



Lanka Nippon Biztech Institute

Web Component

CMP120230 – Web Programming

Name : Gayan Dhananjaya

Index Number : UGC0122020

Batch : SE - 01

# OUTLINE

---

1.Introduction	.....	3
2.Layout	.....	4
3.Usage	.....	5
4.Component Diagram	.....	6
5.References	.....	7

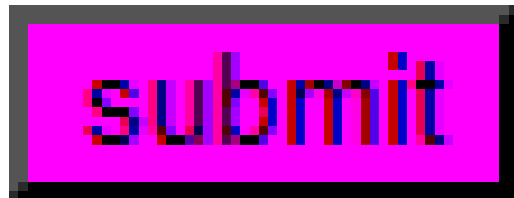
# INTRODUCTION

*Colorful button web component*

“Colorful button” web component is a tool and ready-made solution that helps you add a button color changing feature to your website or application. Allows developers to customize the appearance of the language selector through the associated CSS styles, providing flexibility in adaption the component to match the overall design of the web application.

The `<custom-button>` element is defined as a custom web component using the `customElements.define()` method. It takes two arguments: the name of the custom element (`'custom-button'`) and the class that defines its behavior (`customButton`).

# LAYOUT



*This is the code part that I implemented.*

# HTML

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <style>
7     .custom-button {
8       display: inline-block;
9       padding: 10px 20px;
10      font-size: 16px;
11      cursor: pointer;
12    }
13  </style>
14 </head>
15 <body>
16   <custom-button text="submit" color="#ff00ff"></custom-button>
17
18   <script src="componant.js"></script>
19
20 </body>
21 </html>
```

# JavaScript

```

1
2 document.addEventListener('DOMContentLoaded', function () {
3   class CustomButton extends HTMLElement {
4     constructor() {
5       super();
6
7       this.attachShadow({ mode: 'open' });
8
9       const template = document.createElement('template');
10      template.innerHTML = `
11        <button class="custom-button"></button>
12      `;
13
14      this.shadowRoot.appendChild(template.content.cloneNode(true));
15
16      this.buttonElement = this.shadowRoot.querySelector('.custom-button');
17      this.buttonElement.addEventListener('click', this.handleClick.bind(this));
18    }
19
20    connectedCallback() {
21      this.updateButton();
22    }
23
24    static get observedAttributes() {
25      return ['text', 'color'];
26    }
27
28    attributeChangedCallback(name, oldValue, newValue) {
29      if (oldValue !== newValue) {
30        this.updateButton();
31      }
32    }
33  }
34
35  // Register the custom element
36  customElements.define('custom-button', CustomButton);
37
38  // Create and append the custom button
39  const button = document.createElement('custom-button');
40  document.body.appendChild(button);
41
42  // Set initial text and color
43  button.setAttribute('text', 'Click Me');
44  button.setAttribute('color', 'blue');
45
46  // Add a click event listener to the button
47  button.addEventListener('click', () => {
48    alert('Button clicked!');
49  });
50
51  // Add a text input field and a color picker
52  const textInput = document.createElement('input type="text"');
53  const colorPicker = document.createElement('input type="color"');
54
55  // Append the input fields to the document body
56  document.body.appendChild(textInput);
57  document.body.appendChild(colorPicker);
58
59  // Add event listeners to the input fields
60  textInput.addEventListener('input', () => {
61    button.setAttribute('text', textInput.value);
62  });
63
64  colorPicker.addEventListener('input', () => {
65    button.setAttribute('color', colorPicker.value);
66  });
67
68  // Add a "Reset" button
69  const resetButton = document.createElement('button');
70  resetButton.textContent = 'Reset';
71  document.body.appendChild(resetButton);
72
73  // Add a click event listener to the Reset button
74  resetButton.addEventListener('click', () => {
75    button.setAttribute('text', 'Click Me');
76    button.setAttribute('color', 'blue');
77  });
78
79  // Add a "Toggle" button
80  const toggleButton = document.createElement('button');
81  toggleButton.textContent = 'Toggle';
82  document.body.appendChild(toggleButton);
83
84  // Add a click event listener to the Toggle button
85  toggleButton.addEventListener('click', () => {
86    // Toggle the 'active' attribute
87    button.setAttribute('active', !button.hasAttribute('active'));
88  });
89
90  // Add a "Style" button
91  const styleButton = document.createElement('button');
92  styleButton.textContent = 'Style';
93  document.body.appendChild(styleButton);
94
95  // Add a click event listener to the Style button
96  styleButton.addEventListener('click', () => {
97    // Toggle the 'style' attribute
98    button.setAttribute('style', !button.hasAttribute('style'));
99  });
100
101  // Add a "Color" button
102  const colorButton = document.createElement('button');
103  colorButton.textContent = 'Color';
104  document.body.appendChild(colorButton);
105
106  // Add a click event listener to the Color button
107  colorButton.addEventListener('click', () => {
108    // Toggle the 'color' attribute
109    button.setAttribute('color', !button.hasAttribute('color'));
110  });
111
112  // Add a "Text" button
113  const textButton = document.createElement('button');
114  textButton.textContent = 'Text';
115  document.body.appendChild(textButton);
116
117  // Add a click event listener to the Text button
118  textButton.addEventListener('click', () => {
119    // Toggle the 'text' attribute
120    button.setAttribute('text', !button.hasAttribute('text'));
121  });
122
123  // Add a "Size" button
124  const sizeButton = document.createElement('button');
125  sizeButton.textContent = 'Size';
126  document.body.appendChild(sizeButton);
127
128  // Add a click event listener to the Size button
129  sizeButton.addEventListener('click', () => {
130    // Toggle the 'size' attribute
131    button.setAttribute('size', !button.hasAttribute('size'));
132  });
133
134  // Add a "Font" button
135  const fontButton = document.createElement('button');
136  fontButton.textContent = 'Font';
137  document.body.appendChild(fontButton);
138
139  // Add a click event listener to the Font button
140  fontButton.addEventListener('click', () => {
141    // Toggle the 'font' attribute
142    button.setAttribute('font', !button.hasAttribute('font'));
143  });
144
145  // Add a "Background" button
146  const backgroundButton = document.createElement('button');
147  backgroundButton.textContent = 'Background';
148  document.body.appendChild(backgroundButton);
149
150  // Add a click event listener to the Background button
151  backgroundButton.addEventListener('click', () => {
152    // Toggle the 'background' attribute
153    button.setAttribute('background', !button.hasAttribute('background'));
154  });
155
156  // Add a "Border" button
157  const borderButton = document.createElement('button');
158  borderButton.textContent = 'Border';
159  document.body.appendChild(borderButton);
160
161  // Add a click event listener to the Border button
162  borderButton.addEventListener('click', () => {
163    // Toggle the 'border' attribute
164    button.setAttribute('border', !button.hasAttribute('border'));
165  });
166
167  // Add a "Shadow" button
168  const shadowButton = document.createElement('button');
169  shadowButton.textContent = 'Shadow';
170  document.body.appendChild(shadowButton);
171
172  // Add a click event listener to the Shadow button
173  shadowButton.addEventListener('click', () => {
174    // Toggle the 'shadow' attribute
175    button.setAttribute('shadow', !button.hasAttribute('shadow'));
176  });
177
178  // Add a "Opacity" button
179  const opacityButton = document.createElement('button');
180  opacityButton.textContent = 'Opacity';
181  document.body.appendChild(opacityButton);
182
183  // Add a click event listener to the Opacity button
184  opacityButton.addEventListener('click', () => {
185    // Toggle the 'opacity' attribute
186    button.setAttribute('opacity', !button.hasAttribute('opacity'));
187  });
188
189  // Add a "Z-index" button
190  const zIndexButton = document.createElement('button');
191  zIndexButton.textContent = 'Z-index';
192  document.body.appendChild(zIndexButton);
193
194  // Add a click event listener to the Z-index button
195  zIndexButton.addEventListener('click', () => {
196    // Toggle the 'z-index' attribute
197    button.setAttribute('z-index', !button.hasAttribute('z-index'));
198  });
199
200  // Add a "Position" button
201  const positionButton = document.createElement('button');
202  positionButton.textContent = 'Position';
203  document.body.appendChild(positionButton);
204
205  // Add a click event listener to the Position button
206  positionButton.addEventListener('click', () => {
207    // Toggle the 'position' attribute
208    button.setAttribute('position', !button.hasAttribute('position'));
209  });
210
211  // Add a "Transform" button
212  const transformButton = document.createElement('button');
213  transformButton.textContent = 'Transform';
214  document.body.appendChild(transformButton);
215
216  // Add a click event listener to the Transform button
217  transformButton.addEventListener('click', () => {
218    // Toggle the 'transform' attribute
219    button.setAttribute('transform', !button.hasAttribute('transform'));
220  });
221
222  // Add a "Filter" button
223  const filterButton = document.createElement('button');
224  filterButton.textContent = 'Filter';
225  document.body.appendChild(filterButton);
226
227  // Add a click event listener to the Filter button
228  filterButton.addEventListener('click', () => {
229    // Toggle the 'filter' attribute
230    button.setAttribute('filter', !button.hasAttribute('filter'));
231  });
232
233  // Add a "Cursor" button
234  const cursorButton = document.createElement('button');
235  cursorButton.textContent = 'Cursor';
236  document.body.appendChild(cursorButton);
237
238  // Add a click event listener to the Cursor button
239  cursorButton.addEventListener('click', () => {
240    // Toggle the 'cursor' attribute
241    button.setAttribute('cursor', !button.hasAttribute('cursor'));
242  });
243
244  // Add a "Pointer-events" button
245  const pointerEventsButton = document.createElement('button');
246  pointerEventsButton.textContent = 'Pointer-events';
247  document.body.appendChild(pointerEventsButton);
248
249  // Add a click event listener to the Pointer-events button
250  pointerEventsButton.addEventListener('click', () => {
251    // Toggle the 'pointer-events' attribute
252    button.setAttribute('pointer-events', !button.hasAttribute('pointer-events'));
253  });
254
255  // Add a "User-select" button
256  const userSelectButton = document.createElement('button');
257  userSelectButton.textContent = 'User-select';
258  document.body.appendChild(userSelectButton);
259
260  // Add a click event listener to the User-select button
261  userSelectButton.addEventListener('click', () => {
262    // Toggle the 'user-select' attribute
263    button.setAttribute('user-select', !button.hasAttribute('user-select'));
264  });
265
266  // Add a "Text-align" button
267  const textAlignButton = document.createElement('button');
268  textAlignButton.textContent = 'Text-align';
269  document.body.appendChild(textAlignButton);
270
271  // Add a click event listener to the Text-align button
272  textAlignButton.addEventListener('click', () => {
273    // Toggle the 'text-align' attribute
274    button.setAttribute('text-align', !button.hasAttribute('text-align'));
275  });
276
277  // Add a "Text-decoration" button
278  const textDecorationButton = document.createElement('button');
279  textDecorationButton.textContent = 'Text-decoration';
280  document.body.appendChild(textDecorationButton);
281
282  // Add a click event listener to the Text-decoration button
283  textDecorationButton.addEventListener('click', () => {
284    // Toggle the 'text-decoration' attribute
285    button.setAttribute('text-decoration', !button.hasAttribute('text-decoration'));
286  });
287
288  // Add a "Text-decoration-color" button
289  const textDecorationColorButton = document.createElement('button');
290  textDecorationColorButton.textContent = 'Text-decoration-color';
291  document.body.appendChild(textDecorationColorButton);
292
293  // Add a click event listener to the Text-decoration-color button
294  textDecorationColorButton.addEventListener('click', () => {
295    // Toggle the 'text-decoration-color' attribute
296    button.setAttribute('text-decoration-color', !button.hasAttribute('text-decoration-color'));
297  });
298
299  // Add a "Text-decoration-line" button
300  const textDecorationLineButton = document.createElement('button');
301  textDecorationLineButton.textContent = 'Text-decoration-line';
302  document.body.appendChild(textDecorationLineButton);
303
304  // Add a click event listener to the Text-decoration-line button
305  textDecorationLineButton.addEventListener('click', () => {
306    // Toggle the 'text-decoration-line' attribute
307    button.setAttribute('text-decoration-line', !button.hasAttribute('text-decoration-line'));
308  });
309
310  // Add a "Text-decoration-style" button
311  const textDecorationStyleButton = document.createElement('button');
312  textDecorationStyleButton.textContent = 'Text-decoration-style';
313  document.body.appendChild(textDecorationStyleButton);
314
315  // Add a click event listener to the Text-decoration-style button
316  textDecorationStyleButton.addEventListener('click', () => {
317    // Toggle the 'text-decoration-style' attribute
318    button.setAttribute('text-decoration-style', !button.hasAttribute('text-decoration-style'));
319  });
320
321  // Add a "Text-decoration-width" button
322  const textDecorationWidthButton = document.createElement('button');
323  textDecorationWidthButton.textContent = 'Text-decoration-width';
324  document.body.appendChild(textDecorationWidthButton);
325
326  // Add a click event listener to the Text-decoration-width button
327  textDecorationWidthButton.addEventListener('click', () => {
328    // Toggle the 'text-decoration-width' attribute
329    button.setAttribute('text-decoration-width', !button.hasAttribute('text-decoration-width'));
330  });
331
332  // Add a "Text-decoration-color" button
333  const textDecorationColorButton = document.createElement('button');
334  textDecorationColorButton.textContent = 'Text-decoration-color';
335  document.body.appendChild(textDecorationColorButton);
336
337  // Add a click event listener to the Text-decoration-color button
338  textDecorationColorButton.addEventListener('click', () => {
339    // Toggle the 'text-decoration-color' attribute
340    button.setAttribute('text-decoration-color', !button.hasAttribute('text-decoration-color'));
341  });
342
343  // Add a "Text-decoration-line" button
344  const textDecorationLineButton = document.createElement('button');
345  textDecorationLineButton.textContent = 'Text-decoration-line';
346  document.body.appendChild(textDecorationLineButton);
347
348  // Add a click event listener to the Text-decoration-line button
349  textDecorationLineButton.addEventListener('click', () => {
350    // Toggle the 'text-decoration-line' attribute
351    button.setAttribute('text-decoration-line', !button.hasAttribute('text-decoration-line'));
352  });
353
354  // Add a "Text-decoration-style" button
355  const textDecorationStyleButton = document.createElement('button');
356  textDecorationStyleButton.textContent = 'Text-decoration-style';
357  document.body.appendChild(textDecorationStyleButton);
358
359  // Add a click event listener to the Text-decoration-style button
360  textDecorationStyleButton.addEventListener('click', () => {
361    // Toggle the 'text-decoration-style' attribute
362    button.setAttribute('text-decoration-style', !button.hasAttribute('text-decoration-style'));
363  });
364
365  // Add a "Text-decoration-width" button
366  const textDecorationWidthButton = document.createElement('button');
367  textDecorationWidthButton.textContent = 'Text-decoration-width';
368  document.body.appendChild(textDecorationWidthButton);
369
370  //
```

```

33
34     updateButton() {
35         const buttonText = this.getAttribute('text') || 'Click me';
36         const buttonColor = this.getAttribute('color') || '#3498db';
37
38         this.buttonElement.textContent = buttonText;
39         this.buttonElement.style.backgroundColor = buttonColor;
40     }
41
42     handleButtonClick() {
43         alert('Button clicked!');
44         // You can perform additional actions here when the button is clicked.
45     }
46 }
47
48 customElements.define('custom-button', CustomButton);
49 });
50

```

## USAGE

In the HTML body, a `<custom-button>` element is used with the attributes `text="customText"` and `color="#e74c3c"`. This creates an instance of the custom button with the specified text and color.

- Customizable Styling

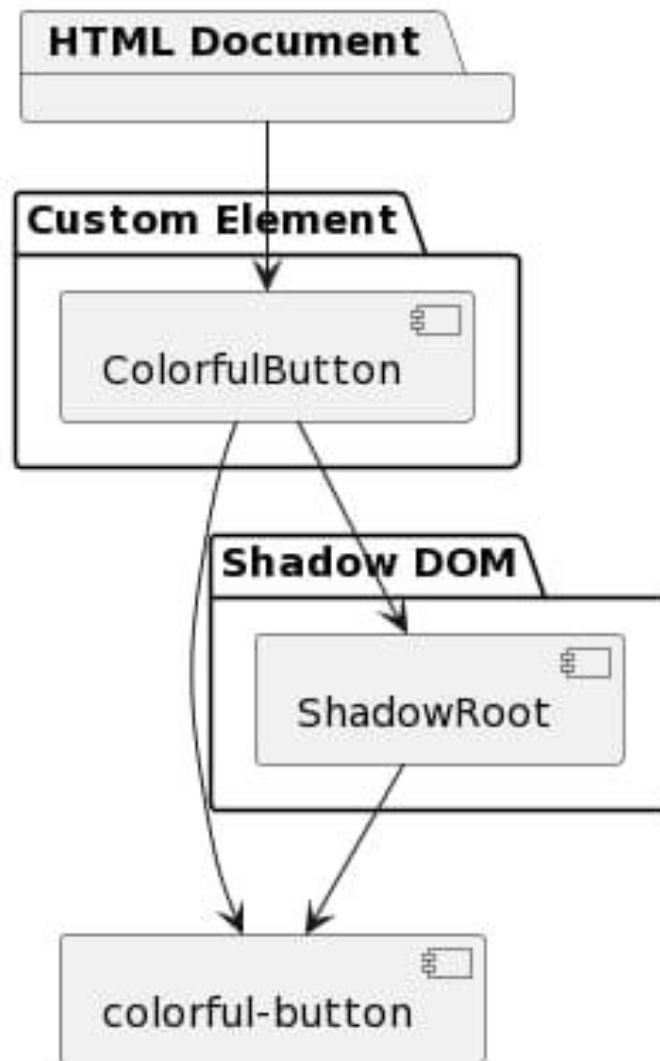
Allows developers to customize the appearance of the language selector through the associated CSS styles, providing flexibility in adapting the component to match the overall design of the web application.

- **Encapsulation**

The button's internal structure and styles are encapsulated within the Shadow DOM, preventing external styles from affecting its appearance and ensuring better isolation from the rest of the document.



# Component Diagram



# REFERENCES

- ✓ [https://developer.mozilla.org/en-US/docs/Web/API/Web\\_components](https://developer.mozilla.org/en-US/docs/Web/API/Web_components)
- ✓ <https://www.webcomponents.org/>