# insertAdjacentHTML/Text/Element:

This method is used to insert HTML. The first parameter is a code word, specifying where to insert. The second parameter is an HTML string.

## 1. beforebegin:

Inserts content immediately before the element.

**Example:**

```
<div id="referenceNode">Reference node</div>

<script>
    const refNode = document.getElementById('referenceNode');
    refNode.insertAdjacentHTML('beforebegin', '<p>Inserted before the reference node.</p>');
</script>
```

Output:

```
Inserted before the reference node.
Reference node
```

## 2. afterbegin:

Inserts content inside the element, at the beginning.

#### Example:

```
<div id="container">Existing content</div>

<script>
    const container = document.getElementById('container');
    container.insertAdjacentHTML('afterbegin', '<p>Inserted at the beginning inside the container.</p>');
</script>
```

Output:

```
Inserted at the beginning inside the container.
Existing content
```

## 3. beforeend:

Insert HTML into element at the end.

#### Example:

```
<div id="container">Existing content</div>

<script>
    const container = document.getElementById('container');
    container.insertAdjacentHTML('beforeend', '<p>Inserted at the end inside the container.</p>');
</script>
```

Output:

```
Existing content
```

```
Inserted at the end inside the container.
```

## 4. afterend:

Insert HTML immediately after element.

**Example:**

```
<div id="referenceNode">Reference node</div>

<script>
    const refNode = document.getElementById('referenceNode');
    refNode.insertAdjacentHTML('afterend', '<p>Inserted after the reference node.</p>');
</script>
```

Output:

```
Reference node

Inserted after the reference node.
```

**In each of the above four, you will be able to see the changes if you navigate to the code. Also the terms are self explanatory.**

# Node Removal:

It is achieved using -

## remove():

It is used to remove an element from the DOM. When this method is called on a DOM element, it removes the element from its parent node.

### Example:

```
<div id="container">
    <p id="paragraph">This is a paragraph that will be removed.</p>
    <p>This paragraph will remain.</p>
</div>

<script>
    const paragraph = document.getElementById('paragraph');
    paragraph.remove(); // Removes the paragraph element from the DOM
</script>
```

Output:

```
This paragraph will remain.
```

# className:

### Example:

```
<div id="myDiv" class="class1 class2">Hello</div>

<script>
    const myDiv = document.getElementById('myDiv');
    console.log(myDiv.className); // Outputs: "class1 class2"

    myDiv.className = "newClass"; // Replaces all classes with "newClass"
    console.log(myDiv.className); // Outputs: "newClass"
</script>
```

# classList:

### Example:

```
<div id="myDiv" class="class1 class2">Hello</div>

<script>
    const myDiv = document.getElementById('myDiv');

    myDiv.classList.add("class3"); // Adds "class3"
    console.log(myDiv.className); // Outputs: "class1 class2 class3"

    myDiv.classList.remove("class1"); // Removes "class1"
    console.log(myDiv.className); // Outputs: "class2 class3"

    myDiv.classList.toggle("class3"); // Toggles "class3" (removes it)
    console.log(myDiv.className); // Outputs: "class2"
</script>
```

### Function of toggle is -

### if the class is present, toggle() removes it.

### If the class is absent, toggle() adds it.