

Contents

TwinCAT HMI Project Generator	2
Home Page	2
Localization Variables	2
Bring User Controls into the project	3
Populate Manual and Automatic Mode contents	4
Operation Mode User Control	4
Internal symbol for Feeder selection	6
Feeder and Operation Mode Menu	6
Events for toggling feeder data	9
Alarming and Messaging	11
Historization	11
Recipe Management	14
Animation User Control	15
Enum Descriptions	18
Appendix – Build User Controls (UCs)	20
Create Automatic Axis Info UC	20
Create Manual Axis UC	21
Create FeederAutomatic UC	23
Create FeederManual UC	24

TwinCAT HMI Project Generator

New TwinCAT HMI Project using Project Generator wizard

1. Base Application template
2. Responsive Application
3. Default theme – Base
4. Navigation menu on the left
5. two views – Mobile and Desktop switched at 1000px width
6. Add 7 pages:
 - a. Home
 - b. Automatic Mode
 - c. Manual Mode
 - d. Events
 - e. Trending
 - f. Recipes
 - g. Settings

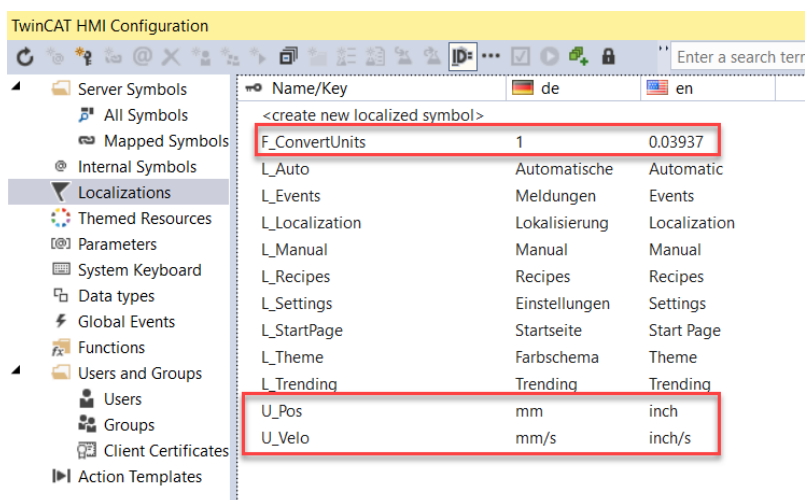
Home Page

1. Click on Gallery Explorer -> Samples -> Woodworking line -> Substations -> Beckhoff_hmi_plain_no_bg_v02.png
2. Drag and drop the picture into the Images folder of the HMI project tree
3. Drag and drop from this image from the solution tree into the Designer Home.content
4. Rename image and change layout to make it fit entire content page

Localization Variables

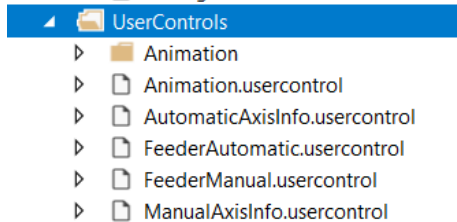
Add the following localization variables (Ensure case matches the case in the picture otherwise pre-done control bindings will not work):

1. U_Position
2. U_Velocity
3. F_ConvertUnits (1 (de) : 0.3937 (en)).

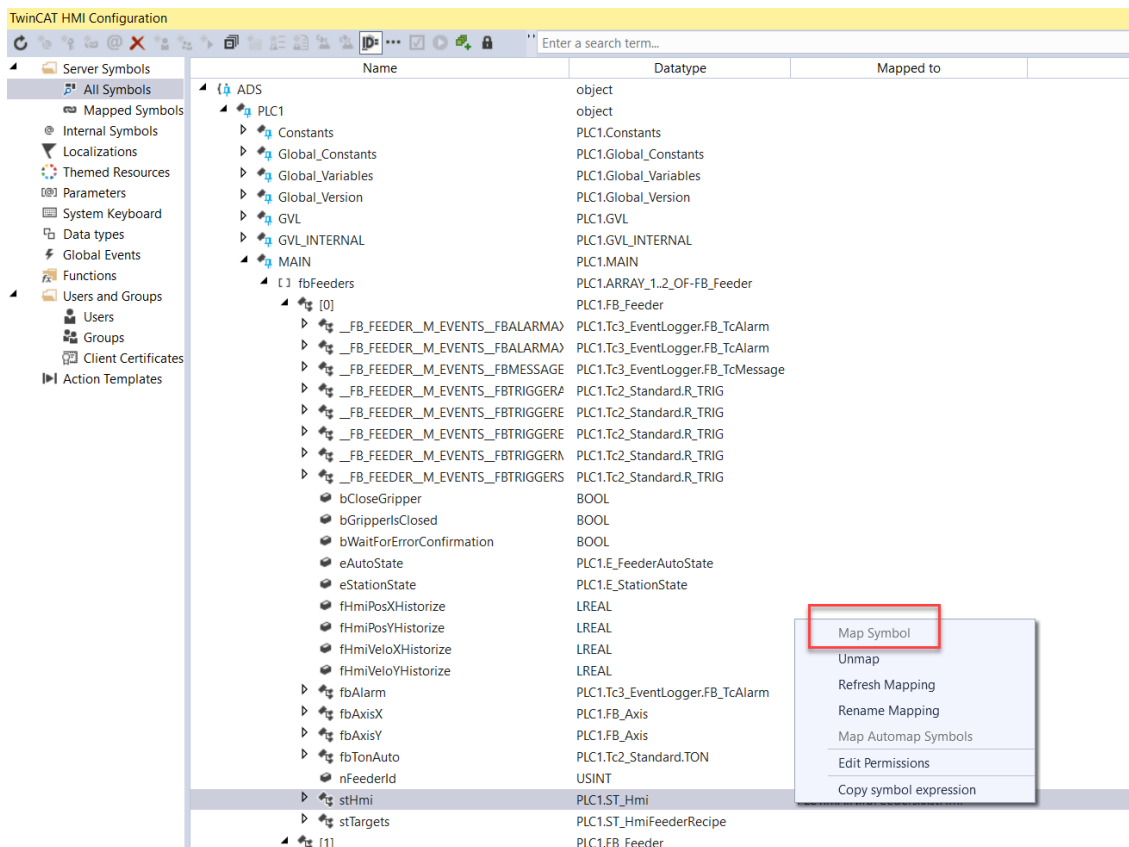


Bring User Controls into the project

1. Create “UserControls” Folder. Ensure name is exact otherwise UC references in HTML might get broken
2. Then add each UC inside the UserControls folder by clicking on “Add Existing Item”

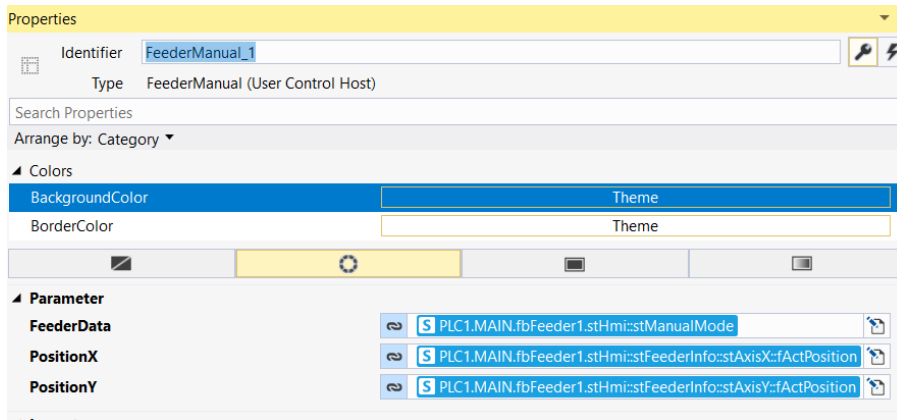


3. Map stHmi variables for both feeders from the PLC. This will make PLC datatypes available inside the HMI project. UC parameters use these datatypes:

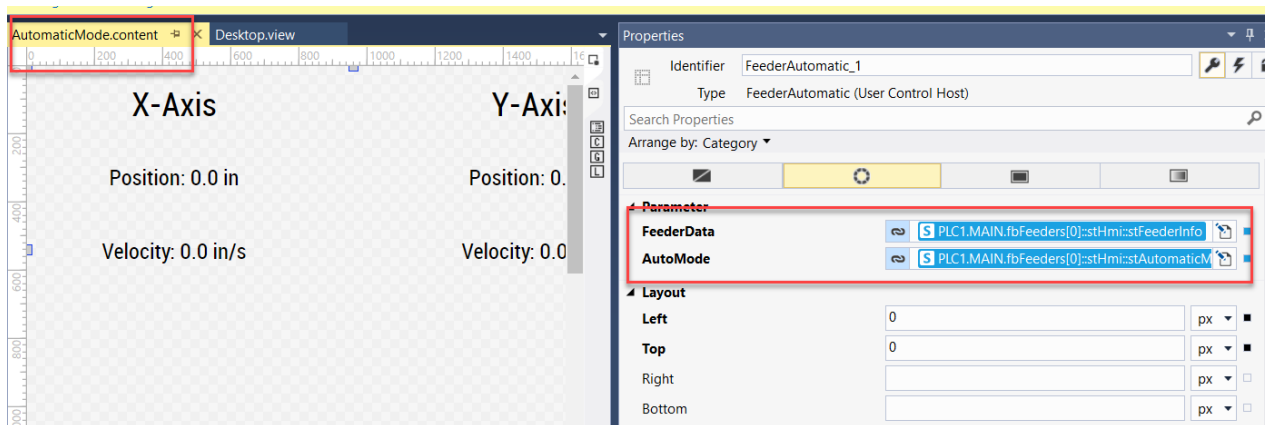


Populate Manual and Automatic Mode contents

1. Add one instance of FeederManual UC on Manual mode content and bind to PLC symbols:

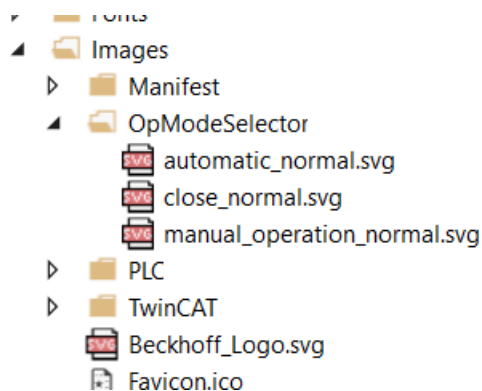


2. Add one instance of FeederAutomatic UC on AutoMode content and bind to PLC symbols:



Operation Mode User Control

1. Create a new UC and add "Grid" control with 3 rows
2. Add one button each to all rows with 2% spacing at Top and Bottom, 30% spacing at Left and Right
3. Add folder under Images and add images from the Images folder in the repo:



4. Add these images as icons to buttons with 100% width and height
5. Add parameter of type `feeders[0].stHmi.OperationMode` to the UC

Edit/Define Parameters

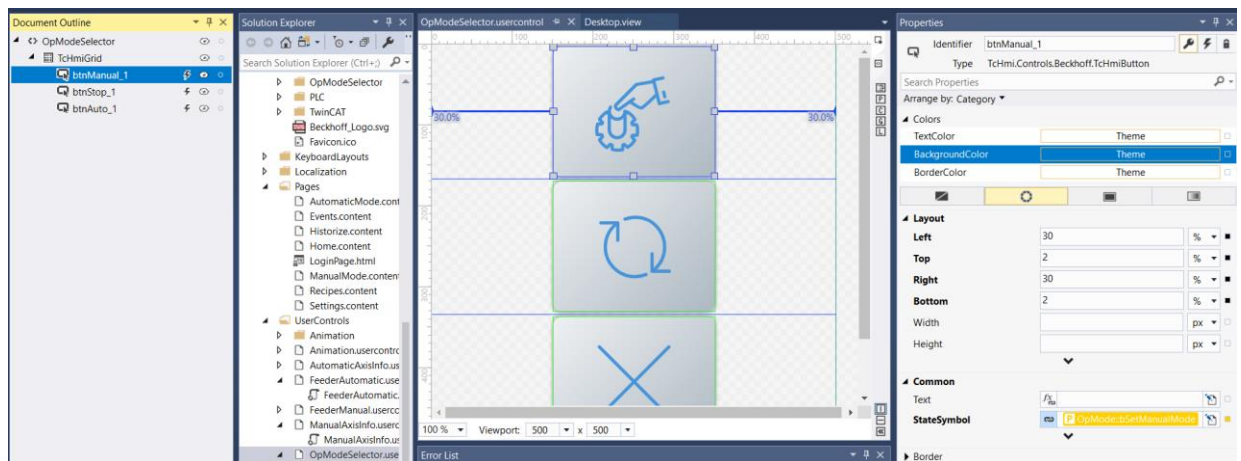
Description:

Parameters:

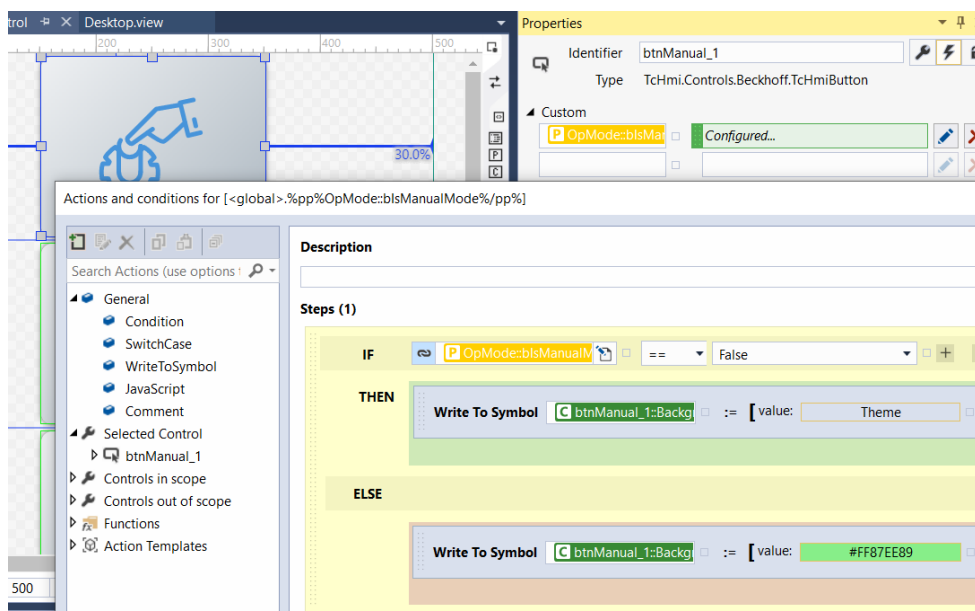
Name	Datatype	(Drop) Default Value	DefaultValueInternal	Bindable	Visibility
OpMode	S PLC1.ST_HmiC	(Object) ...	(Object) ...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

tchmi:server#/definitions/PLC1.ST_HmiOperationMode

6. Map 'State symbol' of each button to parameters `bSetAutoMode`, `bSetManualMode`, `bSetStopMode` respectively

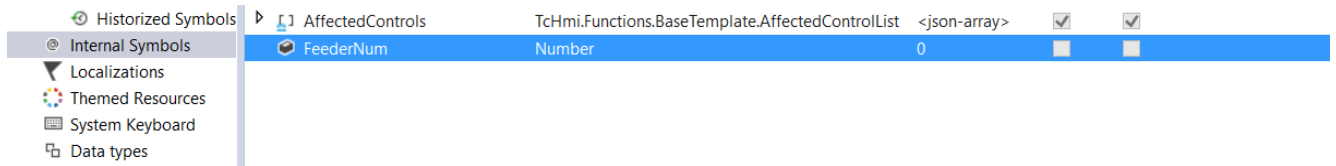


7. Add custom trigger conditions for each button for example: 'isAutoMode' -> if 'isAutoMode' TRUE then background color Green ELSE theme'



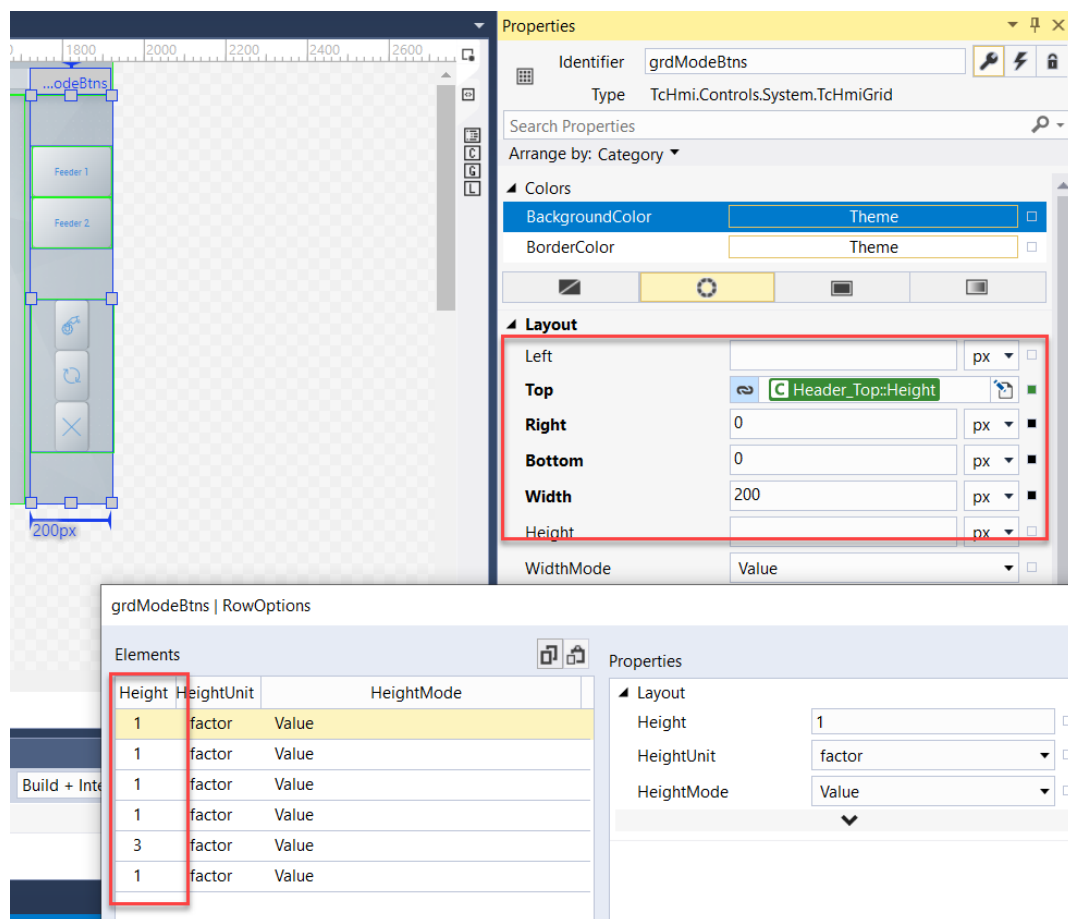
Internal symbol for Feeder selection

We need to add an internal variable to switch Feeders between 1 and 2. This symbol will be used to toggle feeder data.

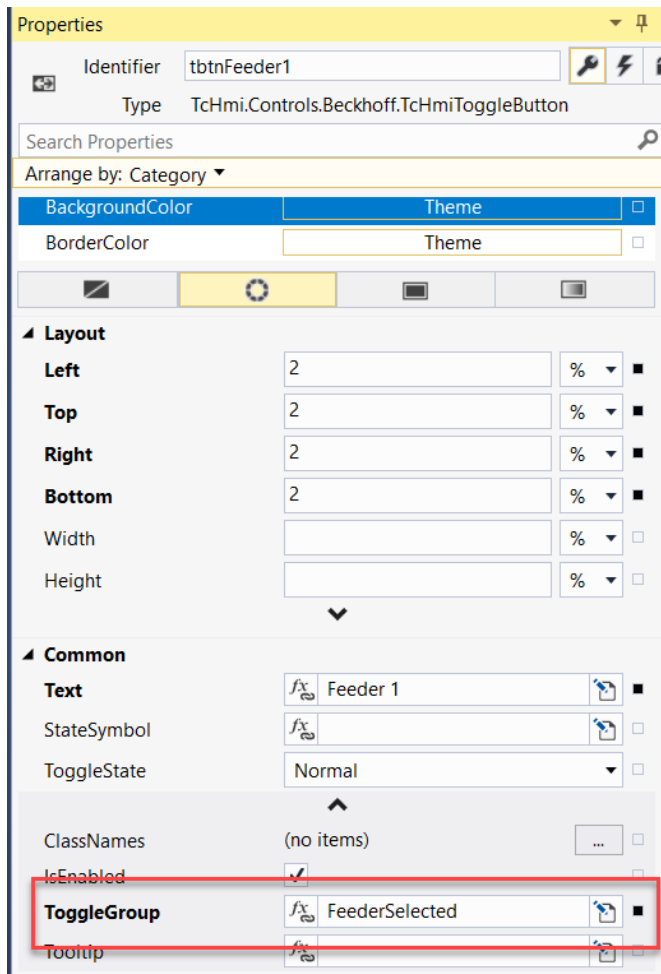


Feeder and Operation Mode Menu

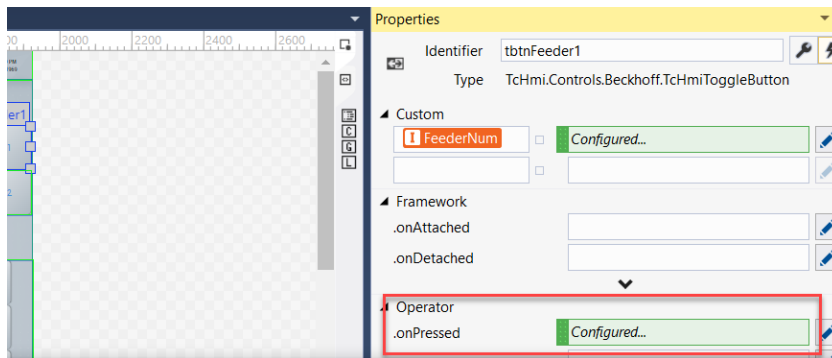
1. Add Grid on the right side of Desktop.view. Bind 'Top' with Height of header bar, 'Right' 0, 'Bottom' 0, 'Width' 200px
2. Change Right of Region_Center to Width of grdModeBtns
3. Add 6 rows – all cells with factor 1, fifth cell with factor 3



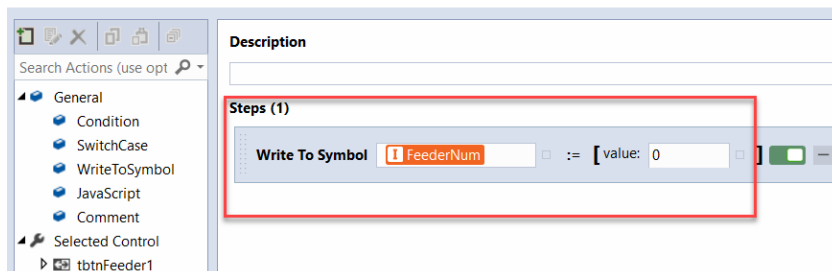
4. Add OpMode UC instance and map to Feeder 1 variable from the HMI
5. Add toggle buttons in second and third cells of the grid – 2% spacing from Left Right Top Bottom
6. Add 'ToggleGroup' for each button, which will ensure only one button from the group is toggled on at a time



7. Write to internal symbol at 'onPressed' event of each toggle button



Actions and conditions for [tbtnFeeder1.onPressed]



8. Add custom trigger condition with internal symbol to change toggleState and BackgroundColor

The screenshot displays the Siemens STEP 7 LAD editor interface. The main workspace shows a ladder logic network with a timer T1 and a callout to a sub-network 'Feeder1'. The 'Properties' window on the right shows the identifier 'tbtnFeeder1' and type 'TcHmi.Controls.Beckhoff.TcHmiToggleButton'. A red box highlights the 'Custom' property, which is set to 'FeederNum' and 'Configured...'. Below the properties, the 'Description' and 'Steps (1)' sections are visible. The 'Steps (1)' section shows a custom trigger condition 'IF FeederNum == 0'. The 'THEN' branch contains two 'Write To Symbol' actions: 'tbtnFeeder1::Toggle' set to 'Active' and 'tbtnFeeder1::Backgr' set to '#FF87EE89'. The 'ELSE' branch contains two 'Write To Symbol' actions: 'tbtnFeeder1::Toggle' set to 'Normal' and 'tbtnFeeder1::Backgr' set to 'Theme'.

derNum%[/i%]

Description

Steps (1)

IF ☐ **FeederNum** ☐ **==** ☐ **+**

THEN

Write To Symbol ☐ **tbtnFeeder1::Toggle** ☐ **:=** ☐ **+**

Write To Symbol ☐ **tbtnFeeder1::Backgr** ☐ **:=** ☐ **+**

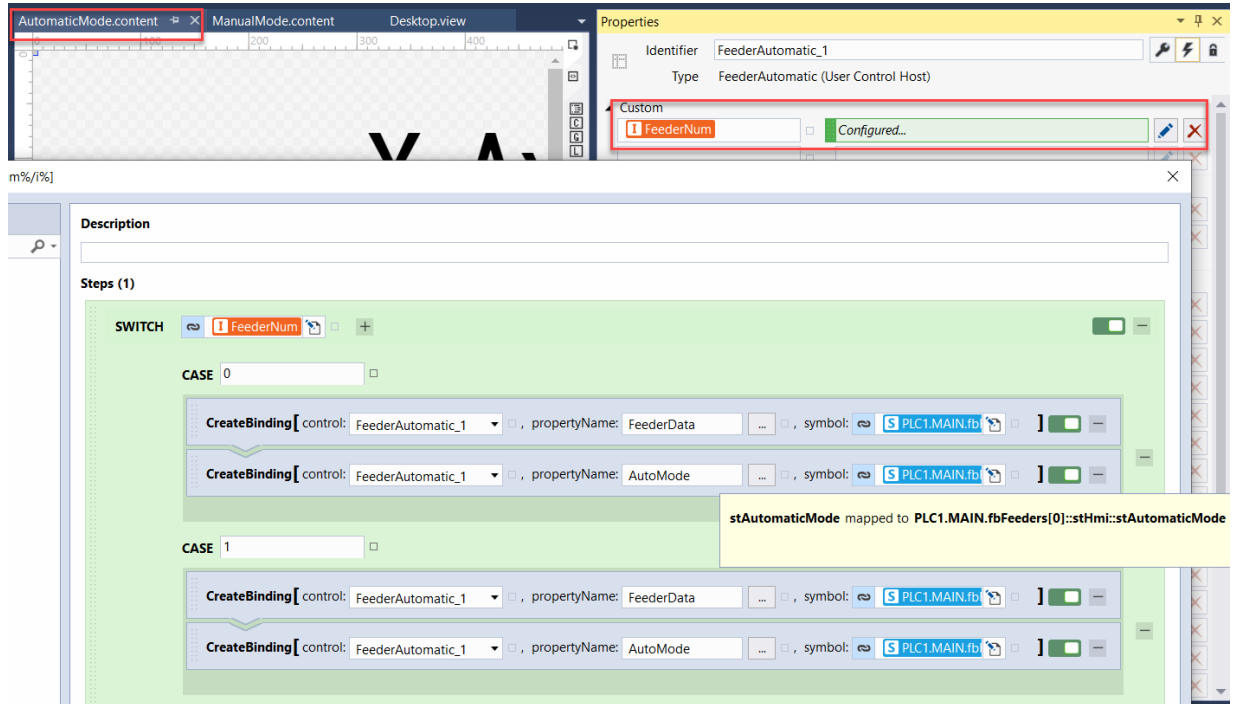
ELSE

Write To Symbol ☐ **tbtnFeeder1::Toggle** ☐ **:=** ☐ **+**

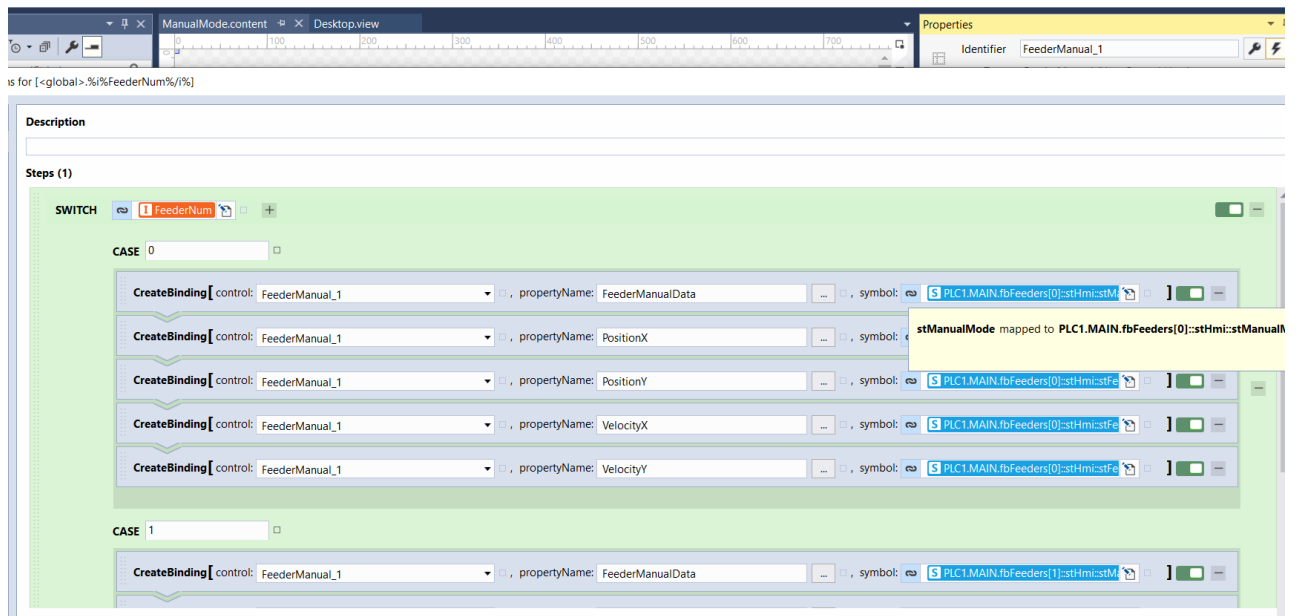
Write To Symbol ☐ **tbtnFeeder1::Backgr** ☐ **:=** ☐ **+**

Events for toggling feeder data

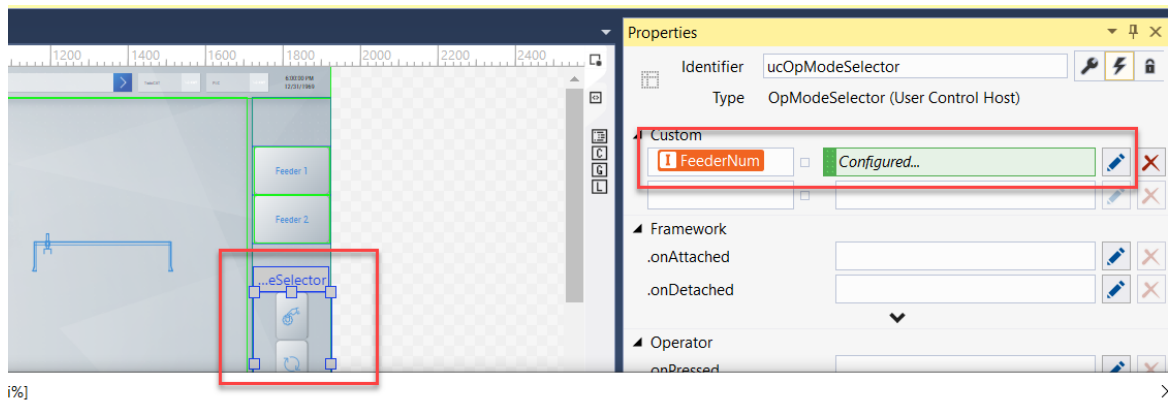
1. Open AutomaticMode.content and add custom event triggered by the internal symbol on this page. Actions will be to switch bindings from Feeder 1 to 2 or vice-versa, which is done by using the CreateBinding function:



2. Add the same custom event on Manual.content and use CreateBinding to change 5 bindings:

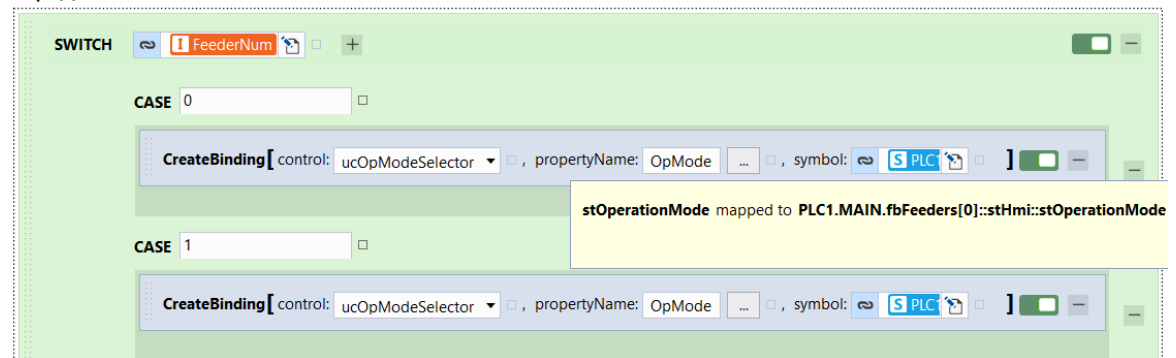


3. Add the same custom event on OpMode UC instance and use CreateBinding to change its parameter binding:



Description

Steps (1)



Alarming and Messaging

1. Event Grid has been added for us on the Events page
2. Attributes of the grid include showing menu bar, setting filters
3. On the runtime menu bar, domains include ADS, TcHmiSrv and more, but not “EventLogger”
4. Add event logger server extension
5. This will show events from the EventLogger domain where:
 - a. Messages can be triggered by switching operation modes
 - b. Alarms are triggered by jogging with a high velocity (which can be set on the PLC)

Historization

1. Add Trendline control instance on the Trending.content page
2. Set ‘Start’ attribute to PT2M and ‘End’ to Latest
3. Add 2 y-axes (we need to ensure they don’t overlap so data can be seen clearly)
 - a. Position from 0-5000 with units

tlcHistorize | Y-Axis Definitions

Id	Axis Name
1	Position
2	Velocity

Properties

Colors

- Label font color: Theme
- Axis color: Theme
- Axis name font color: Theme

General

- Id: 1
- Position: Left
- Show axis: ☒
- Main-tick min-value: 0
- Main-tick max-value: 5000 * L.F.ConvertUnits
- Auto scaling: ☐
- Logarithmic scale: ☐
- Axis labeling: Number
- Decimal places: 2
- Show labels: ☒
- Show axis name: ☒
- Axis name: Position
- Axis name font-family: [font family]
- Axis name font-size: 15
- Axis name font-size-u...: px
- Axis name font-weight: Bold
- Unit: U_pos

OK Cancel

- b. Velocity from -2000-2000 with units

tlcHistorize | Y-Axis Definitions

Id	Axis Name
1	Position
2	Velocity

Properties

Colors

General

Id: 2

Position: Right

Show axis: ☒

Main-tick min-value: $-2000 \cdot \frac{1}{F} \cdot \text{ConvertUnits}$

Main-tick max-value: $2000 \cdot \frac{1}{F} \cdot \text{ConvertUnits}$

Auto scaling: ☐

Logarithmic scale: ☐

Axis labeling: Number

Decimal places: 2

Show labels: ☒

Show axis name: ☒

Axis name: $\frac{1}{F} \cdot \text{Velocity}$

Axis name font-family: $\frac{1}{F} \cdot$

Axis name font-size: 15

Axis name font-size-u...: px

Axis name font-weight: Bold

Unit: $\frac{1}{F} \cdot \text{U.Velo}$

OK Cancel

4. Add 4 LineGraphDescriptions – set colors, symbol, y-axisID, Scale factor, Legend name

tlcHistorize | LineGraphDescriptions

Id	Y-Axis Name	Line Width
1	Position	1
1	Position	1
2	Velocity	1
2	Velocity	1

Properties

Colors

Line color: #FF4794DA

Point dot fill color: #FF4794DA

Point dot stroke color: #FF4794DA

Fill color: #FF4794DA

General

Symbol: PLC1.MAIN.fbFeeders.0.fHmiPosXHistoriz

Y-Axis Id: Position

LineWidth: 1

Scale factor: $\frac{1}{F} \cdot \text{F} \cdot \text{ConvertUnits}$

Legend name: $\frac{1}{F} \cdot \text{Position X}$

Point dot: ☐

Point dot radius: 3

Point dot stroke width: 1

Point dot in stopmode: ☒

FillMode: None

FillTransparency: 50

OK Cancel

- There are no historization symbols to pick from. We need to add historize server extension from NuGet package management.
- Map the following symbols and historize:

TwinCAT HMI Configuration

Name	Datatype	Interval	Max Entries	Recording
PLC1.MAIN.fbFeeder1.fHmiPosXHistorize	LREAL	PT1S	10000	<input checked="" type="checkbox"/>
PLC1.MAIN.fbFeeder1.fHmiPosYHistorize	LREAL	PT1S	10000	<input checked="" type="checkbox"/>
PLC1.MAIN.fbFeeder1.fHmiVeloXHistorize	LREAL	PT1S	10000	<input checked="" type="checkbox"/>
PLC1.MAIN.fbFeeder1.fHmiVeloYHistorize	LREAL	PT1S	10000	<input checked="" type="checkbox"/>
PLC1.MAIN.fbFeeder2.fHmiPosXHistorize	LREAL	PT1S	10000	<input checked="" type="checkbox"/>
PLC1.MAIN.fbFeeder2.fHmiPosYHistorize	LREAL	PT1S	10000	<input checked="" type="checkbox"/>
PLC1.MAIN.fbFeeder2.fHmiVeloXHistorize	LREAL	PT1S	10000	<input checked="" type="checkbox"/>
PLC1.MAIN.fbFeeder2.fHmiVeloYHistorize	LREAL	PT1S	10000	<input checked="" type="checkbox"/>

- Historizing with interval of 1s will show us “unsmooth” data. Change historize settings to 10ms:
- Add trigger condition for Feeder number and change LineGraphDescriptions:

The screenshot shows the TwinCAT HMI Configuration interface with several windows open:

- Data Definitions:** A table with columns 'Id', 'Y-Axis Name', and 'Line Width'. It contains two rows of generated data.
- Properties:** A window showing the 'Colors' section with 'Line color', 'Point dot fill color', 'Point dot stroke color', and 'Fill color' all set to '#FF6AECF5'. The 'General' section shows 'Symbol' as 'PLC1.MAIN.fbFeeder1', 'Y-Axis Id' as '1', 'Line Width' as '1', 'Scale factor' as '1', and 'Legend name' as 'Position X'.
- Properties:** A window showing the 'Custom' section with 'FeederNum' set to 'Configured...'. The 'Write To Symbol' section shows 'TicHistorizeLineGra' with a value of '(4 items)'.

Recipe Management

1. Add Recipe Server extension
2. Add two recipe types – Feeder1 and Feeder2
3. Select Feeder1 Recipe symbols and rename display names

Recipes.content Feeder1 Desktop.view

Display Name	Symbol	Default	Min	Max	Unit	Enabled
Target1XPos	PLC1.MAIN.fbFeeders[0]::stHmi::stFeederRecipe::fTarget1PositionX	500	0	1000	mm	<input checked="" type="checkbox"/>
Target1YPos	PLC1.MAIN.fbFeeders[0]::stHmi::stFeederRecipe::fTarget1PositionY	500	0	1000	mm	<input checked="" type="checkbox"/>
Target2XPos	PLC1.MAIN.fbFeeders[0]::stHmi::stFeederRecipe::fTarget2PositionX	1000	0	1000	mm	<input checked="" type="checkbox"/>
Target2YPos	PLC1.MAIN.fbFeeders[0]::stHmi::stFeederRecipe::fTarget2PositionY	1000	0	1000	mm	<input checked="" type="checkbox"/>
Velocity	PLC1.MAIN.fbFeeders[0]::stHmi::stFeederRecipe::fVelocity	50	0	100	%	<input checked="" type="checkbox"/>
RecipeName	PLC1.MAIN.fbFeeders[0]::stHmi::stFeederRecipe::sRecipeName	Default				<input checked="" type="checkbox"/>
Target1Name	PLC1.MAIN.fbFeeders[0]::stHmi::stFeederRecipe::sTarget1Name	Target 1				<input checked="" type="checkbox"/>
Target2Name	PLC1.MAIN.fbFeeders[0]::stHmi::stFeederRecipe::sTarget2Name	Target 2				<input checked="" type="checkbox"/>

4. Repeat steps for Feeder2
5. Add 2 folders under Recipes and add one Default recipe for each FeederType:

Enter a search term...

Name	Based on Recipe Type
Feeder1	
DefaultFeeder1	Feeder1
Feeder2	
DefaultFeeder2	Feeder2

6. Add Recipe Edit control on Recipe page:

Toolbox

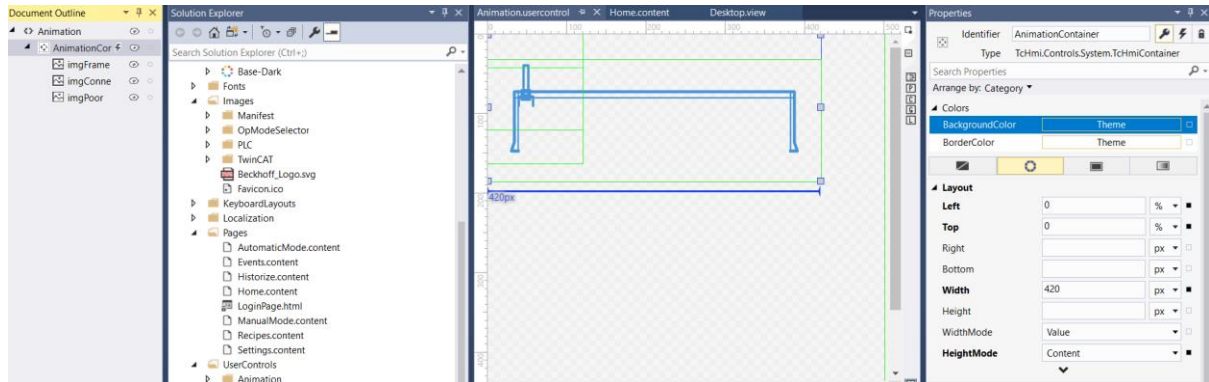
Search Toolbox

	Linear Gauge
	Localization Select
	Numeric Input
	Object Browser
	Password Input
	Radial Gauge
	Radio Button
	Recipe Edit
	Recipe Select
	Spinbox Input

This control allows to modify, add, delete and activate recipes on runtime

Animation User Control

1. Add folder under UserControls and add images from the "Images" folder in the repo
2. Add container with the following Layout settings:



3. Create UC parameters:

Edit/Define Parameters

Description:

Parameters:

Name	Datatype	(Drop) Default Value	DefaultValueInternal	Bindable	Visibility
Xpos	Number			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Ypos	Number			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

4. Add frame, connection, poor images with the following layout:

Identifier:

Type:

Search Properties

Arrange by: Category

BorderColor:

Layout

Left: px

Top: px

Right: px

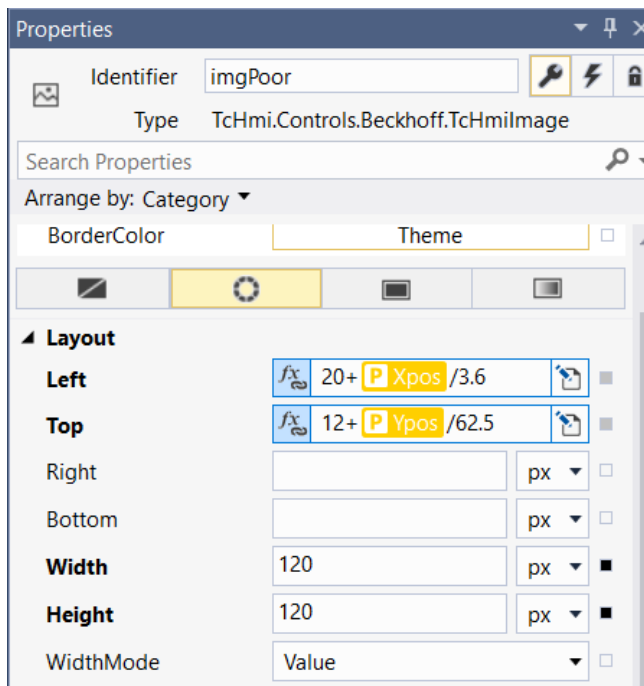
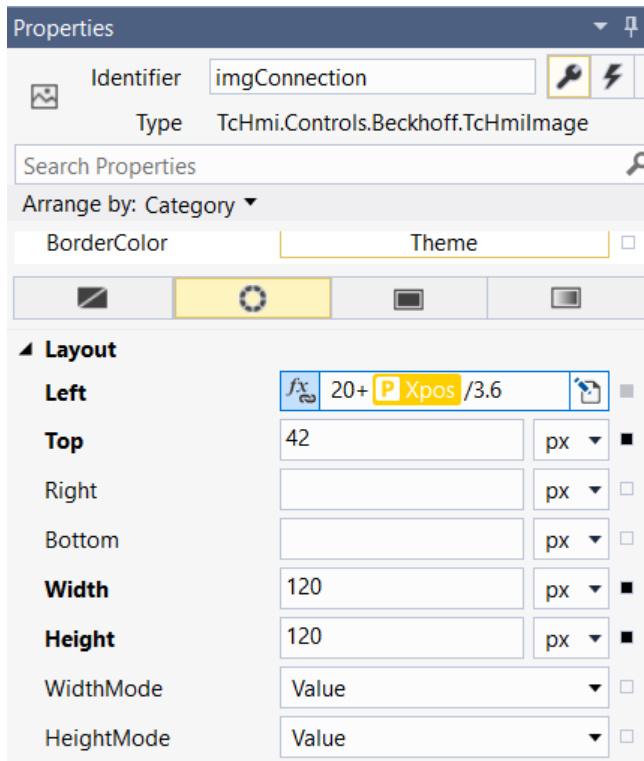
Bottom: px

Width: px

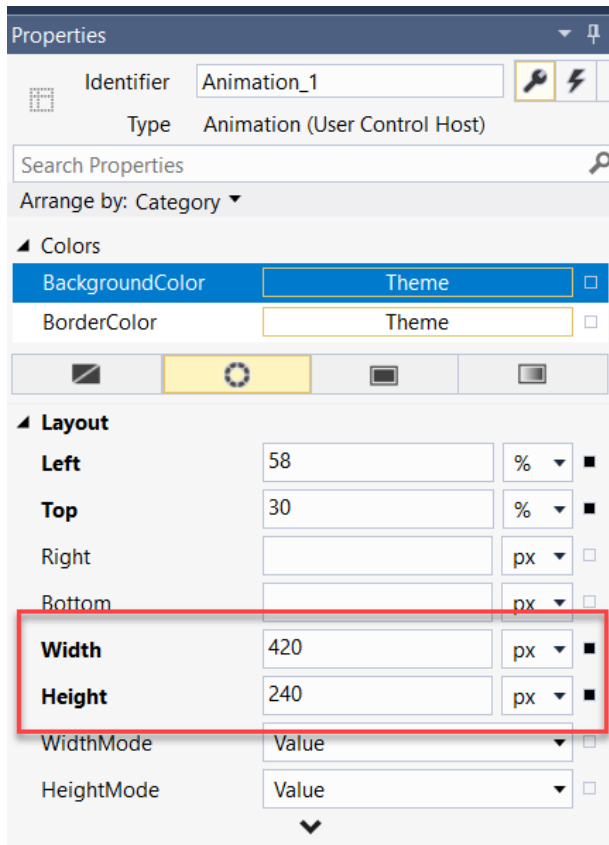
Height: px

WidthMode:

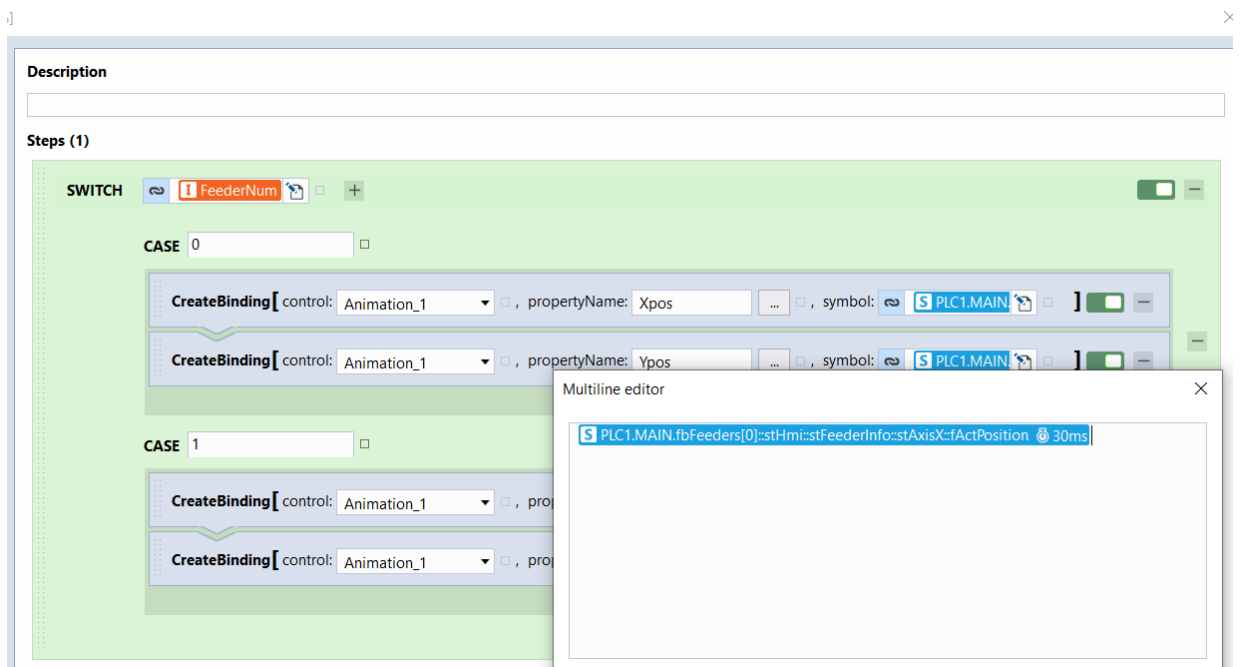
HeightMode:



5. Add one instance of the Animation UC on the Home page with the following layout properties:



6. Add trigger event for feeder change and configure actions to bind X and Y positions from Feeder 1 or 2:



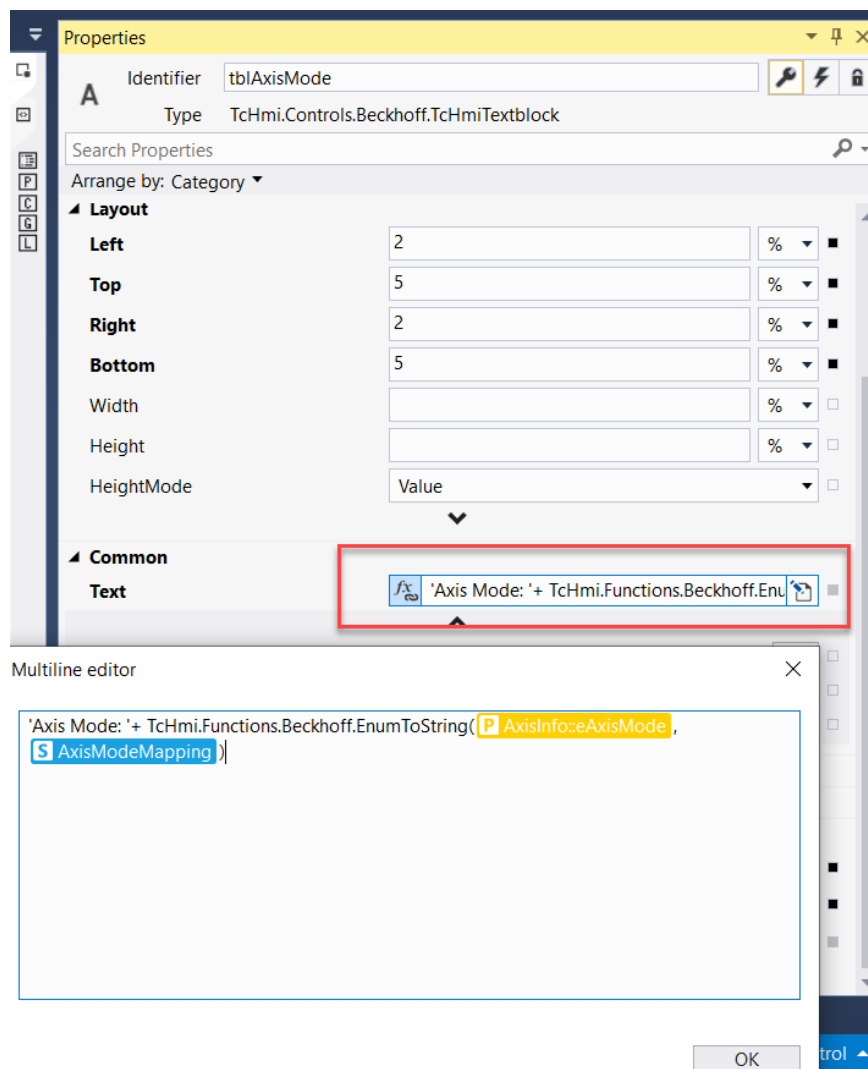
Enum Descriptions

TcHmi offers a function to convert Enum numbers to strings in the HMI. For this, we need to create a "Mapping variable"

1. Create server symbol 'AxisModeMapping' of type EnumMapping with Default value:

```
{  
  "0": "INIT",  
  "1": "ENABLE",  
  "2": "JOG",  
  "3": "POSITIONING",  
  "4": "STOP",  
  "5": "RESET",  
  "6": "ERROR"  
}
```

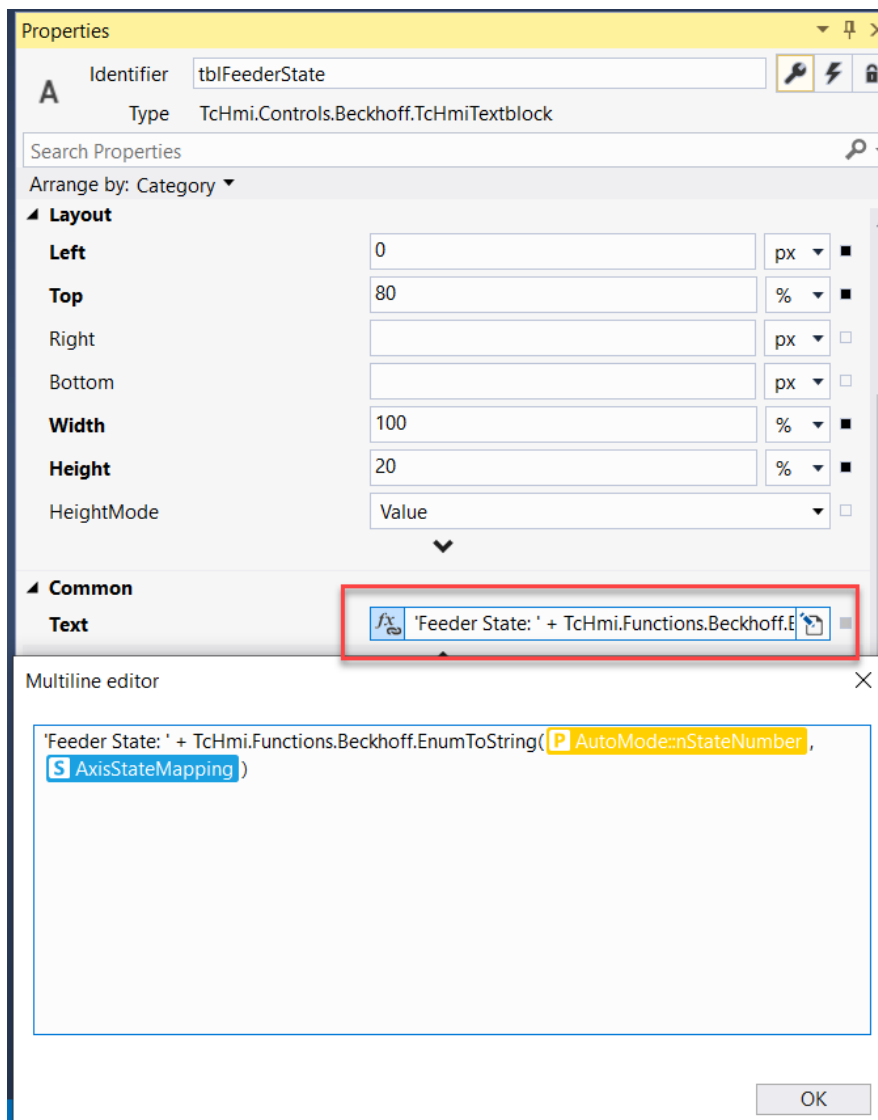
Bind Axis mode textblock in Automatic base UC:



2. Create server symbol 'AutoStateMapping' of type EnumMapping with Default value:

```
{  
    "0": "Reset",  
    "1": "MoveBackToTop",  
    "2": "MoveToTarget1PosX",  
    "3": "MoveToTarget1PosY",  
    "4": "Grab",  
    "5": "MoveBackToTop",  
    "6": "MoveToTarget2PosX",  
    "7": "MoveToTarget2PosY",  
    "8": "Place"  
}
```

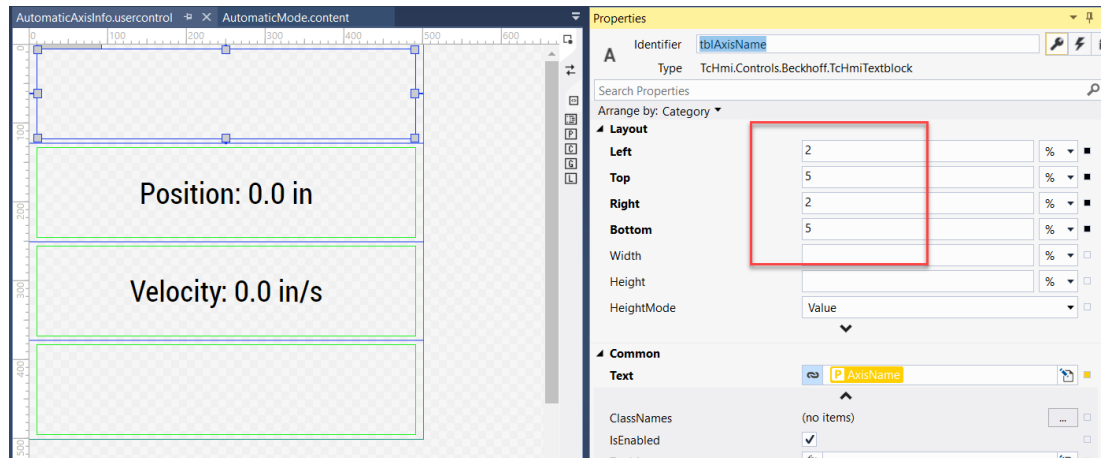
Bind FeederAutomatic UC:



Appendix – Build User Controls (UCs)

Create Automatic Axis Info UC

1. Grid centered inside UC host with 4 rows
2. Add textblock inside each UC – rename each textblock to AxisName, Position, Velocity, AxisMode:



3. Create 2 parameters for UC – AxisName (String), AxisInfo (PLC.ST_HmiAxisInfo)

Edit/Define Parameters

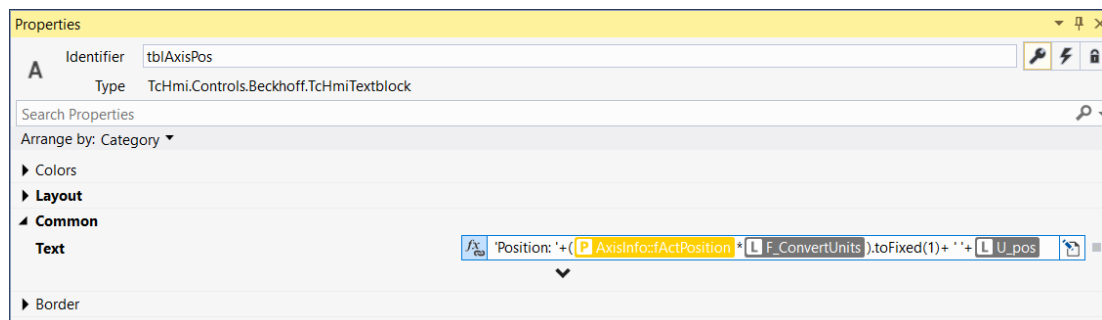
Description:

Parameters:

Name	Datatype	(Drop) Default Value	DefaultValueInternal	Bindable	Visibility
AxisName	G String			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
AxisInfo	S PLC1.ST_HmiA	(Object)	(Object)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

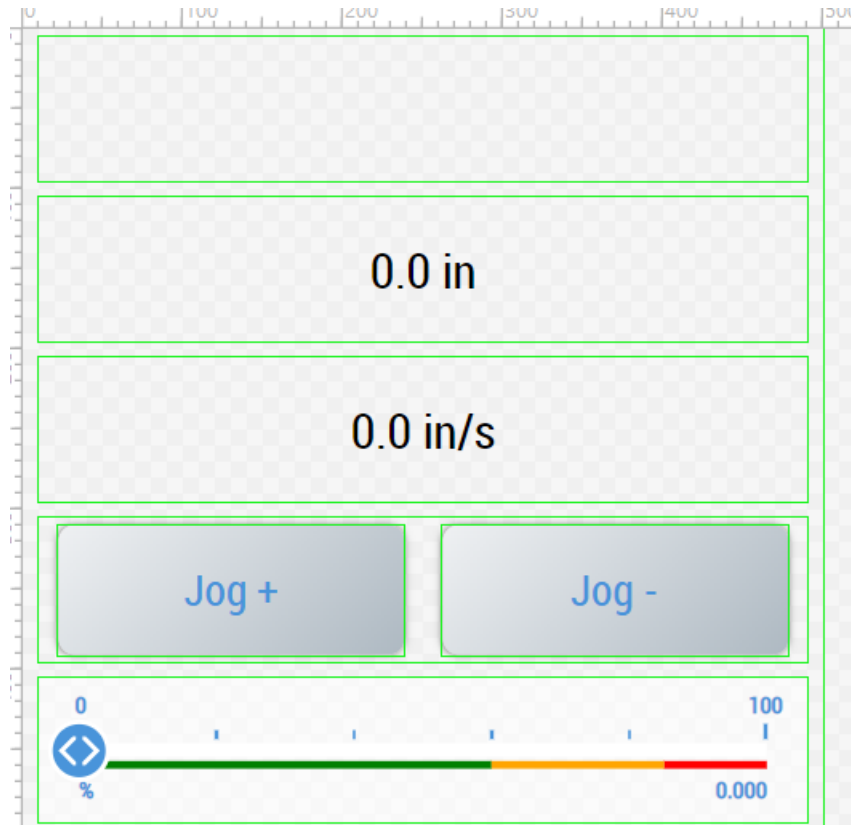
tchmi:server#/definitions/PLC1.ST_HmiAxisInfo

4.
 - a. Bind Axis Name parameter to “Text” attribute.
 - b. Create similar binding of sub-variables from AxisInfo parameter with other textblocks in the UC host. Make it a function binding and add raw text, F_ConvertUnits, U_Pos,.toFixed(1) for showing only one decimal point:



Create Manual Axis UC

1. Grid centered inside UC host with 5 rows
2. Add textblocks in first 3 rows, grid with two columns in the fourth cell, linear gauge in 5th cell:






3. Create 6 parameters for UC – JogPlus, JogMinus, PositionFeedback, VelocityFeedback, VelocityCmd, AxisName:

Edit/Define Parameters


Description: <input type="text"/>						
Parameters:						
Name	Datatype	(Drop) Default Value	DefaultValueInternal	Bindable	Visibility	
AxisName	String			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
JogPlus	BOOL	False ▼	False ▼	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
JogMinus	BOOL	False ▼	False ▼	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
VelocityCmd	Number			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
PositionFeedback	Number			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
VelocityFeedback	Number			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

4. Bind from parameters to Axis Name, Axis ActPos (use localization symbol), Axis ActVelo 'Text' attribute
5. Bind from parameters to Jog + and – state symbol attribute
6. Create two-way binding with LinearGauge “value” and set range for display:

Properties

Identifier   

Type TcHmi.Controls.Beckhoff.TcHmiLinearGauge

Search Properties 

Arrange by: Category ▾

BorderColor Ineme

Layout

Left % ▾ ■

Top % ▾ ■




Right % ▾ ■

Bottom % ▾ ■

Width % ▾ □

Height % ▾ □

Common

Value   

MinValue □

MaxValue ■

SetPoint □

Editable ☒ ■

IgnoreInvalidValues ☐ □


IgSetVelocity | Range

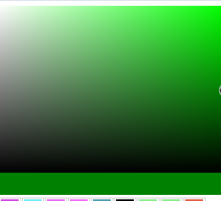
Elements

Start	End	Color
0	60	#FF008000
60	85	#FFFFA500
85	100	#FFFF0000

Properties

Colors

Color 



R 0
G 128
B 0
A 255

#FF008000




General

Start □


End □

OK Cancel

Properties

Identifier   

Type TcHmi.Controls.Beckhoff.TcHmiLinearGauge

Search Properties 

Arrange by: Category ▾


Editable ☒ ■


IgnoreInvalidValues ☐ □

Border

Background Image

Value

Step  ■

Range  ■

ShowValueText ☒ ▾

Labels

ShowLabels ☒ ■


LabelRange ■

LabelPosition ▾ □

Ticks

ShowTicks ☒ ▾

Unit

Unit  ■

UnitOrientation ▾ □

Create FeederAutomatic UC

1. Add 2 parameters:

Edit/Define Parameters

Description:

Parameters:

Name	Datatype	(Drop) Default Value	DefaultValueInternal	Bindable	Visibility
FeederData	S PLC1.ST_HmiFeederInfo	(Object) ...	(Object) ...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
AutoMode	S PLC1.ST_HmiAutomaticMode	(Object) ...	(Object) ...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

2. Add two instances of automatic axis Info UC and map their "parameter attributes" to X and Y AxisInfo parameters from parent UC:

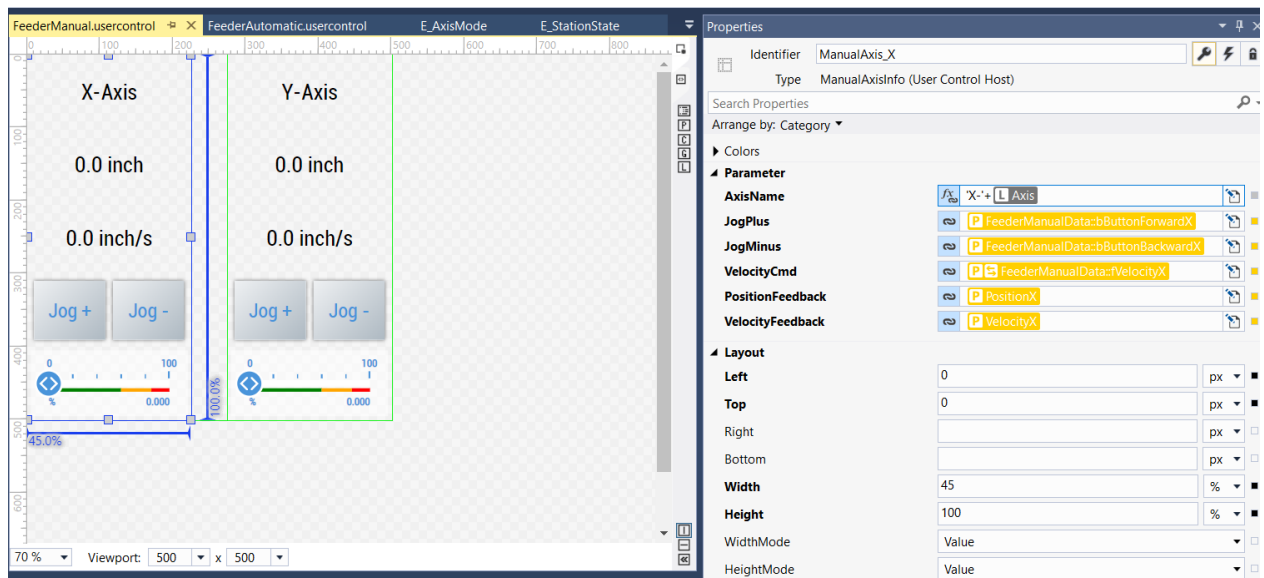
The design view shows two vertical panels labeled 'X-Axis' and 'Y-Axis'. Each panel contains two text elements: 'Position: 0.0 in' and 'Velocity: 0.0 in/s'. The Properties window on the right is open for the 'AxisInfo.X' control. Under the 'Parameter' section, 'AxisName' is set to 'X-+ L Axis' and 'AxisInfo' is set to 'FeederData.stAxisX'. Under the 'Layout' section, the 'Left' and 'Top' properties are set to 0, 'Right' is empty, 'Bottom' is empty, 'Width' is 45, and 'Height' is 80.

3. Add textblock in bottom 20% of space:

The design view shows the 'Feeder Automatic' user control with two vertical panels for 'X-Axis' and 'Y-Axis'. Each panel displays 'Position: 0.0 inch' and 'Velocity: 0.0 inch/s'. Below these panels, a text block labeled 'tblFeederState' displays 'Feeder State: 0'. The Properties window on the right is open for 'tblFeederState'. Under the 'Colors' section, 'TextColor', 'BackgroundColor', and 'BorderColor' are all set to 'Theme'. Under the 'Layout' section, 'Left' is 0, 'Top' is 80, 'Right' is empty, 'Bottom' is empty, 'Width' is 100, and 'Height' is 20. Under the 'Common' section, the 'Text' property is set to 'Feeder State: ' + AutoMode.stStateNumber'.

Create FeederManual UC

1. Add two instances of manual axis UC



2. Add 5 parameters – FeederData (ST_HmiFeederInfo), PosX and PosY (numbers), VeloX and VeloY (numbers)

Edit/Define Parameters

Description:							
Parameters:							
Name	Datatype	(Drop) Default Value	DefaultValueInternal	Bindable	Visibility	Category	
FeederManualData	S PLC1.ST_HmiM	(Object)	(Object)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
PositionX	G Number	tchmi:server#/definitions/PLC1.ST_HmiManualMode		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
PositionY	G Number			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
VelocityX	G Number			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
VelocityY	G Number			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

3. Bind nested UC parameters to parent UC

