DOM Manipulation

The DOM

Now that we have an idea of the basics of JavaScript, we can combine it with HTML and CSS to change the website.

The `DOM` stands for "Document Object Model", which is a representation of your webpage in the browser. It's a way for JavaScript to interact with and change the the page. For example, if your HTML has a `<div>` tag, the DOM lets JavaScript find it and change it.

Changing Elements

As an example, let's say that we want to change the background color of a div. For this to work, we're going to have to follow a couple steps:

1. Find the element on the DOM
2. Change the background color

That's it! But what would this look like in JavaScript code?

First we need to find the element on the DOM. There are a few different methods that are provided with JavaScript to do this:

```
const div1 = document.getElementById("divId"); // Get an element by its id
const div2 = document.getElementsByClassName("divClass"); // Get all elements with
a certain class
const div3 = document.getElementsByTagName("div"); // Get all elements with a
certain tag
const div4 = document.querySelector(".className"); // Gets the first element with
this class
const div5 = document.querySelector("#id"); // Gets the first element with this id
const div6 = document.querySelector("div"); // Gets the first div on the page
const div7 = document.querySelectorAll("div"); // Gets ALL the divs on the page
and puts them in an array
```

All of these work similarly; they all select a div from your HTML document, but in different ways. You can select a div according to its id, class, or tag name. Which one you want to use depends on how your HTML is set up as well as personal preference. I tend to use getElementById the most since I've found it to be the easiest, typically. Of course, you will need to add IDs to the elements you are planning on selecting.

You'll notice that we're accessing the 'document' before we call the method to select our element. This is because these methods "belong" to the document, so we need to go to the document first in order to access one of its methods.

We then store the element(s) in a variable so we can access it later.

Once we've selected our element with our preferred method, we need to change its background color:

```
const div1 = document.getElementById("divId");
div1.style.backgroundColor = "blue";
```

As you can see, we access the variable that holds the element we want to change. Then, to change any part of its styling, we access the style property, then the backgroundColor, and then set it to a color. You can change whatever properties you want:

```
div1.style.border = "2px solid black";
div1.style.width = "200px";
div1.style.marginLeft = "5px";
```

If you want to change the text inside the div, you can change it like this:

```
div1.innerHTML = "This text will be inside the div";
```

We don't need to access the style property before innerHTML, because we aren't changing the styling, we're only changing the HTML.

Adding/Removing Elements

There may be times where you need to add or remove HTML elements on the screen. Adding elements is slightly more complicated than the previous steps, but still not too tedious:

1. Create your new element
2. Change the styling of this new element
3. Add it to the page.

Let's go through each step to see what the code looks like:

1. To create the element we want to add to the page, we use the 'createElement' method from the document. When calling the function, you put the name of the tag you want to create. In this example, I want to make a `<div>`, so I put "div" as an argument for the method.

```
const newDiv = document.createElement("div");
```

2. This step is very similar to the previous second step. Whatever styling you need to change, you should do it now before we officially add it to the document.

```
newDiv.style.backgroundColor = "red";
newDiv.style.height = "200px";
newDiv.style.width = "200px";
```

3. We created our div, but we haven't actually added it to the document yet. Using document.body.append() we can add the element we just made to the document's body. In the function call, we use our div that we made as an argument.

```
document.body.append(newDiv);
```

That line of code above will add our newDiv to the body of the document, but if you don't want to add it to the body, you can also select a div from the HTML and append it there if you need to:

```
const newDiv = document.createElement("div");
const div = document.getElementById("divId");
div.append(newDiv);
```

Here, I'm making a new div, selecting another one that's already in my HTML, and adding it there. Now, it will be a child of that div.