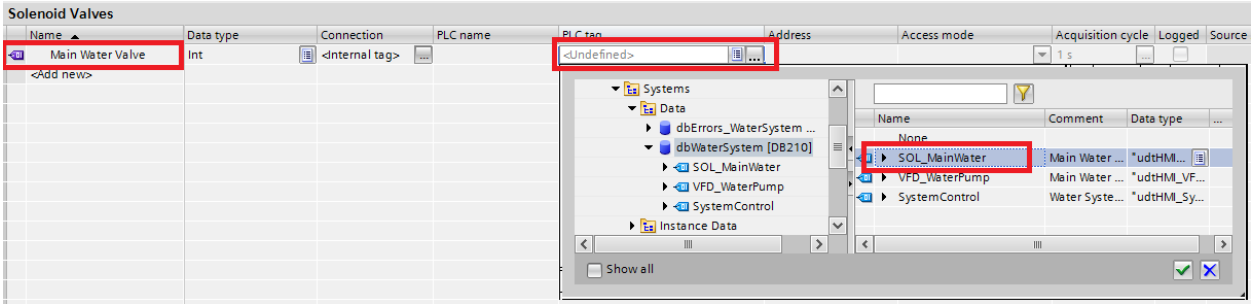


2. Drag the Pop-up Faceplate, fpValve_Solenoid_Popup in this example, from the library onto the new Pop-up screen. The faceplate can be resized to best fit the size of the HMI that is being used, however, increasing text size requires making a new version of the object with larger text. Be sure that the Pop-up screen size is equivalent to the size of the faceplate. If the screen size is smaller than the faceplate, it will be cut off. If the screen size is larger than the faceplate, there will be a white boarder around the Pop-up.

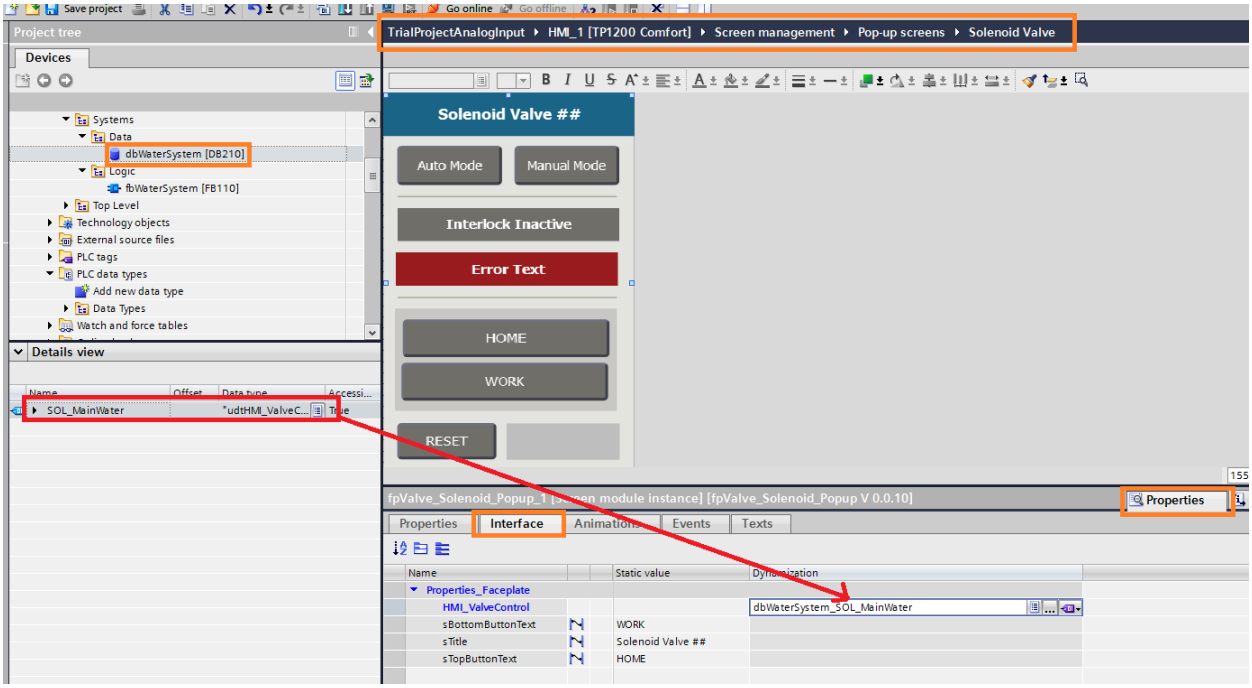




3. Create an HMI tag for the object in the HMI Tag Table and map it to the HMI UDT that was created on the PLC, in this case: "dbWaterSystem".SOL_MainWater. As an additional option, the tag can be automatically created by dragging and dropping the UDT directly from the PLC into the HMI Interface.
   Below shows the method for creating the object tag from the HMI Tag Table:

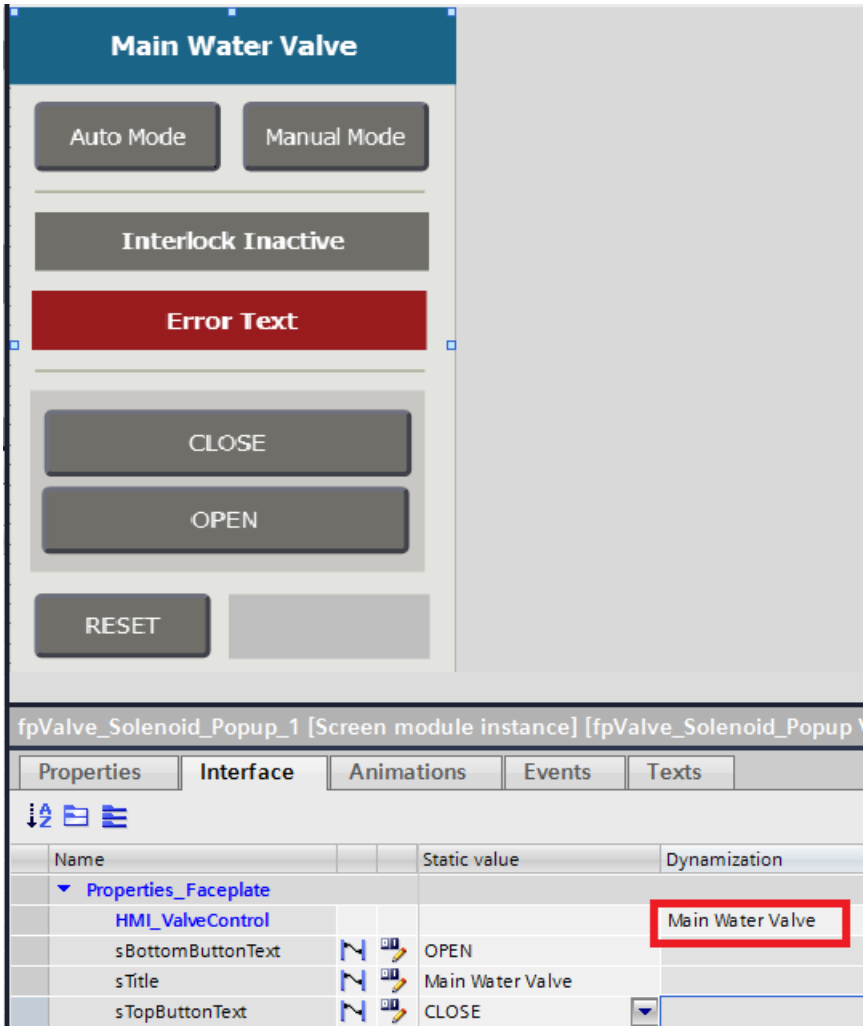


   And below shows how to have Portal automatically have the tags created for you, by dragging and dropping the UDT directly into the interface:

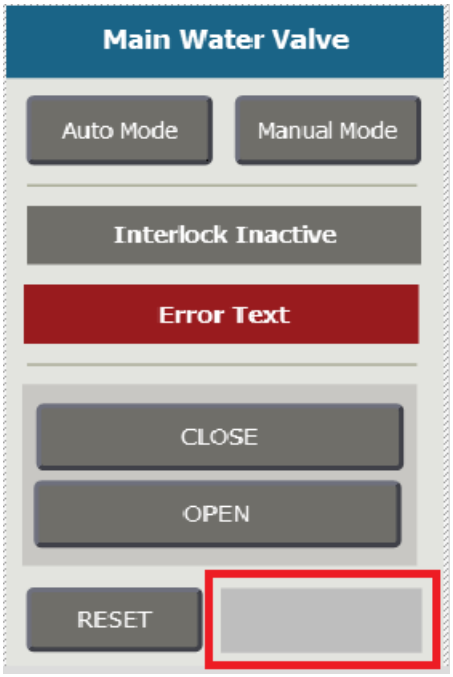


4. Configure the required variables in the Interface of the faceplate. If the Interface tab is not visible, right click on the faceplate, select properties, and the window should appear on the bottom of the screen. Each library object has a slightly different interface. The details of each can be seen in the documentation for the specific object. In this case, the interface contains the following variables: HMI_ValveControl, sBottomButtonText, sTopButtonText, and sTitle. Configure the HMI_ValveControl using the HMI tag created in step 3, or by

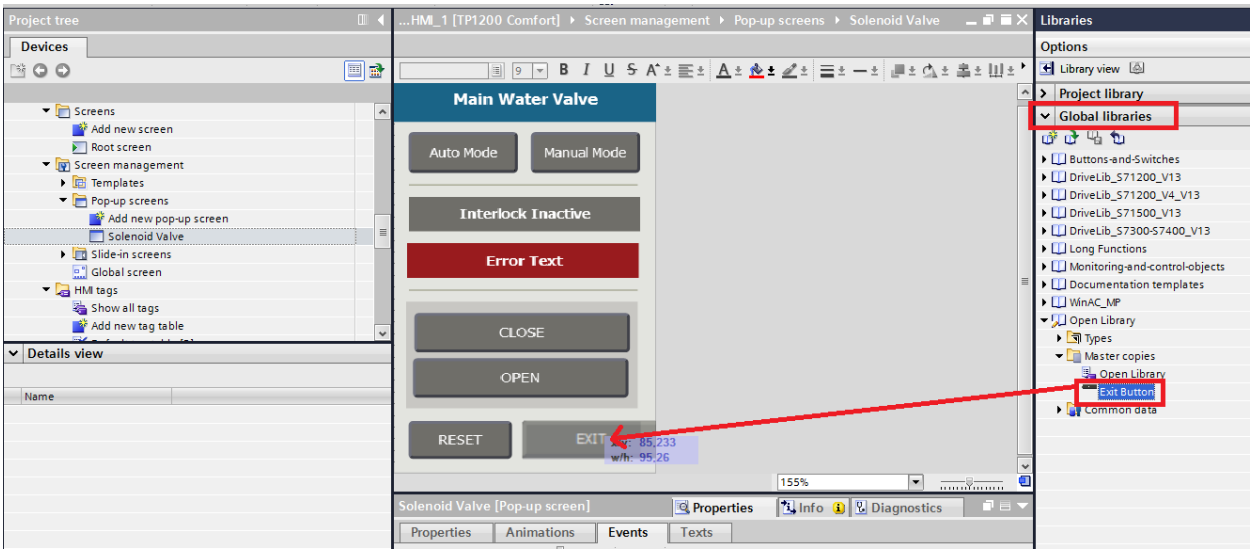
dragging and dropping the tag from the PLC. Configure any strings required by typing the fixed text into the interface.



5. You will notice on the faceplate that there is a greyed-out area at the bottom of the faceplate, typically in the bottom right area. This is for the addition of an exit button to the popup screen, allowing you to successfully close the popup when you wish to revert back to the original screen.

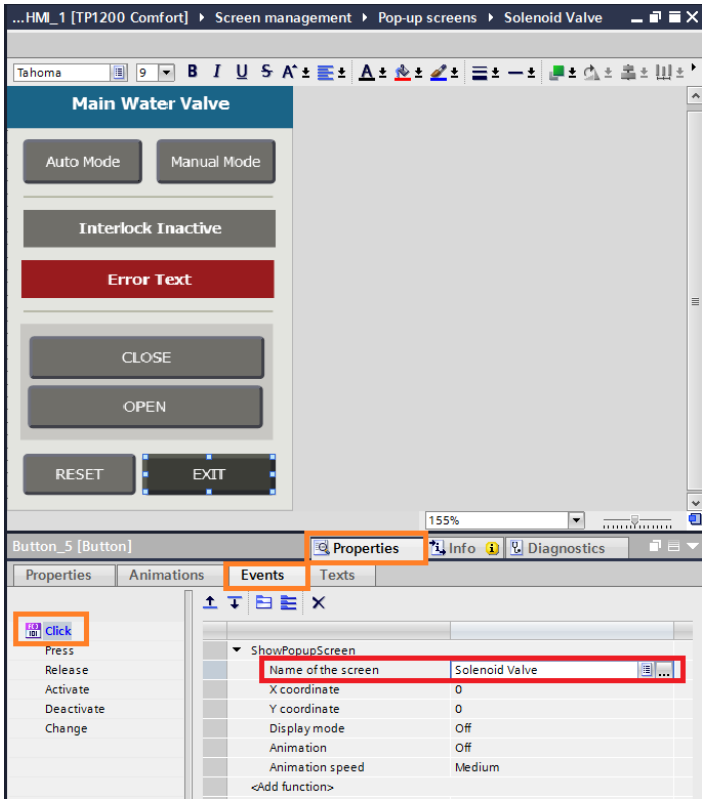


6. To add this button, you need to retrieve it from the Open Library under 'Master Copies'. It will be named 'Exit Button'. Drag and drop it onto the faceplate. Note: the button size will vary depending on the faceplate, and may need to be resized to fit correctly.



7. Adding the popup functionality: We want the exit button to exit out of the popup screen. This can be achieved by using the ShowPopupScreen function. To do this, click on the Exit button, navigate to the button properties tab (either by right-clicking and selecting properties, or alternatively just pressing Alt + Enter). In the Properties sub-tab click over to

the Events tab. There should already be a ShowPopupScreen Function added to the function list. You will need to simply add the name of the screen that you wish to exit, in this case it is 'Solenoid Valve'. Now the button is set up to turn the popup screen off when pressed.

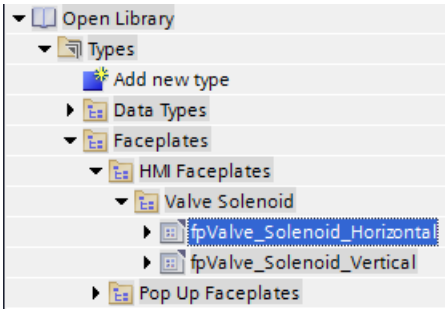


6.3.4. HMI Icon Creation and Configuration (no SiVArc)

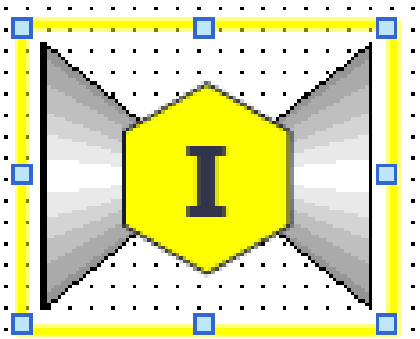
This section covers how to configure the HMI Icon. The library contains several predefined HMI Icons. Many more can be created, or existing icons can be modified using the Symbol Object Library. For details on customizing objects, see '7- Siemens Open Library – Library Customization.'

Details regarding the functionality, colors, and interface to each HMI Icon can be found in the documentation for that object.

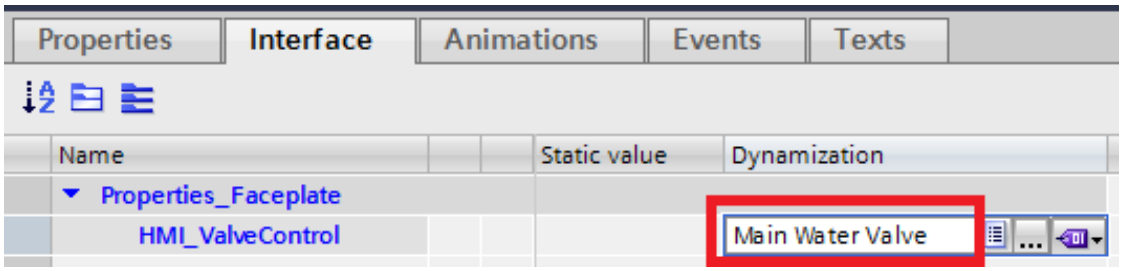
1. Navigate to the correct HMI Icon Faceplate in the Open Library. In this case we will use fpValve_Solenoid_Horizontal



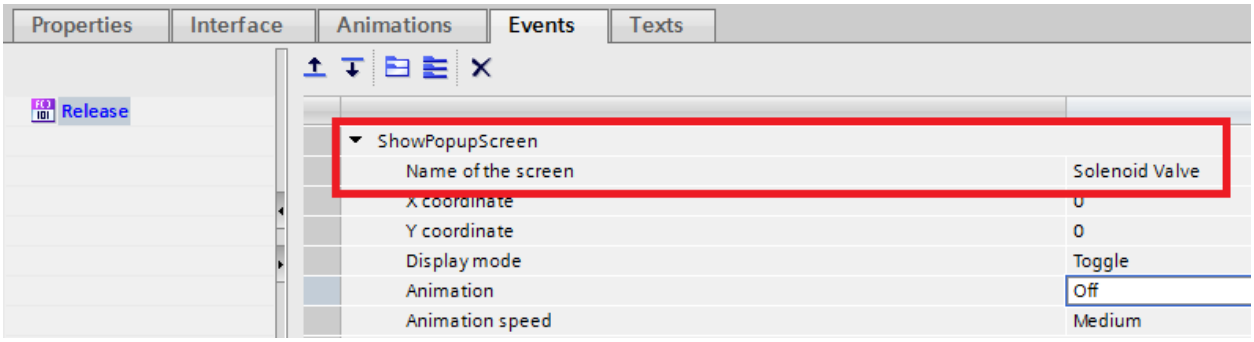
2. Pull the Icon Faceplate into the desired screen. If the device is rotated, the 'I' will be at the angle of the device. There is a Horizontal and Vertical version of all devices to allow for vertical alignment with a correctly oriented interlock symbol.



3. Configure the Interface to the Icon Faceplate using the tag configured in the HMI Pop-up, or by dragging and dropping the UDT from the PLC.



4. Create an event that will open the Pop-up window in runtime when the valve on the HMI screen is clicked.


Properties
Interface
Animations
Events
Texts
Release
ShowPopupScreen
Name of the screen
Solenoid Valve
X coordinate
0
Y coordinate
0
Display mode
Toggle
Animation
Off
Animation speed
Medium