Lesson Weekend

Introduction to Programming (/introduction-to-programming) / Arrays and Looping (/introduction-to-programming/arrays-and-looping) / Adding and Removing HTML Elements

Text Cheat sheet

Workflow Tip

- Test out code to add new elements in the DevTools console.
- Use the Elements tab of the DevTools to inspect newly created elements to verify their placement in the DOM.
- Reference documentation as needed.

Code Examples

Creating New Elements

To create new HTML elements, we'll use the `document.createElement()` (https://developer.mozilla.org/en-US/docs/Web/API/Document/createElement) method.

- Argument: the tag name as a string.
- Return value: empty tag, an HTML element object corresponding to the tag created.
- Visit the MDN documentation on the HTML DOM API (https://developer.mozilla.org/en-

US/docs/Web/API/HTML_DOM_API#html_element_interfaces_2) to see a list of all HTML element objects.

```
> const pElement = document.createElement("p");
> pElement;
<p></p>
> Object.prototype.toString.call(pElement);
"[object HTMLParagraphElement]"
```

Adding Attributes

To add, remove, or get attributes to any HTML element, we need to use these methods:

- `Element.setAttribute()` (https://developer.mozilla.org/en-US/docs/Web/API/Element/setAttribute)
- `Element.removeAttribute()` (https://developer.mozilla.org/en-US/docs/Web/API/Element/removeAttribute)
- `Element.getAttribute()` (https://developer.mozilla.org/en-US/docs/Web/API/Element/getAttribute)

Adding Text

We'll use the `Element.append()` (https://developer.mozilla.org/en-US/docs/Web/API/Element/append) method to add text to an element. We call this method on the element that we want to add text to, and the argument we pass in will be string with the text we want to add. Let's look at the code!

```
> const pElement = document.createElement("p");
> pElement;
<p></p>
> pElement.append("text");
> pElement;
<p>text</p>
> p.innerText;
"text"
```

Adding Element to DOM with `Element.append()`, `Element.prepend()`, `Element.after()`, `Element.before()`

The (`Element.append()` (https://developer.mozilla.org/en-US/docs/Web/API/Element/append)), (`Element.prepend()` (https://developer.mozilla.org/en-US/docs/Web/API/Element/prepend)), (`Element.after()` (https://developer.mozilla.org/en-US/docs/Web/API/Element/after)), and (`Element.before()` (https://developer.mozilla.org/en-US/docs/Web/API/Element/before)) methods all add elements to the DOM, but at different locations:

- the `Element.prepend()` method will add a new element inside of the element we're calling the method on, at the top.
- the `Element.append()` method will add a new element inside of the element we're calling the method on, at the bottom.
- the `Element.after()` method will add a new element after the element we're calling the method on.
- the `Element.before()` method will add a new element before the element we're calling the method on.

```
> const pElement = document.createElement("p");
> pElement.append("text");
> const firstDiv = document.querySelector("div");
> firstDiv.append(pElement);
> const liElement = document.createElement("li");
> liElement.append("text");
> const firstUl = document.querySelector("ul");
> firstUl.prepend(liElement);
```

```
> const p1 = document.createElement("p");
> p1.append("text");
> const p2 = document.createElement("p");
> p2.append("text");
> const firstH2 = document.querySelector("h2");
> firstH2.after(p1);
> firstH2.before(p2);
```

Special Note

Note that you can only add the same element once, no matter which method you are using (`Element.append()` , `Element.prepend()` , `Element.after()` , `Element.before()`). You can't do it twice. For example, the following code would only leave the paragraph after the heading:

```
> const pElement = document.createElement("p");
> pElement.append("text");
> const firstH2 = document.querySelector("h2");
> firstH2.before(pElement);
> firstH2.after(pElement);
```

And we'll see the same behavior with `Element.append()` and `Element.prepend()` .

If we want a paragraph added before and after the heading we're targeting, we'll have to create two paragraph elements:

```
> const p1 = document.createElement("p");
> const p2 = document.createElement("p");
> p1.append("text");
> p1.append("other text");
> const firstH2 = document.querySelector("h2");
> firstH2.before(p1);
> firstH2.after(p2);
```

Removing an Element

If we want to remove an element from the DOM, we simply have to get it with a `document` method and call the `Element.remove()` (https://developer.mozilla.org/en-US/docs/Web/API/Element/remove) method on it.

```
> const firstDivOnPage = document.querySelector("div");
> firstDivOnPage.remove();
```

Adding Multiple Elements to the DOM at Once

We can add multiple elements at once with with the `Element.append()`, `Element.prepend()`, `Element.after()`, and `Element.before()` methods by passing in a series of arguments, each one representing an HTML element.

Here's an example with `Element.append()`:

```
> const ul = document.createElement("ul");
> ul.setAttribute("id", "iceCream");
> document.querySelector("div").append(ul);
> const liOne = document.createElement("li");
> const liTwo = document.createElement("li");
> const liThree = document.createElement("li");
> liOne.append("Chocolate");
> liTwo.append("Vanilla");
> liThree.append("Strawberry");
> document.getElementById("iceCream").append(liOne, liTwo,
liThree);
```

Previous (/introduction-to-programming/arrays-and-looping/document-query-methods-that-return-collections)
Next (/introduction-to-programming/arrays-and-looping/debugging-in-javascript-using-a-linter)

Lesson 10 of 50
Last updated February 28, 2023

disable dark mode

Epicodus (http://www.epicodus.com)