

Web Scripting 1

Day 04

Agenda

- Assignment 3 walkthrough
- Review
- Selectors
- DOM Traversal
- DOM Manipulation
- jQuery Introduction

Assignment 03

Walkthrough

Progress Check-In

- At this time you should have a rough idea of the following concepts
 - A basic understanding of:
 - Conditional statements
 - Loops
 - Arrays

JavaScript DOM Selector Methods

getElementById()

- Selects an HTML element that has an “id” attribute that is equal to the string passed into the “getElementById()” method
- Each ID attribute on an HTML page must be unique
 - You can’t have multiple HTML elements with the same ID value
- Since ID attribute values must be unique, JavaScript will select the first DOM element that equals the ID value passed into the “getElementById()” method
- The “getElementById()” is a very performant JavaScript selector method

getElementById()

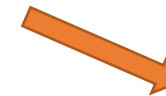
HTML

```
<p id="foo">Select me...</p>
```



JS

```
const para1 = document.getElementById('foo');  
para1.style.color = 'red';
```



Browser output

Select me...

Live Selectors

- Live Selectors allow you to select DOM elements by an HTML tag or by a class name
- Live Selectors will automatically update if DOM elements that match the selection are added or removed
- JS live Selector methods
 - `getElementByTagName()`
 - `getElementByClassName()`
- These methods can select multiple DOM elements,
- These methods return an HTML collection, which is similar to an array
 - An HTML collection does not include the `forEach()` method

getElementsByTagName()

- Selects all elements in the DOM that match an HTML tag passed into the "getElementsByTagName()" method
- Returns an HTML collection which is similar to an array
- You can access individual items in an HTML collection in a similar way to how you access items in an array
 - Example:
 - Select the first item in an HTML collection stored in a variable called "paras"
 - `paras[0]` // returns the first DOM element in the collection

getElementsByTagName()

HTML

```
<p>Select me...</p>  
<p>Select me as well...</p>
```



JS

```
const paras = document.getElementsByTagName('p');  
  
paras[0].style.color = 'red';  
paras[1].style.color = 'green';
```



Browser output

Select me...

Select me as well...


getElementsByTagName()

- Selects all the DOM elements that have a “class” attribute that is equal to the string passed into the “getElementsByTagName()” method
- Returns an HTML collection which is similar to an array
- You can access individual items in an HTML collection in a similar way to how you access items in an array
 - Example:
 - Select the first item in an HTML collection stored in a variable called “elFoo”
 - elFoo[0] // returns the first DOM element in the collection

getElementsByClassName()


HTML

```
<h2 class="foo">Select me...</h2>  
<p class="foo">Select me as well...</p>
```



JS

```
const elFoo = document.getElementsByClassName('foo');  
  
elFoo[0].style.color = 'red';  
elFoo[1].style.color = 'green';
```



Browser output

Select me...

Select me as well...

Query Selectors

- Query selectors allow the selection of DOM elements by passing in a CSS selector to a query selector
- Query selectors will not automatically update if DOM elements that match the selection are added or removed
- JS query Selector methods
 - `querySelector()`
 - Selects the first DOM element that matches a CSS selector passed into the method
 - `querySelectorAll()`
 - Selects all DOM elements that match a CSS selector passed into the method
 - Returns a `NodeList` which is similar to an array
 - A `NodeList` has access to the `forEach()` method, which enables easier looping through all the DOM elements in the selection

querySelector()

- Selects the first DOM element that matches a CSS selector passed into the “querySelector()” method
- Returns a single HTML

querySelector()

HTML

```
<div class="container">  
  <p>Select me...</p>  
</div>
```



JS

```
const para1 = document.querySelector('.container p');  
para1.style.color = 'red';
```



Browser output

Select me...

querySelectorAll()

- Selects all the DOM elements that matches a CSS selector passed into the “querySelectorAll()” method
- Returns a NodeList which is similar to an array
- You can access individual items in a NodeList in a similar way to how you access items in an array
 - Example:
 - Select the first item in an HTML collection stored in a variable called “elFoo”
 - elFoo[0] // returns the first DOM element in the collection
- NodeLists have a forEach() method for easier looping over all the DOM elements in a NodeList

querySelectorAll()

HTML

```
<div class="container">  
  <p>Select me...</p>  
  <p>Select me as well...</p>  
</div>
```

Browser output

Select me...
Select me as well...

JS

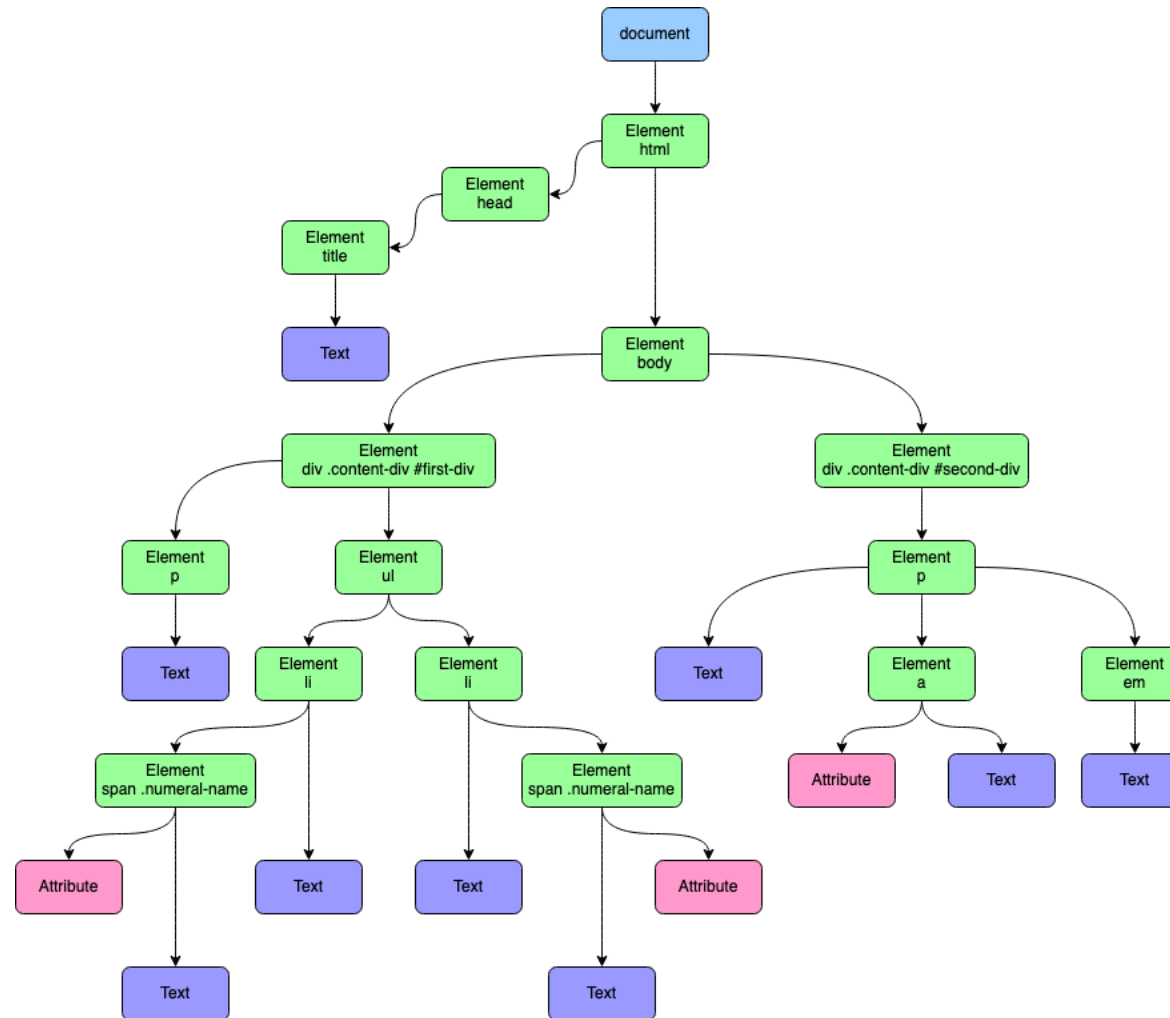
```
const paras = document.querySelectorAll('.container p');  
paras.forEach(el => el.style.color = 'red');
```

Narrowing the Selection Search

- Usually, all the DOM selection methods are called from the document object
 - Example:
 - `document.querySelectorAll('.container p')`
- You can also call any DOM selection method from any variable that contains a DOM element as its value
 - Example:
 - `const container = document.querySelector('.container');`
 - `const containerParas = container.querySelector('p');`
- This can have some performance benefits as the browser only has to search part of the DOM for a selection as opposed to searching the entire DOM

JavaScript DOM Traversal

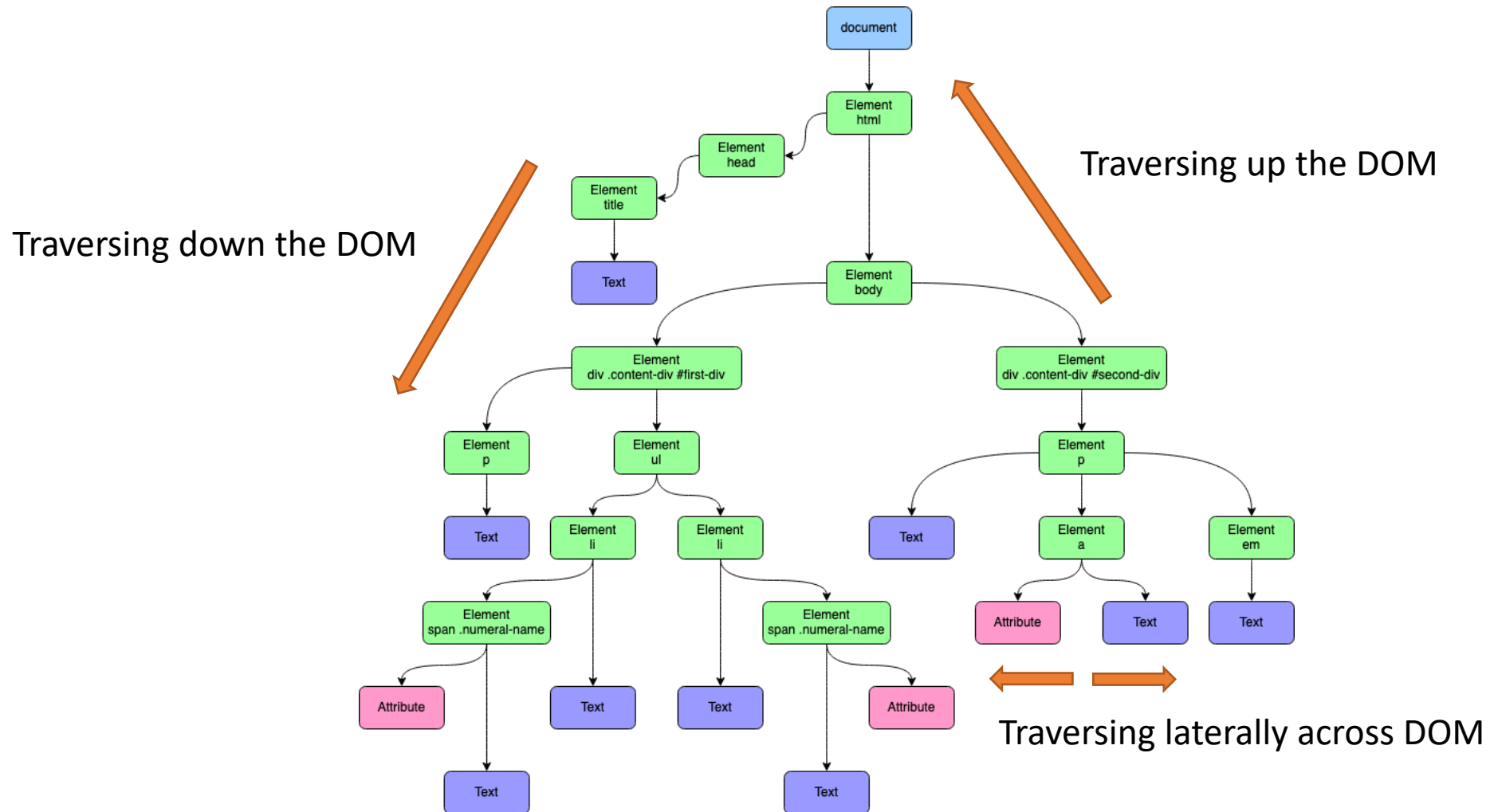
JavaScript DOM Traversal



JavaScript DOM Traversal

- DOM Traversal, means moving around the document from one DOM element to another
- JavaScript for the browser provides several methods for traversing the DOM
- These DOM traversal methods allow you to traverse up the DOM tree, down the DOM tree, and laterally at the same level of the DOM tree

JavaScript DOM Traversal



JavaScript DOM Traversal

- JavaScript DOM Traversal learning resources
 - W3Schools – Javascript DOM Navigation
 - https://www.w3schools.com/js/js_htmlDOM_navigation.asp
 - MDN - Using the DOM
 - https://developer.mozilla.org/en-US/docs/Web/API/Document_object_model/Using_the_W3C_DOM_Level_1_Core
 - JS in Plain English – How to Traverse the DOM
 - <https://javascript.plainenglish.io/how-to-traverse-the-dom-in-javascript-d6555c335b4e>

JavaScript DOM Manipulation

JavaScript DOM Manipulation

- DOM Manipulation, means adding, removing or changing DOM elements
- JavaScript for the browser provides several methods for manipulating the DOM
- These DOM manipulation methods allow you to clone elements, remove elements and modify elements
- Use Google to find JavaScript syntax for manipulating the DOM
 - Try these search phrases
 - “clone DOM element JavaScript”
 - “remove DOM element JavaScript” or “delete DOM element JavaScript”
 - “change text of DOM element JavaScript”
 - “change attribute of DOM element JavaScript”

jQuery

About JQuery

- Created by John Resig and introduced on August 26, 2006
- John originally wanted to call the library “jselect”...but all the domains were taken...so he chose jQuery, even though jQuery has little to do with making queries
- JQuery's goal was to simplify the process of writing cross browser JavaScript code
- Currently used by over 52% of the 10,000 most visited sites on the internet

About JQuery

- JQuery is JavaScript
 - JQuery is simply a large JavaScript object with methods
- JQuery is a JavaScript Library that sits on top of JavaScript
- JQuery will not run if JavaScript is turned off by the user
- A JavaScript Library is a library of pre-written JavaScript that allows for easier use of JavaScript
- Since JQuery is JavaScript it is client side scripting and is used for front-end development
- JQuery does not completely replace JavaScript syntax, it only makes some elements of JavaScript easier to write and use, such as selecting elements or DOM traversal
 - Other areas of JavaScript are still written in plain JavaScript, such as loops and “if” statements

JQuery = JavaScript

Benefits of JQuery

- JQuery makes writing JavaScript easier
- JQuery includes easy ways to select elements on the DOM using common CSS syntax to select elements on a web page
- JQuery helps with cross-browser compatibility issues that are present in native JavaScript
- JQuery helps you to separate scripting behaviour from presentation
 - JQuery encourages you to keep your scripting coding separate from your HTML code by making scripts easier to write that do not require JavaScript to be mixed in with the HTML code
 - This is similar to the way CSS works by separating style from presentation
- JQuery is widely used, consequently there are a lot of resources available on the web to help you with your JQuery code

Downsides of jQuery

- It is extra code that sits on top of JavaScript
 - Some purists may argue that it is more efficient to write pure JavaScript
- Due to jQuery's ease of use it can often be easily overused
 - Be careful to only use jQuery if it is truly needed
 - Extra animations and special effects should be thought of as icing on the cake and not the cake itself
- Always keep the ideals of progressive enhancement in mind
 - Not everyone surfs the web with JavaScript enabled
 - Be mindful of how your web page will appear if JavaScript is disabled

jQuery Resource

- For general information on learning jQuery visit this web site:

<http://learn.jquery.com>

- For information on the jQuery API visit this web site:

<http://api.jquery.com>

Adding the JQuery Library to a Web Page

- The jQuery library requires JavaScript to run
- jQuery is just a single .js JavaScript file that must be included on your web page, just like any other external JavaScript file
- You have a couple of options for including jQuery on your web page
 - You can download the jQuery source code and store it in a file locally or on your web server and then link to that file in your HTML file
 - You can link to a jQuery file that is hosted by a CDN (Content Delivery Network)

jQuery CDN

- The best way to include the jQuery library is through their CDN
- Visit the link below to get the CDN link code to include on your HTML page

<https://code.jquery.com>

Adding the JQuery Library to a Web Page

- Compressed vs Uncompressed
 - Use compressed or minified version wherever possible
 - Use the uncompressed version if you wish to look at the code of the library and give it a read and explore the inner workings of jQuery

jQuery Code

- jQuery uses a single function called jQuery
- Using this function prevents code conflicts between the jQuery library and custom code that developers create
- Writing the function name jQuery can quickly become tiresome
 - The jQuery function name can be shortened to just “\$”
 - Both “jQuery” and “\$” will work for your jQuery scripts

jQuery = \$

jQuery Code

- Each jQuery command is made up of four parts
 1. jQuery object (“jQuery” or “\$”)
 2. Selector
 3. Actions
 4. Parameters

