Set 20: Document Object Model Part 2

Skill 20.01: Write code to add an element to an HTML page
Skill 20.02: Write code to remove or hide an HTML element
Skill 20.03: Write code to modify an attribute

Skill 20.01: Write code to add an element to an HTML page

Skill 20.01 Concepts

In our previous lesson about the DOM we learned how to select and modify existing elements on our webpage. But, just as the DOM allows scripts to modify existing elements, it also allows for the creation of new ones. The *.createElement(tagName)* method creates a new element based on the specified tag name. However, it does not append it to the document. It creates an empty element with no inner HTML.

To create an element and add it to the web page, you must assign it to an element that already exists on the DOM. We call this process appending. The *.append* method will add the element as the last child node.

The following code (1) creates a new paragraph element and assigns the element to a variable called *myParagraph*, (2) adds text to the new element's *innerHTML*, and (3) appends it to the body of the document:

```
var myParagraph = document.createElement('p');
myParagraph.innerHTML = "The text inside paragraph";
document.body.append(myParagraph);
```

In our previous lessons we have learned how to control elements on the page by accessing their id's. In the example above, the paragraph we created does not have an *id* associated with it. To specify an id we can use the id property as follows,

```
myParagraph.id = "p1";
```

Now that we have associated an *id* to our new element, we can modify it using the same techniques we learned before. In the below example, the background of the new paragraph we created is changed to *seagreen*.

```
document.getElementById("p1").style.backgroundColor = "seagreen";
```

Skill 20.01 Exercise 1

Skill 20.02: Write code to remove or hide an HTML element

Skill 20.02 Concepts

In addition to modifying or creating an element from scratch, the DOM also allows for the removal of an element. The *.remove* method removes a specified element.

```
var e = document.getElementById("someElement");
e.remove();
```

If you want to hide an element because it does not need to be loaded initially, the *.hidden* property allows you to hide it by assigning it as true or false,

```
var e = document.getElementById("someElement");
e.hidden = true;
```

Skill 20.02 Exercise 1

Skill 20.03: Write code to modify an attribute

Skill 20.03 Concepts

Attributes are values that contain additional information about HTML elements. They usually come in name/value pairs, and may be essential depending on the element. Consider the *img* tag below. The image tag below requires the *src* attribute to indicate the location of the image.

```
<img id= “myImage” src = “path/to/my/image.jpg”>
```

The *a* tag below is another example. In this example, the *href* attribute indicates the path to the link and the target attribute indicates where the link should open – in this case, “BLANK” indicates the page should open in a new page.

```
<a href = “path/to/my/page.html” target = “_BLANK”>
```

The code below could be used to modify the *img* tag above,

```
var image = document.getElementById('myImage');

image.getAttribute('src');    // returns path/to/my/image.jpg
image.removeAttribute('src'); // removes the src attribute
image.setAttribute('src', 'diff.jpg'); // changes the src attribute
```

The attribute for any element can be accessed by using the dot notation. Below is another example of how we can access and reassign the path of the image element above.

```
var image = document.getElementById('myImage');
image.src = ‘diff.jpg’;
```

Skill 20.03 Exercises 1