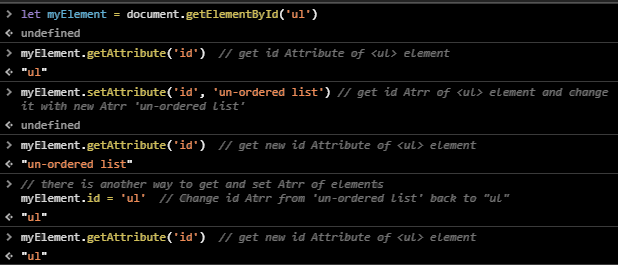
**Object’s Attributes & Content**

|  |  |
| --- | --- |
| Changing HTML Elements | |
| **element.outerHTML = new html content** |  |
| **element.style.property = new style** | Change the style of an HTML element |

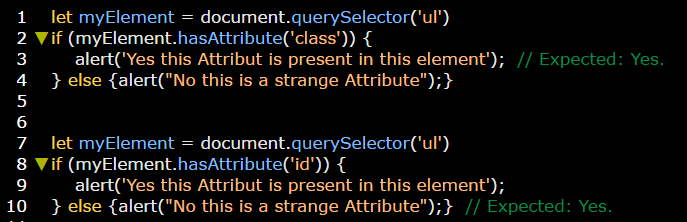
|  |  |
| --- | --- |
| **HTML** **Attributes of Elements** | |
| **Element.getAttribute(“Attr”)** | **Getting** attribute of a specific Html element |
| **Element.attribute = new value** | **Getting** and **setting** attribute value of an HTML element **in one step**. |
| **Element.setAttribute(“Attr name”, “Atrr value”)** | **Setting** attribute of a specific Html element |
| **Element.hasAttribute (“Attr”)** | **Checking** the presence of specific attribute. If it present **true** returns else **false** returns. |
| **document.createAttributeNode** | **Creating** new attribute for the element. |
| **AttributeName.value = “your value”** | **Setting** value to the attribute. |
| **Element.setAttributeNode(AttributeName)** | **Appending** attribute to specific element. |
| **Element.removeAttribute (“Attr”)** | **Removing** specific attribute from the object. |
| **Element.classList** | **Getting** all classes of specific HTML element. **[Not supported in IE9]** |
| **Element.classList.length** | **Getting** the length of classes list**.** |
| **Element.classList.contains(“class name”)** | **Checking** the presence of specific class on HTML element. If it present **true** returns else **false** returns. |
| **Element.classList.item(“index-number”)** | **Getting** class from the list according to its index-number. |
| **Element.classList = ‘class name’** | **Setting** new class for specific HTML element. |
| **Element.classList.add = “class | classes”** | **Adding** new class or classes for specific HTML element. |
| **Element.classList.remove = “class | classes”** | **Removing** class | classes from specific HTML element. |
| **Element.classList.toggle(‘class-name’);** | **Adding** class if it’s not present and **Removing** it if it present. **True** will returns when adding while **false** returns on removing. |

**Note:** **Element.hasAttribute (“Attr”) often used with if statement.**

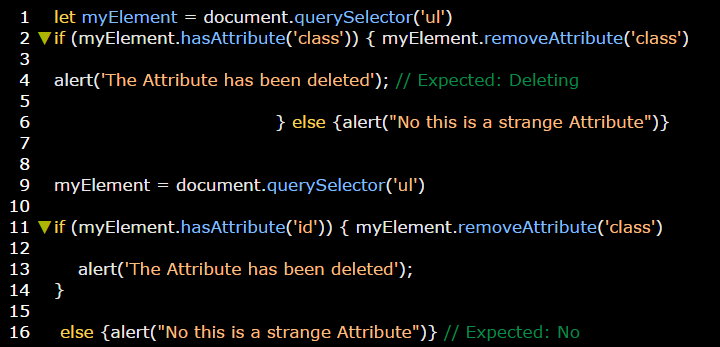
**Examples:**



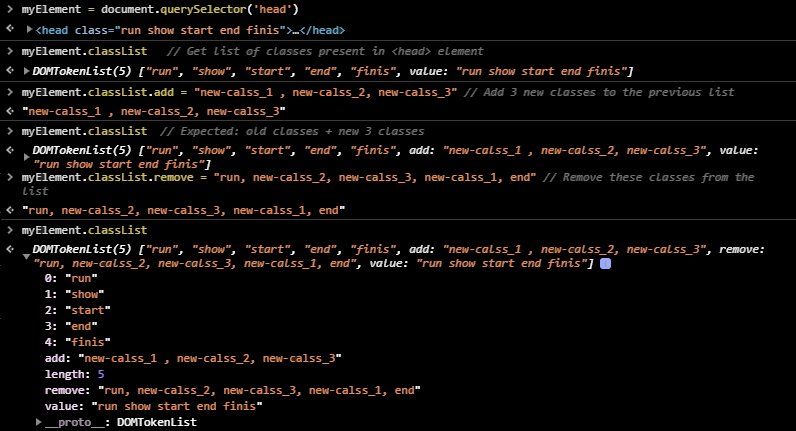
**Example:** on using **Element.hasAttribute (“Attr”) with if statement.**



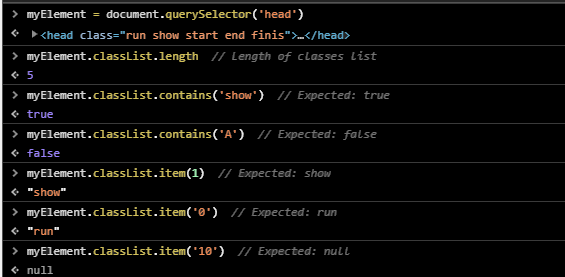
**Example:** on using **Element.removeAttribute (“Attr”).**



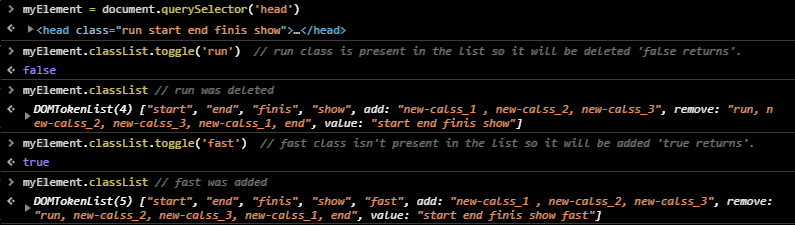
**Examples:** on class List.



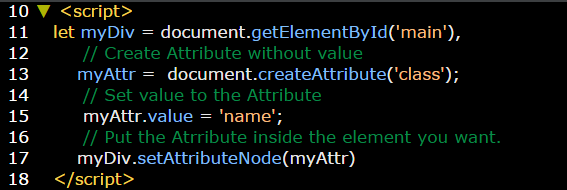
**Example:** on class list.



**Example:** on toggle.



**Example:** on adding new attribute to a specific element.



|  |  |
| --- | --- |
| **HTML** **Content of Elements** | |
| **Element.innerHTML** | **Getting** the inner **Content** (child elements & text) of an element |
| **Element.textContent** | **Getting** the text **Content** of an element |
| **Element.innerHTML = ‘new html content’** | **Setting** the inner HTML **Content** (text and html elements as tags) of an element |
| **Element.textContent = ‘new html content’** | **Setting** text **Content** of an element |
| **Element.InnerText = ‘new content’** | **Setting** text **Content** of an element. **[Not standard]** |
| **Element.outerText = ‘new content’** | **[Not standard]** |

**Note:**

1. **textContent** is best practice than **innerHTML** as the first one is fast (more performance as it deals with only texts) while the last one is slightly slow (as it deals with texts and tags) in addition to it may represent a gate to penetrate (XSS attack).
2. Using **&ls;** **&amp;** **&gr;** as html entities and use **innerHTML** with it, **&ls;** **&amp;** **&gr;** will returns as its.
3. Using **&ls;** **&amp;** **&gr;** as html entities and use **textContent** with it, **<&>** will returns.

**Example:**

