JavaScript Document Object Model

After learning JavaScript topics in the previous classes, we are ready to combine the three languages (HTML, CSS and JavaScript) to start creating some awesome webpages/applications.

In order to combine the three languages, we have to first look at the **Document Object Model**, also known as the **DOM**.

The Document Object Model (DOM) is a programming interface for HTML and XML documents. It represents the page so that programs can change the document structure, style, and content. The DOM represents the document as nodes and objects. That way, programming languages can connect to the page.

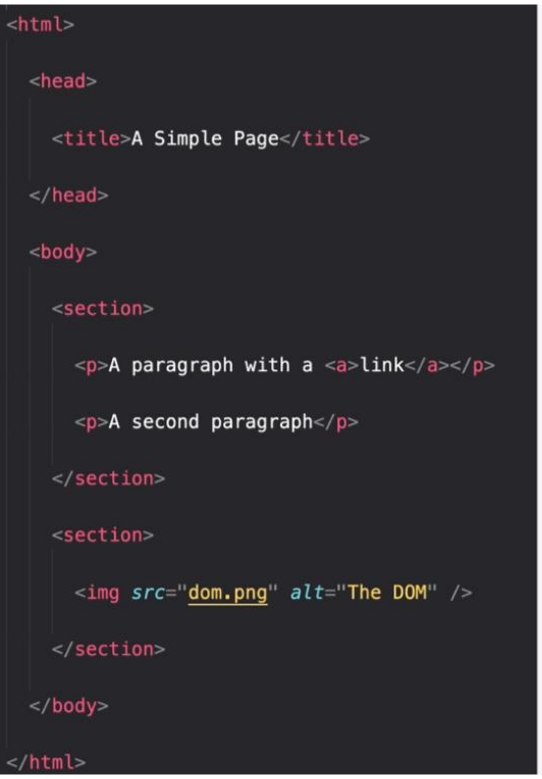
A Web page is a document. This document can be either displayed in the browser window or as the HTML source. But it is the same document in both cases. The Document Object Model (DOM) represents that same document so it can be manipulated. The DOM is an object-oriented representation of the web page, which can be modified with a scripting language such as JavaScript.

$$\begin{tikzcd}\text{JavaScript} \arrow[r, leftrightarrow] & \textbf{DOM} \arrow[r, leftrightarrow] & \text{HTML CSS}\end{tikzcd}$$

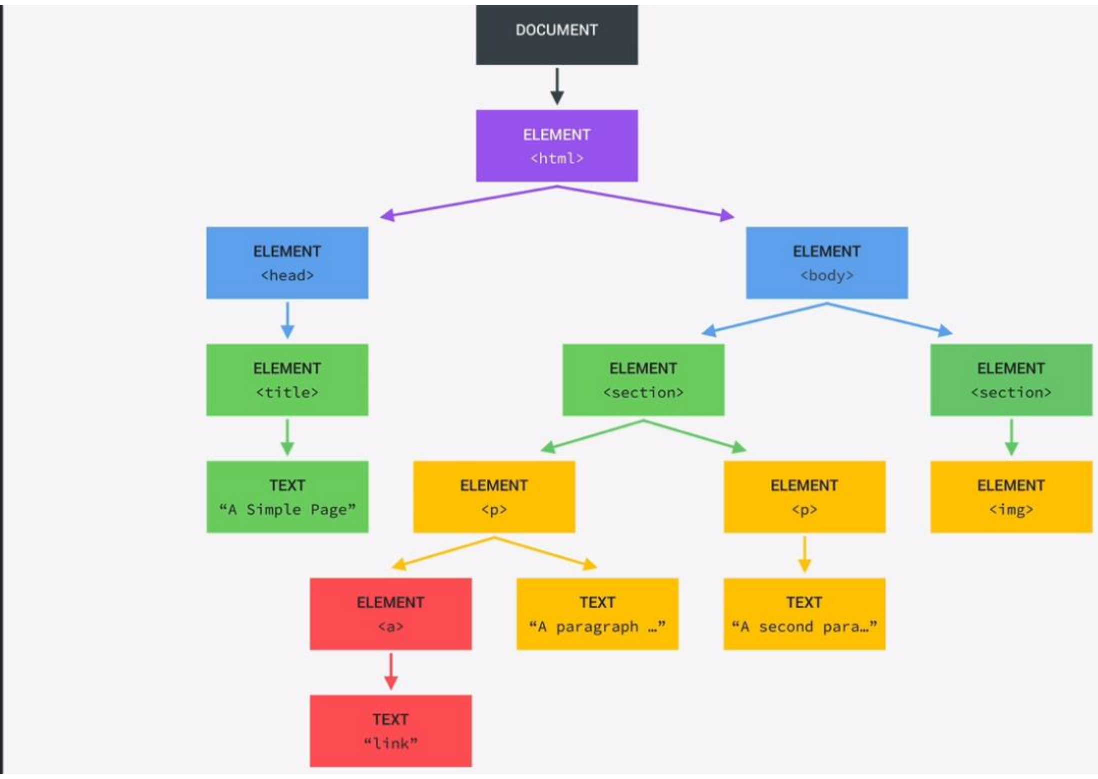
The DOM above contains two types of nodes: element nodes and text nodes.

Elements nodes are represented using angle brackets.
There are a total of 10 element nodes in the HTML file below:
`<html>`, **`<head>`**, **`<title>`**, **`<body>`**, **`<section>`**, **`<img>`**, **`<a>`** and two **`<p>`** nodes.
These element nodes are arranged to show the parent-child and sibling relationships between them.

Text nodes: If we consider the **`<title>`** node, you can see that it has a text node as its child. This text node refers to the text "A Simple Page" enclosed within the **`<title>`**…**`</title>`** tags.



```html
<html>
  <head>
    <title>A Simple Page</title>
  </head>
  <body>
    <section>
      <p>A paragraph with a <a>link</a></p>
      <p>A second paragraph</p>
    </section>
    <section>
      <img src="dom.png" alt="The DOM" />
    </section>
  </body>
</html>
```



Selecting Nodes in DOM

Nodes in the DOM can be accessed and modified in Javascript. In order to access nodes in DOM, we need to use the **document** object. It comes with a few useful methods and properties for accessing and modifying DOM nodes.

getElementById() method :

This method allows us to select an element in JavaScript using its **id** attribute.

For example, to select the element (assume you have below code in your HTML file)

```
<div id ="block1"> The first block tag </div>
```

You can write below code to select the <div> element:

```
document.getElementById("block1");
```

getElementsByClassName() method.
This method selects all the elements that have a certain class name.

getElementsByTagName() method.

This method allows us to select elements using tag names.

querySelector()/querySelectorAll()

Besides the three methods mentioned above, we also have the **querySelector()** and **querySelectorAll()** methods. These two methods allow us to select element(s) in JavaScript using the same notation that we use in CSS.

Properties with the DOM elements

style property

We can use JavaScript to change the CSS style of an element. To do that, we use the **style** property.

This property uses a similar naming convention to CSS. The main difference is, instead of using hyphens when the property name is made up of two or more words, we use camel casing.

```
document.getElementById("block1").style.color = "Red";
```

```
document.getElementById("block1").style.backgroundColor = "Yellow";
```

className property

We can also use **className** property to change the class name of an element:

```
document.getElementById("block1").className = "blockAll";
```

You can verify that the class name of the second <div> has indeed been changed by adding:
```
console.log(document.getElementById("second").className);
```

Interacting with Text Content of Elements

To get the `textContent` of the <div >, we use the corresponding `textContent` property. Add the following line to *script.js*:

```
console.log(document.getElementById("block1").textContent);
```

```
document.getElementById("block1").textContent = "Hello Again"
```

```
console.log(document.getElementById("block1").textContent);
```

Selecting Parent, Child and Sibling nodes Properties

```
<ul>
  <li>A</li>
  <li>B</li>
  <li>C</li>
</ul>
```
childNodes.length

```
console.log(document.querySelector("ul").childNodes.length);
```

firstChild and lastChild / childNodes[ ] / previousSibling and nextSibling / parentNode Properties