Current Skill DOM Manipulation

Styling DOM Elements

We can also apply style to HTML elements to change the visual presentation of HTML with the dynamic use of JavaScript.

We can set all the styles for the elements including fonts, colors, margins, borders, background images, text alignment, width and height, position, and so on.

```
<body>
  <p id="intro">This is a paragraph.</p>
  <p>This is another paragraph.</p>

  <script>
    // Selecting element
    var elem = document.getElementById('intro');

    // Appling styles on element
    elem.style.color = 'blue';
    elem.style.fontSize = '18px';
    elem.style.fontWeight = 'bold';
    console.log('elem:', elem);
  </script>
</body>
```

Output

```
elem:  <p id="intro" style="color: blue; font-size: 18px; font-weight: bold;">This is a paragraph.</p>
```

Adding Elements

We can explicitly create new elements in an HTML document using the document.createElement() method.

This method creates a new element but it doesn't add it to the DOM.

We will be doing that in a separate step, as shown in the following example:

```
<body>
  <div id="main">
    <h1 id="title">Hello World!</h1>
    <p id="hint">This is a simple paragraph.</p>
  </div>

  <script>
    // Creating a new div element
    var newDiv = document.createElement('div');
    // Creating a text node
    var newContent = document.createTextNode('Hi, how are you doing?');
    // Adding the text node to the newly created div
    newDiv.appendChild(newContent);
    // Adding the newly created element and its content into the DOM
    var currentDiv = document.getElementById('main');
    document.body.appendChild(newDiv, currentDiv);
  </script>
</body>
```

Output

Getting or Setting HTML Contents to DOM

We can get or set the contents of the HTML elements easily with the innerHTML property.

This property sets or gets the HTML markup contained within the element (i.e. content between its opening and closing tags). Check out the following example to see how it works:

```
<body>
  <div id="main">
    <h1 id="title">Hello World!</h1>
    <p id="hint">This is a simple paragraph.</p>
  </div>

  <script>
    // Getting inner HTML conents
    var contents = document.getElementById('main').innerHTML;
    console.log('contents', contents);

    // Setting inner HTML contents
    var mainDiv = document.getElementById('main');
    mainDiv.innerHTML = '<p>This is <em>newly inserted</em> paragraph.</p>';
  </script>
</body>
```

Output

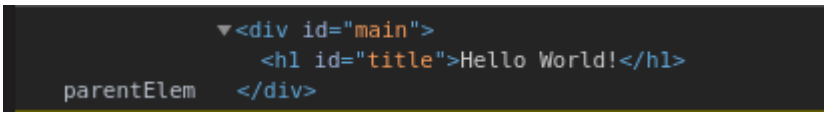
Remove Elements

Similarly, we can use the removeChild() method to remove a child node from the DOM.

This method also returns the removed node. Here's an example:

```
<body>
  <div id="main">
    <h1 id="title">Hello World!</h1>
    <p id="hint">This is a simple paragraph.</p>
  </div>
  <script>
    var parentElem = document.getElementById('main');
    var childElem = document.getElementById('hint');
    parentElem.removeChild(childElem);
    console.log("parentElem", parentElem)
  </script>
</body>
```

Output



Previous next