HTML

- HTML stands for HyperText Markup Language
- Browsers are configured to read standard HTML and render them into what you see as a web page
- HTML is a series of tags wrapped in angle brackets
  - <p>Paragraph</p>
  - <br /> //line break
- Some tags are self closing, like the break tag, while others require an open and closing tag, such as the paragraph tag

HTML Versions

- Like most languages, they are continuously updated over time into new versions
- HTML 5 has been out for a reasonably long time at this point, and thus, most major browsers support most/all of the standard
  - Most notably, the last version of IE did not fully support HTML 5, but Edge does a reasonably good job
- HTML 5 introduces new tag options, as well as modifications to older tags, and in some cases, recommends no longer using certain older tags
- We will focus on using HTML 5 in the way that Chrome/Firefox currently support it

HTML Versions

- If you were doing web development for a company, you would need to determine what browsers they expect the site to work in, and use code that is compatible with those browsers
- Internet Explorer is often the limiting factor in what portion of HTML 5 can be used
  - Even though Edge was supposed to replace IE, there are still plenty of companies/users who continue to use IE
- For this course, we will assume we are building for an internal audience, where we can better control the versions of browsers being used, so we can leverage the full power of the newer HTML 5 tags

Common HTML Tags

- I will go over some of the primary HTML tags you are going to want to use, but we will only be able to barely scratch the surface of what tags are available in HTML
- So, you will want to become familiar with http://www.w3schools.com/ which has probably the best set of examples of HTML/CSS/JavaScript and more
- HTML Tags: http://www.w3schools.com/tags/default.asp

Common HTML Tags

- Lets go ahead and take a look at what a basic HTML page looks like and add a few common tags to see how they display
- Later on, we will look into styling these tags, but for the time being we will focus on what they render as by default

Common HTML Tags

- <!doctype>
- <html>
- <head>
- <title>
- <body>
- <div>
- <span>
- <p>
- <br />
- <table>
  - <thead>
  - <tbody>
  - <tfoot>
  - <tr>
  - <th>
  - <td>
- <ul> / <ol>
  - <li>

Common HTML Tags

- <fieldset>
  - <legend>
- <a>
- <img>
- <code>
- <h1> - <h6>
- <hr>

- <form>
  - <input>
    - <label>
  - <select>
    - <option>
  - <textarea>

HTML Tag Attributes

- HTML tags can also have attributes applied to them within the angular brackets to further define how they should be rendered by the browser
- Each tag has it’s own unique list of possible attributes
- There are some attributes that are shared across all tags, and we will look at those later
- Let’s look at some simple options on the <table> tag to adjust how our table renders
- If you are ever curious about attribute options for a tag, just check out W3Schools

HTML Tags

- Again, there are several tags that I skipped
- Some of these tags we will look at as we get into various sections throughout the course, others you may want to look into on your own
- If you ever have an idea of what you want to accomplish, but aren’t sure what tags might be able to do it, just let me know and I’ll try to help point you in the right direction

Page Source

- Often times, we get ideas about what we want to do on our site from what we see on other sites
- When it comes to the HTML tags being used, it is extremely easy to see how a page is constructed
  - We will look more into styling the tags next time, which we can also get ideas from viewing page source
- Firefox page source will highlight possible errors
  - Most browsers will take a “best guess” at what you meant, but if they guess wrong, it can be challenging to figure out why a small change you made caused such a large change in the page

Review

- HTML Tags
  - Format for sending content to a browser in a way it can interpret
  - Open and Close tags OR self closing tag
    - <a></a> OR <br />
  - Attributes to further describe how to render a tag
    - <table border=“1”>

CSS

- CSS stands for Cascading Style Sheet
- This is where we can define how we want a given element to display differently than a browser would normally render it
- Styles can be applied in three different places
    - On the element you want to style [inline]
    - Within <style> tags on the page (usually in the <head> section) [internal]
    - In a separate file, with a call to use the file in the <head> section [external]

CSS Preferred Location

- Our preference is going to be to always use the separate file
- This allows us to use the same style sheet on multiple pages to produce a consistent look and feel more easily

Inline CSS

- To apply styles on the element, we use a common attribute:
  - <div style=“”>, <span style=“”>, etc
- Then we can put our styling properties inside the quotes
  - <div style=“background-color:blue”>
- Again, there is a very large number of styling properties available, and we will only scratch the surface
- Reference: http://www.w3schools.com/cssref/

Internal CSS

- To add styles on an individual page, we wrap the style section in <style> tags, usually in the <head> section
- Because the styles aren’t on an individual tag, we need a way to specify what tag the style applies to
  - tag name
  - .class
  - #id
- There are also some advanced CSS selectors we will talk about

Basic CSS Selectors

- Tag Name
  - div{}
    - Affects all <div> tags on the page
- Class
  - .myClass
    - Affects all tags on the page with class="myClass" attribute
- ID
  - #unique
    - Affects the single tag on the page with the id="unique" attribute

External CSS

- Finally, we can put the CSS into a separate file and include it into our page using <link> tags in the <head> section
  - <link rel=“stylesheet” href=“styles.css” />
- Inside the styles.css file, we don’t need any html tags, we can just go straight to using our CSS selectors to specify how to style elements on the page

CSS Colors

- CSS colors are defined by a hexadecimal representation
- RGB – Red, Green, Blue
  - 0-255 (or 00-FF in hex)
  - This is the intensity of each color light
  - #FF0000 (255, 0, 0) = Red (red turned all the way on, others off)
  - #555555 (85, 85, 85) = Gray
- Color Picker:
  - https://www.w3schools.com/colors/colors_picker.asp

CSS Colors

- There are also several color keywords that can be used
- https://www.w3schools.com/colors/colors_names.asp

- There are several online resources to help you build a set of complimentary colors
  - http://paletton.com

CSS Sizes

- When determining the size of elements, often you will see:
  - px – Pixels
  - % - Percentage
  - em – Relative to font-size
- The full list is available here: http://www.w3schools.com/cssref/css_units.asp
- We see less ‘px’ used, because ‘%’ and ‘em’ can be better manipulated as a page is resized for responsive designs, which we will talk about in a few weeks

Common CSS Properties

- background-color
- border-width
- border-style
- border-color
- height/width
- margin
- padding
- letter-spacing
- word-spacing
- line-height
- text-align
- vertical-align
- text-transform
- text-decoration

Common CSS Properties

- font-family
- font-size
- font-weight

- float
- clear
- display
- max/min height/width
- position
- top/left/right/bottom

Advanced CSS Selectors

- As a note, I will be using mostly tag names in the following examples, but they could be replaced with any of the basic selectors
- http://www.w3schools.com/cssref/css_selectors.asp

Advanced CSS Selectors

- We can start by just combining the basic selectors
  - div table{}
    - selects tables inside of div tags
    - space = contained within
  - div.class{}
    - selects div tags that have the given class
    - this can be useful when you use the same class on several different types of tags
  - td, th{}
    - all td’s and th’s
    - this makes it easier to style multiple tags the same way

Advanced CSS Selectors

- div table{} can select a table inside of a div, even if the table is also inside of other elements (div doesn’t have the be the direct parent)
- div > table{} requires that div be the direct parent of table
- div + table{} selects tables positioned directly after div tags
- div ~ table{} selects tables that are preceded by a div tag (does not have to be directly preceded)

Advanced CSS Selectors

- Square brackets can be used to specify that we want to look at attributes on a given tag to determine which tags to select
  - [target] – selects tags with the target attribute applied
  - [target=_blank] – target attribute equals _blank
  - a[href^=“https”] – a tag with href that begins with https
  - a[href$=“.pdf”] – a tag with href that ends with .pdf
  - a[href*=“text”] – a tag with href that contains “text”

Advanced CSS Selectors

- td:first-child – selects the first td in every row, where the td is the first-child
- td:nth-child(2) – selects the second td in every row
- td:last-child – selects the last td in every row
- There are several other advanced CSS selectors, and most of these can be mixed and matched to select just the elements you want to on the page
- More often than not, in this course, you will be able to just put unique classes on whatever you want to style, but sometimes these come in handy

Basic Layout

- Now that we have some HTML and CSS tools, let’s put together a very simple page layout using <div> tags, widths/heights/float

JavaScript

- JavaScript (JS) is our client side programming language
- This means that the code is executed by the user's browser and not by the server providing the page
- It is important to understand this concept, because a user could potentially change settings in their browser to not render JS at all, or use tools to modify what it is supposed to do on a single page load
- For this reason, when building larger sites, we often need to consider how the site would render without JS

JavaScript

- Again, for this course, we are assuming internal customers who have the correct browser settings
- When we get into PHP and MySQL later on, we will be working with forms and validating input data
  - We can use JS to do some client side validation, but we should never leave that as the only type of validation
- JS excels at performing tasks such as modifying HTML content, attributes, and CSS

Where to put on the page

- JS can go in the <head> or <body> wrapped in <script> tags, as well as in an external file (.js)
- We might put some JS in an external file that takes care of making our menu’s function, and then put some JS in the page to manage how a unique element works on that page (example: directory table sorting)
- Some JS also goes in tags as part of attributes, such as:
  - <img onclick=“myFunction()” />

Basic Language Syntax

- Basic JS syntax
    - variables
        - data types
    - arrays
    - functions/methods
        - variable scope
    - conditionals
    - loops
    - Date
        - Basic object manipulation

Modify Content

- Modify contents of an HTML tag
  - document.getElementById(“divID”).innerHTML = “”;
- Modify attributes of an HTML tag
  - document.getElementById(“myImg”).src = “”;
- Modify styles of an HTML tag
  - document.getElementById(“divID”).style.fontSize = “”;

JavaScript DOM

- DOM = Document Object Model
- This is the hierarchy of objects on the page and how we access the items using JS
  - Document Node
    - http://www.w3schools.com/jsref/dom_obj_document.asp
  - HTML Element Node
  - HTML Attribute Node

Document

- addEventListener()
- anchors
  - <a> tags with names
- baseURI
  - full page URL
- body
- createElement()
- getElementById()
- getElementsByClassName()
- getElementsByName()
- getElementsByTagName()
- write()

- getElements* returns an array of elements, even if only one item returned

Element

- addEventListener()
- appendChild()
- attributes
- childElementCount
- childNodes
  - includes text/comments
- children
  - excludes text/comments
- className
- cloneNode()
- firstChild
- firstElementChild
- lastChild
- lastElementChild
- nextSibling
- nextSiblingElement
- parentNode
- parentElement

Element

- getAttribute()
  - value
- getAttributeNode()
- hasAttribute()
- hasAttributes()
- getElementsByClassName()
- getElementsByTagName()
- hasChildNodes()
- click()
- focus()
- id
- innerHTML
- style
- textContent

Attribute

- name
- value
- specified

Review

- Is JS executed by the Server or the Browser?
- What tasks does JS excel at?
- When we declare a variable inside a function without the “var” keyword, is it scoped to be local to the function or global to the page?
- What does DOM stand for?

JavaScript Events

- onclick
- onmousedown/up
- onmouseenter/leave
- onmousemove
- onmouseover/out
- onkeydown/up/press
- onload
- onresize
- onscroll
- setInterval(func, milli)
- setTimeout(func, milli)
- Forms
  - onchange
  - onfocus
  - oninput
  - onreset
  - onsubmit

JavaScript

- eval() //execute code written as text (generally unsafe)
- setInterval() //run at set interval
  - window.clearInterval()
- setTimeout() //run once after certain amount of time

JQuery

- In this course, we are going to look primarily at JQuery
- JQuery currently has the largest usage across the internet, estimated at over 70% of websites using it, which is around 97% of the websites that use any JS library
- You are not locked into using only one JS library, but many sites choose only one to simplify development
- Usage Stats:
    - http://w3techs.com/technologies/overview/javascript_library/all

JQuery

- http://jquery.com
- Running a JQuery command requires that we include the library
  - We can download and host the library locally, or we can use a CDN (Content Delivery Network), basically someone else hosting the file for us

JQuery CDN

- One of the benefits of using a CDN is lightening the load on your web server, and possibly improved load times for your visitors, as their browser may have cached the file when they visited another website using the same CDN
  - So, lets use the Google CDN:
    - <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
  - Google has a list of libraries they host: https://developers.google.com/speed/libraries/#jquery

JQuery

- Now that we have the library added, we can begin using the various functions available
  - JQuery functions are called from a root JQuery object:
    - JQuery()
  - Which has an alias for faster typing:
    - $()
- So, our command last week to change the content of a div tag:
  - document.getElementById(“myDiv”).innerHTML = “”;
  - $(“#myDiv”).html(“”);

JQuery

- The JQuery object function expects a selector to specify what you want to modify, then has function calls for how you want to modify the given item
    - $(selector).action();
- I recommend generally using either the $() or jQuery(), but not mixing and matching them in your code for readability
- Many examples you find online use the $() version

JQuery

- How you select elements is one of the most powerful components of JQuery, so having a good grasp of what is possible is important to writing short JQuery commands
- http://api.jquery.com/category/selectors/

- W3Schools has a good tutorial on JQuery, as does JQuery.com, which also has the full API:
  - http://api.jquery.com/

JQuery

- One of the most common ways to open up a JQuery section is with
  - $(document).ready(function(){//your code});
- This ensures the page has finished loading before beginning the JS commands are executed
  - Think back to the onload event we discussed
- If we try to affect an element on the page before it has loaded, we often will get no results, because the JS doesn't see the element until it finishes initially loading

JQuery Selectors

- CSS selectors (tag/class/id/attribute/etc)
- :contains(‘text’)
- :disabled / :enabled
- :empty
- :even / :odd
- :focus
- :has(selector)
- :not(selector)
- :hidden / :visible

JQuery Functions

- html() val()
- addClass()
- removeClass()
- hasClass()
- toggleClass()
- animate()
- fadeIn() fadeOut()
- fadeTo() fadeToggle()
- hide() show()
- slideDown() slideUp()
- slideToggle()
- bind() unbind()
- click() change()
- ready() resize()
- scroll() hover()
- keydown() keyup()
- mouseover() mousemove()

JQuery Functions

- append()
- after()
- before()
- clone()
- empty()
- replaceAll()
- replaceWith()
- text()
- wrap() unwrap()

- each()
- size()
- children()
- filter()
- find()
- not()
- parent()
- siblings()

JQuery Plugins

- There are several plugins available online that have been created by various developers
- Most of these are free and provide very straight forward instructions for use
  - Generally, you download the JS file and sometimes a CSS file
  - Include those into your page
  - Add a small amount of JS to initialize the plugin
  - Add the appropriate HTML for the plugin to operate on

TinyMCE Library

- http://www.tinymce.com/
- This library is used to turn a <textarea> box into a WYSIWYG (What You See Is What You Get) box
- This gives the extra ability to gather some HTML from whoever is filling out the form that we can store in a database and display on a page later