DOM Manipulation Methods and Properties

1. insertBefore() Method

The insertBefore() method is used to insert a node as a child of a parent node before a specified child node.

Syntax:

```
parentNode.insertBefore(newNode, referenceNode);
```

Parameters:

- parentNode (required): The parent element where the new node will be inserted.
- newNode (required): The node to insert.
- referenceNode (optional): The node before which the new node will be inserted. If null, newNode is appended to the end of parentNode.

Key Points:

- The newNode must be a Node object (e.g., an element, text node, or fragment).
- If the referenceNode is not provided or is null, the newNode is inserted at the end of parentNode.
- insertBefore doesn't create a new node; it moves the existing node or inserts a created one.

Example:

HTML Code:

```
<ul id="list">
  <li>Item 1</li>
  <li class="current">Item 2</li>
  <li>Item 3</li>
</ul>
```

JS Code:

```
let list = document.getElementById('list');
let newItem = document.createElement('li');
newItem.textContent = 'New Item';
list.insertBefore(newItem, list.children[1]); // Inserts before
"Item 2"
```

Modified HTML Output:

```
<ul id="list">
  <li>Item 1</li>
  <li>New Item</li> <!-- Newly inserted item -->
  <li class="current">Item 2</li>
  <li>Item 3</li>
</ul>
```

2. nextElementSibling Property

The nextElementSibling property is used to access the next sibling element (not including text nodes, comments, etc.) of a specific element in the DOM.

Syntax:

```
element.nextElementSibling;
```

Key Points:

- Returns the next sibling that is an element node.
- If the current element has no next sibling element, it returns null.
- Unlike nextSibling, this property skips non-element nodes such as text and comment nodes.

Example:

HTML:

```
<ul id="list">
  <li>Item 1</li>
  <li>New Item</li> <!-- Newly inserted item -->
  <li class="current">Item 2</li>
  <li>Item 3</li>
</ul>
```

JS Code:

```
let current = document.querySelector('.current');
console.log('Next sibling element:',
current.nextElementSibling.textContent);
```

Output:

```
Next sibling element: Item 3
```

3. previousElementSibling Property

The previousElementSibling property is used to access the previous sibling element (ignoring non-element nodes) of a specific element in the DOM.

Syntax:

```
element.previousElementSibling;
```

Key Points:

- Returns the previous sibling that is an element node.
- If the current element has no previous sibling element, it returns null.
- Similar to previousSibling, but skips non-element nodes.

Example:

HTML:

```
<ul id="list">
  <li>Item 1</li>
  <li>New Item</li> <!-- Newly inserted item -->
  <li class="current">Item 2</li>
  <li>Item 3</li>
</ul>
```

JS Code:

```
console.log('Previous sibling element:',
current.previousElementSibling.textContent);
```

Output:

```
Previous sibling element: New Item
```

Practical Application Example

```
<ul id="list">
  <li>Item 1</li>
  <li class="current">Item 2</li>
  <li>Item 3</li>
</ul>
<script>
  let list = document.getElementById('list');

  // Using insertBefore
  let newItem = document.createElement('li');
  newItem.textContent = 'New Item';
  list.insertBefore(newItem, list.children[1]); // Inserts before
"Item 2"

  // Using nextElementSibling
  let current = document.querySelector('.current');
  console.log('Next sibling element:',
current.nextElementSibling.textContent); // Outputs: "Item 3"

  // Using previousElementSibling
  console.log('Previous sibling element:',
current.previousElementSibling.textContent); // Outputs: "Item 1"
</script>
```

This example demonstrates how to use all three methods and properties together in a real-world scenario.

Conclusion

The insertBefore method, along with the nextElementSibling and previousElementSibling properties, provides powerful tools for dynamically navigating and manipulating the DOM. While insertBefore allows precise control over where new nodes are added, the sibling properties enable efficient traversal between elements while ignoring non-element nodes. Together, these

methods simplify working with the DOM structure, making them essential for modern web development.

Resources

- https://developer.mozilla.org/en-US/docs/Web/API/Node/textContent
- https://developer.mozilla.org/en-US/docs/Web/API/Node/insertBefore
- https://developer.mozilla.org/en-US/docs/Web/API/Element/previousElementSibling
- https://developer.mozilla.org/en-US/docs/Web/API/Element/nextElementSibling
- https://developer.mozilla.org/en-US/docs/Web/API/Element