Adding New Child Elements to the Document

In this lesson, we will add new child elements to our document tree.
Let's begin!

WE'LL COVER THE FOLLOWING

After you have grabbed a document element, you can add child elements to it with the `appendChild()` and `beforeInsert()` methods.

The first method appends a new child to the end of the list of exiting children

The second method allows specifying the position of insertion. Listing 6-8 demonstrates using them.

Listing 6-8: Using `appendChild()`, and `beforeInsert()` methods

```
<!DOCTYPE html>
<html>
<head>
  <title>Add new elements</title>
</head>
<body>
  <p>These are items:</p>
  <ol id="list" start="1">
    <li id="item1">Item #1</li>
    <li id="item2">Item #2</li>
    <li id="item3">Item #3</li>
  </ol>
  <button onclick="addToTop()">
    Add new item to top
  </button>
  <br />
  <button onclick="addAfterFirst()">
    Add new item after the first child
  </button>
  <br />
  <button onclick="addToBottom()">
    Add new item to bottom
  </button>
```

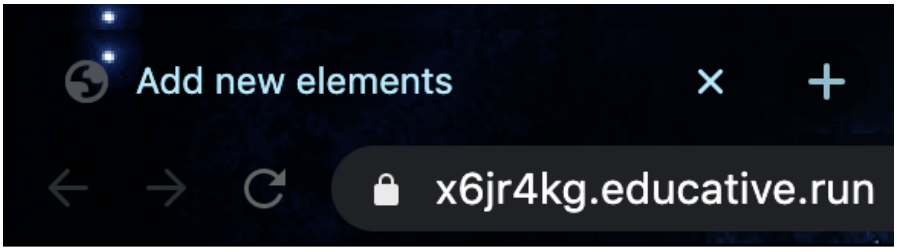
```
  </button>
  <script>
    function addToTop() {
      var list = document.getElementById('list');
      var firstItem = list.firstElementChild;
      var newNode = createNewNode("addToTop");
      list.insertBefore(newNode, firstItem);
    }

    function addAfterFirst() {
      var list = document.getElementById('list');
      var firstItem = list.firstElementChild;
      var secondItem = firstItem.nextElementSibling;
      var newNode = createNewNode("addAfterFirst");
      list.insertBefore(newNode, secondItem);
    }

    function addToBottom() {
      var list = document.getElementById('list');
      var newNode = createNewNode("addToBottom");
      list.appendChild(newNode);
    }

    function createNewNode(message) {
      var item = document.getElementById('list');
      var count = item.childElementCount;
      var newNode = document.createElement('li');
      newNode.textContent =
        "item #" + (count + 1) + " ("
        + message + ")";
      return newNode;
    }
  </script>
</body>
</html>
```

The original page rendered by this markup is shown in the image below:



These are items:

1. Item #1
2. Item #2
3. Item #3

Add new item to top
Add new item after the first child
Add new item to bottom

The original content of the page defined in Listing 6-8

The three buttons activate the `addToTop()`, `addAfterFirst()`, and `addToBottom()` methods. Each method gets the `<ol>` tag, invokes the `createNewNode()` method, and then inserts the new child node.

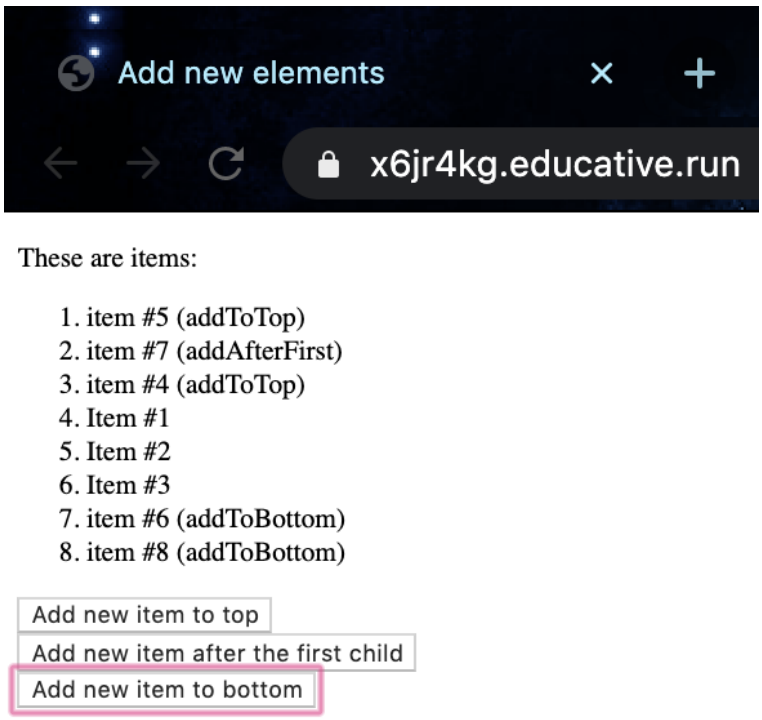
In order to add a new element to the document tree, you first need to create one. The `createNewNode()` uses the `createElement()` method of document.

This is the way to create a new element, and this new object can be assigned only to the document tree it has been created in. You have to pass the type of the element as the argument of `createElement()`, and it retrieves the object that represents the new node.

The implementation in `createNewNode()` accepts a message that is added to the textual content of the new `<li>` element, together with an item number based on the count of child elements.

The `addToBottom()` method uses the `appendChild()` method, while `addAfterFirst()` and `addToTop()` leverage the `beforeInsert()` method. Observe that all of them use the `<ol>` node (through the list object) to add the new child element to, however, each of them uses it a different way. When calling `beforeInsert()`, the first argument is the new node, the second one identifies the reference element to add the new child before.

The image below shows how the list is expanded after adding five new list items. Using the item numbers and messages written between parentheses, you can guess the order of operations invoked.



The page after adding five new list items

In the *next lesson*, we add adjacent elements in the document tree.