

1. Introduction
 - a. Show Power Point "Framework Control - Dynamic Control Generation"
 - b. Create TwinCAT HMI controls dynamically and remove them
 - c. Show final solution
 - i. TcHmiButton is generated if user presses on another button
 - ii. Another button removes the button again
 - iii. Reaction on the dynamic added TcHmiButton with an "onPressed" event
2. Create new Framework Control "TcHmiDynamicControls"
3. Template.html:

4. Style.css on project layer:

```
16 .tc-hmi-dynamic-controls-button-create {
17     position: absolute;
18     width: 100px;
19     height: 50px;
20     top: 0px;
21     left: 0px;
22 }
23
24 .tc-hmi-dynamic-controls-button-remove {
25     position: absolute;
26     width: 100px;
27     height: 50px;
28     top: 0px;
29     right: 0px;
30 }
```

5. Build project, link project with TcHmi project, instantiate the Framework Control
6. Source.js
 - a. Member variables:

```

58 function TcHmiDynamicControls(element, pcElement, attrs) {
59     /** Call base constructor */
60     _super.call(this, element, pcElement, attrs);
61
62     // initialize member variables
63     this.__createdButton = undefined;
64     this.__destroyCreatedButtonOnPressed = undefined;
65 }

```

2. Prev-Init: Grab HTML elements:

```

70 TcHmiDynamicControls.prototype.__previnit = function () {
71     /** Handle template elements. Should be done before call to __previnit of super class. */
72     this.__elementTemplateRoot = this.__element.find('.tc-hmi-dynamic-controls-template');
73     this.__elementContainer = this.__elementTemplateRoot.find('.tc-hmi-dynamic-controls-container');
74     this.__buttonCreate = this.__elementContainer.find('.tc-hmi-dynamic-controls-button-create');
75     this.__buttonRemove = this.__elementContainer.find('.tc-hmi-dynamic-controls-button-remove');

```

c. Add internal function "__generateButton" with the use of TcHmi.ControlFactory:

```

150 // internal function for the button generation
151 TcHmiDynamicControls.prototype.__generateButton = function ($this) {
152
153     // check if the button already exists
154     if ($this.__createdButton) {
155         return;
156     }
157
158     // call control factory to generate button
159     $this.__createdButton = TcHmi.ControlFactory.createEx(
160         'tchmi-button',
161         $this.getId() + '.MyButton',
162         {
163             'data-tchmi-bottom': 0,
164             'data-tchmi-left': 0,
165             'data-tchmi-width': 100,
166             'data-tchmi-width-unit': '%',
167             'data-tchmi-height': 50,
168             'data-tchmi-text': 'Generated Button',
169             'data-tchmi-background-color': {
170                 'color': 'rgba(55, 55, 55, 1)'
171             }
172         },
173         // add the control a a child of the Framework Control
174         $this
175     );

```

```

177 // check if the container element and the button are valid
178 if ($this.__elementContainer && $this.__createdButton) {
179     // append the control to the container element
180     $this.__elementContainer.append($this.__createdButton.getElement());
181 }
182
183 // register event listener for new control
184 $this.__destroyCreatedButtonOnPressed = TcHmi.EventProvider.register(
185     $this.__createdButton.getId() + '.onPressed',
186     function (e, data) {
187         alert("Created Button was pressed");
188     });
189 };

```

d. Add internal function "__removeButton":

```

191 // internal function to remove the button
192 TcHmiDynamicControls.prototype.__removeButton = function ($this) {
193
194     // check if the button exists
195     if ($this.__createdButton) {
196
197         // call the destroy function of the button
198         $this.__createdButton.destroy();
199
200         // set reference to undefined
201         $this.__createdButton = undefined;
202     }
203
204     // check if the destroy function exists
205     if ($this.__destroyCreatedButtonOnPressed) {
206         // destroy the event listener
207         $this.__destroyCreatedButtonOnPressed();
208
209         // set destroy function to undefined
210         $this.__destroyCreatedButtonOnPressed = undefined;
211     }
212 }
213 };

```

e. Scroll to attach function: Add event listeners for onClick events and call the internal functions with the \$this flag

```

102 // reference to current control instance for other scopes
103 var $this = this;
104
105 // register event listener for create button
106 this.__buttonCreate.on('click', function (e) {
107     $this.__generateButton($this);
108 });
109
110 // register event listener for remove button
111 this.__buttonRemove.on('click', function (e) {
112     $this.__removeButton($this);
113 });

```

f. Detach function: Remove event listeners and destroy created button

```

129 // remove event listeners
130 this.__buttonCreate.off('click');
131 this.__buttonRemove.off('click');
132
133 // destroy created button and remove event listeners
134 this.__removeButton(this);

```

7. Build project, reload Desktop.view and start LiveView

- a. Generate button
- b. Test onClick event
- c. Remove button
- d. etc.