Module 3-8

DOM

Objectives

- Difference between the DOM and HTML
- Select elements from the DOM
- Describe the DOM structure
- innerText on HTML elements
- Create new DOM elements
- Traverse the DOM
- Investigate the living DOM in the browser

Document Object Model

- The Document Object Model (DOM for short) is a tree representation of all the HTML elements on a given web page.
- Most browsers have a “Developer Tools” interface that allows for quick inspection of a DOM element and how it relates to other elements on the page.
- The focus of today’s lecture is how to use JavaScript to interact with the DOM.

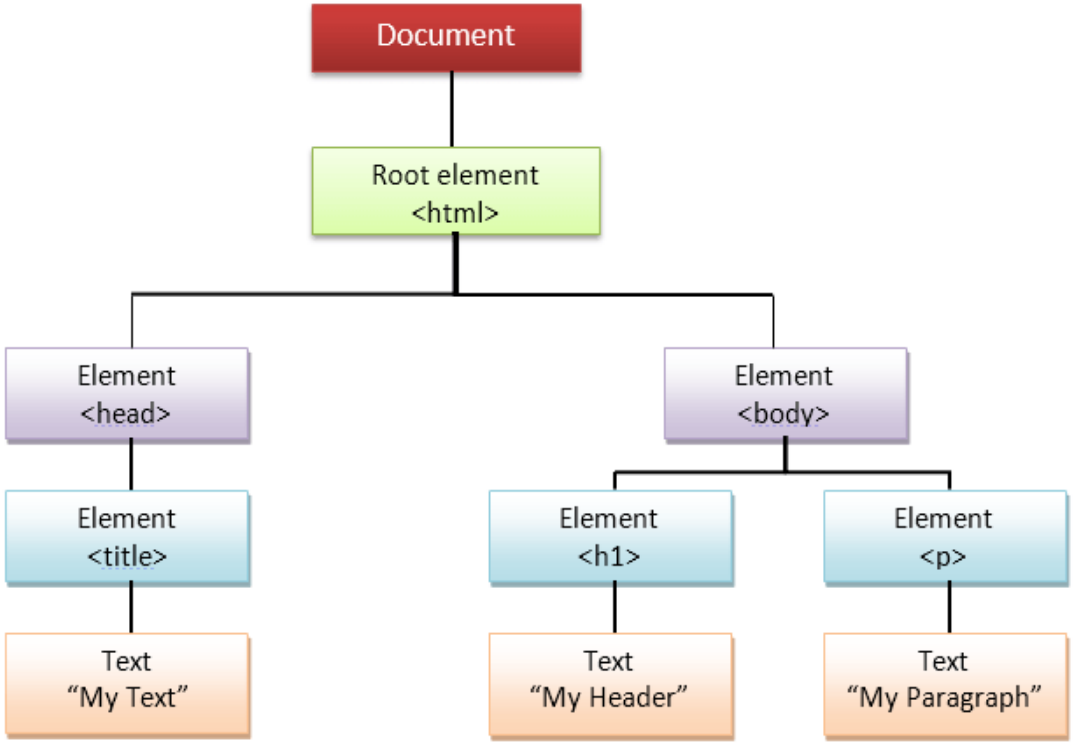
DOM Manipulation

- Writing code to change and select information on the DOM using JavaScript while the page is loaded in the browser. We will be using Vanilla JavaScript for DOM manipulation
- Vanilla JavaScript
  - JavaScript that does not rely on any outside utility libraries to do things that can be done with functions and objects defined in the ECMAScript specification.

DOM vs. HTML

- The DOM is a model of a document with an associated API for manipulating it.
- HTML is markup language that lets you represent a certain kind of DOM in text.
- DOM is tree model to represent HTML.
- DOM doesn’t always match the HTML source code

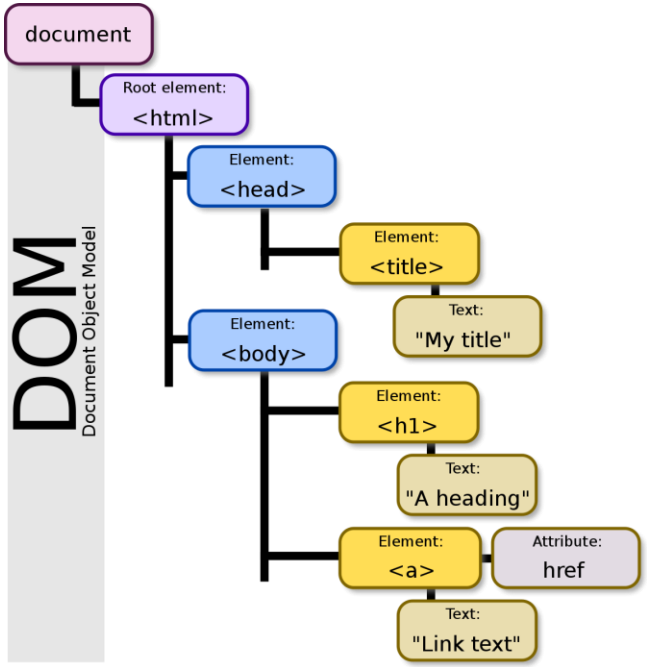
DOM


Document
Root element
<html>
Element
<head>
Element
<body>
Element
<title>
Element
<h1>
Element
<p>
Text
“My Text”
Text
“My Header”
Text
“My Paragraph”

Document Object Model (the DOM)

The Document Object Model (DOM) is an **internal, in-memory representation** of a web page's structure, typically stored in RAM as a **nested tree of objects** that represent the elements of the page.

- It is *not* the page source
- It allows developers to:
  - look for an element with JavaScript
  - find an element's parents, siblings, children
  - add/remove css classes via JavaScript
  - add/remove elements from the page
  - manipulate pretty much anything on the page



DOM Selection Functions

https://book.techelevator.com/content/dom-api-javascript.html#dom-selection-functions

- getElementBylD()
  - This function will get a single HTMLElement from the DOM and return a reference to it.
  - https://developer.mozilla.org/en-US/docs/Web/API/Document/getElementById
- querySelector()
  - Takes a standard CSS selector and returns the first element it finds that matches that selector
  - https://developer.mozilla.org/en-US/docs/Web/API/Document/querySelector
- querySelectorAll()
  - This will return a NodeList of all the elements, which you can use as an array
  - https://developer.mozilla.org/en-US/docs/Web/API/Document/querySelectorAll

Selector Review… Here's Just a Few

https://www.w3schools.com/cssref/css_selectors.asp

Selector	Example	Example description
.class	.intro	Selects all elements with class="intro"
.class1.class2	.name1.name2	Selects all elements with both *name1* and *name2* set within its class attribute
.class1 .class2	.name1 .name2	Selects all elements with *name2* that is a descendant of an element with *name1*
#id	#firstname	Selects the element with id="firstname"
*	*	Selects all elements
element	p	Selects all <p> elements
element.class	p.intro	Selects all <p> elements with class="intro"
element,element	div, p	Selects all elements and all <p> elements
element element	div p	Selects all <p> elements inside elements
element>element	div > p	Selects all <p> elements where the parent is a element
element+element	div + p	Selects all <p> elements that are placed immediately after elements
element1~element2	p ~ ul	Selects every element that are preceded by a <p> element

Changing Elements

- innerText
  - Updates any text information on the page
  - All text (including html tags) is replaced!
  - Insert text treated as literals: no interpreting of HTML
- innerHTML
  - Updates any text information on the page
  - All text (including html tags) is replaced!
  - Interprets HTML for display
  - Do not use with user input! (Why? --see demo)

Manipulating Classes

- classList accesses the classes applied to an element

```
// Get the first line item
let firstListItem = document.querySelector('#todos li');

// Add the class `done`
firstListItem.classList.add('done');

// Remove the class `priority`
firstListItem.classList.remove('priority');
```

Chrome Developer Tools Demo

DOM Elements: ID’s and Classes

Let’s review id and classes for HTML elements. Consider the following HTML code:

```
<p id='intro'>I dedicate this page to my dog Horace</p>

<p class = 'content'>Some Widgets are Doodads</p>
<p class = 'content'>Some Doodads are Thingamagjigs</p>
<p class = 'content'>All Thingamajigs are Whatchamacallits</p>
```

- The first paragraph is marked with an id - ideally we use an id to uniquely identify one element.
- All other paragraphs are marked with a class - ideally we can apply a class to several elements that we feel share some commonality.

DOM Elements: Properties

The id and class names are properties of a DOM Object. We have already dealt with a lot of these properties while learning CSS: height, width, color, etc.

getElementById

We can use getElementById to identify and assign a DOM element to a JavaScript variable. We can then interrogate or change its properties. Consider this example:

HTML

```
<body>
<p id='intro'>I dedicate this page to my dog.</p>
<script src="thisScript.js"></script>
</body>
```

JS

```
let introParagraph = document.getElementById('intro');

console.log(introParagraph.innerText);
introParagraph.innerText = 'I dedicate this page to
Horatio The Cat';

console.log(introParagraph.innerText);
```

- Note that we start off by targeting the intro paragraph, since we know it has an id of intro we can use the getElementById method.
- We assigned this DOM object to a variable called introParagrah.
- We changed the innerText property to contain a different sentence.

getElementById

- The end result of this example is that the HTML page will have “I dedicate this page to Horatio The Cat”, thus changing the original text.

- There is a similar property called innerHTML, that should be avoided as it allows for injection of unwanted JavaScript content beyond the text.
  - innerHTML that takes input from a user sets your page up for XSS
  - Rule of thumb - if you want to change text, use innerText like we have done here.

querySelectorAll

- getElementById is useful for identifying one DOM element but sometimes we need to identify several elements in one blow.

- In order to do this, we can leverage querySelectorAll which will return all matching elements and place them in an array.

querySelectorAll

Let's look at this example again:

HTML

```html
<p id='intro'>I dedicate this page to my dog.</p>
<p class = 'content'>Some Widgets are Doodads</p>
<p class = 'content'>Some Doodads are Thingamagjigs</p>
<p class = 'content'>All Thingamajigs are Whatchamacallits</p>
```

JS

```js
let paragraphs = document.querySelectorAll('.content');
console.log(paragraphs.length);

for (i = 0; i < paragraphs.length; i++) {
   let paragraph = paragraphs[i];
   paragraph.style.color = 'blue';
}
```

browser:

I dedicate this page to my dog.

Some Widgets are Doodads

Some Doodads are Thingamagjigs

All Thingamajigs are Whatchamacallits

querySelectorAll

Here’s another example note what we’ve passed to the querySelectorAll method:

HTML

```html
<p id='intro'>I dedicate this page to my dog.</p>
<p class = 'content'>Some Widgets are Doodads</p>
<p class = 'content'>Some Doodads are Thingamagjigs</p>
<p class = 'content'>All Thingamajigs are Whatchamacallits</p>
```

JS

```js
let paragraphs = document.querySelectorAll('p');
console.log(paragraphs.length);

for (i = 0; i < paragraphs.length; i++) {
   let paragraph = paragraphs[i];
   paragraph.style.color = 'blue';
}
```

browser:

I dedicate this page to my dog.

Some Widgets are Doodads

Some Doodads are Thingamagjigs

All Thingamajigs are Whatchamacallits

querySelector

Finally, we have querySelector() which returns the first element found that matches a given criteria.

```
<p id='intro'>I dedicate this page to my dog Horace</p>
<p class = 'content'>Some Widgets are Doodads</p>
<p class = 'content'>Some Doodads are Thingamagjigs</p>
<p class = 'content'>All Thingamajigs are Whatchamacallits</p>
```

```
let paragraph = document.querySelector('p');
console.log(paragraphs.innerText);
```

“I dedicate this page to my dog Horace”

Let’s Try This Out!

value and checked properties

value gets the value from a text field. checked returns status of radio or checkbox elements:

HTML

```
Name: <input type="text" id="myText" value="Mickey"><br><br>
<form>
  What color do you prefer?<br>
  <input type="radio" name="colors" id="red">
  <label for="red">Red</label><br>
  <input type="radio" name="colors" id="blue">
  <label for="blue">Blue</label>
</form>
```

JS

```
document.getElementById("myText").value = "Johnny Bravo";
document.getElementById("red").checked = true;
```

Using value, set the text field to “Johnny Bravo”

Using checked, set the red box to true (or checked).

Creating DOM Elements

We can create brand new DOM elements from scratch. Consider the following code:

HTML

```
<ul id='theList'>
   <li>Some Widgets are Doodads</li>
   <li>Some Doodads are Thingamagjigs</li>
   <li>All Thingamajigs are Whatchamacallits</li>
</ul>
<script src="thisScript.js"></script>
```

JS

```
let extraListItem = document.createElement('li');
extraListItem.innerText = 'All Foos are Bars';

let parentList = document.getElementById('theList');
parentList.appendChild(extraListItem);
```

A brand new element (a list item) is being created.

We identify the parent.

Append the brand new element to the parent.

Assigning a class to an element

We can create brand new DOM elements from scratch. Consider the following code:

HTML

```
<ul id='theList'>
   <li>Some Widgets are Doodads</li>
   <li>Some Doodads are Thingamagjigs</li>
   <li>All Thingamajigs are Whatchamacallits</li>
</ul>
<script src="thisScript.js"></script>
```

JS

```
let extraListItem = document.createElement('li');
extraListItem.innerText = 'All Foos are Bars';
extraListItem.setAttribute('class', 'importantStuff');

let parentList = document.getElementById('theList');
parentList.appendChild(extraListItem);
```

CSS

```
.importantStuff {
   color:red;
}
```

browser:

- Some Widgets are Doodads
- Some Doodads are Thingamagjigs
- All Thingamajigs are Whatchamacallits
- All Foos are Bars

Inserting elements into the DOM

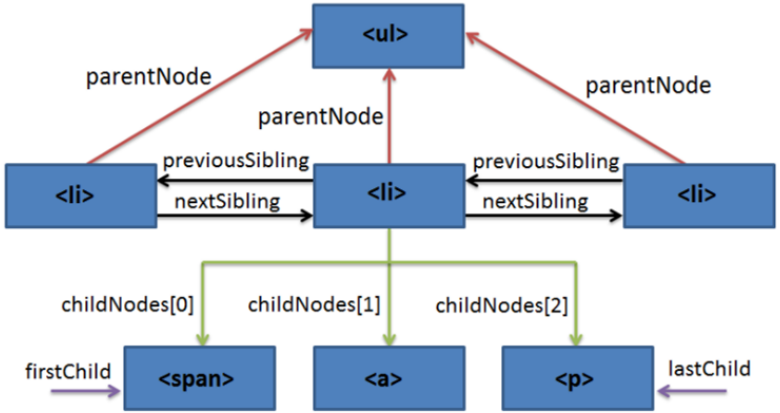
- insertAdjacentElement
  - beforeBegin
  - afterBegin
  - beforeEnd
  - afterEnd

```
<!-- beforebegin -->
<p>
  <!-- afterbegin -->
  foo
  <!-- beforeend -->
</p>
<!-- afterend -->
```

Traversing the DOM

```
<ul>
<li>node</li>
<li><span>node</span><a href="#">node</a><p>node</p></li>
<li>node</li>
</ul>
```

- previousSibling
- nextSibling
- childNodes
- firstChild
- lastChild
- parentNode



Selecting children with children and childNodes

- children
  - Returns an HTML collection, which you can turn into array
  - Returns elements that are children
    - Only contains HTML elements
    - Not text that might be in element
- childNodes
  - Returns a NodeList object that contains all nodes inside element (can also turn into array)
  - Returns nodes that are children of element
    - Includes text and comments that are in DOM

parentNode and adjacent elements

- parentNode
  - Returns parent of element
- Adjacent elements
  - nextElementSibling
  - previousElementSibling

Let’s Code!!

(but first)…

Product Reviews for Cigar Parties for Dummies

Host and plan the perfect cigar party for all of your squirrelly friends.

Malcolm Gladwell

★★★ **What a book!**

It certainly is a book. I mean, I can see that. Pages kept together with glue (I hope that's glue) and there's writing on it, in some language.

Tim Ferriss

★★★★ **Had a cigar party started in less than 4 hours.**

It should have been called the four hour cigar party. That's amazing. I have a new idea for muse because of this.

Ramit Sethi

★ **What every new entrepreneurs needs. A door stop.**

When I sell my courses, I'm always telling people that if a book costs less than $20, they should just buy it. If they only learn one thing from it, it was worth it. Wish I learned something from this book.

Gary Vaynerchuk

★★★ **And I thought I could write**

There are a lot of good, solid tips in this book. I don't want to ruin it, but prelighting all the cigars is worth the price of admission alone.