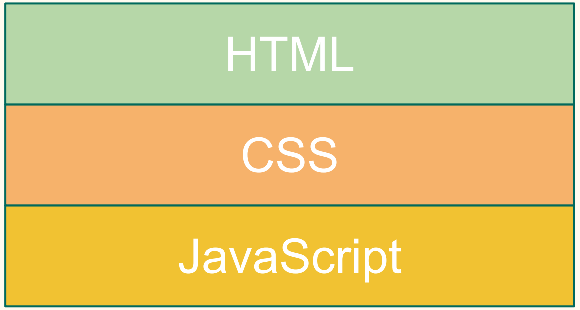
Web Basics

HTML, CSS, Javascript

Three Layers

- HTML, is the markup language that gives our web page structure. Like images, paragraphs, buttons, videos, etc.
- CSS, is a language of style rules that we use to style our HTML structure. Like color, size, position, fonts, etc.
- JavaScript, is adding functionality to your website. You can do a lot with just a few lines of JavaScript code.



HTML

What is HTML?

- HTML stands for Hyper Text Markup Language.
- HTML elements are used to structure web pages using markup.
- HTML elements are marked by things call tags
- They can also be nested within each other.
- Browsers do not render HTML tags, they use them to display content appropriately within the browser
- Tags normally come in pairs
- The end tag is written like the normal tag but prefixed with a forward slash

```
<tagname>put content here…</tagname>
```

Web Page Structure

- The `<!DOCTYPE html>` tells the browser that this is an HTML5 document.
- The `<html>` element is the root of the webpage.
- The `<head>` element contains meta information about the document.
- The `<title>` element specifies the title of the page.
- The `<body>` element is the content that is visually displayed in the browser.
- The `<h1>` defines a large heading.
- The `<p>` defines a paragraph

```
<!DOCTYPE html>
<html>
  <head>
    <title>HTML Demo</title>
  </head>
  <body>
    <h1>Hello World!</h1>
    <p>This is my first webpage.</p>
  </body>
</html>
```

Create A Basic Web Page

Try to create a basic web page using the structure listed to the right.

1. Make a new file and name it `index.html`
2. Open that file in Visual Studio Code or whatever IDE you prefer
3. Type out the structure to the right and replace the text in RED with text you wish to display.
4. Open your file in a browser by double clicking it in a windows explorer(or finder for Mac).

```
<!DOCTYPE html>
<html>
  <head>
    <title>Place Page Title here</title>
  </head>
  <body>
    <h1>Place Heading here</h1>
    <p>Place description here</p>
  </body>
</html>
```

Attributes

- HTML elements can have attributes
- Attributes are used to associate more information to an element.
- Attributes always are specified in the start tag.
- Attributes are usually defined with a key value pair. `name=”value”`
- One of the unique attributes that can be applied to any element is the `style` attribute.
- The style attribute is used to specify styling of the element. We’ll cover more in the CSS section.

```
<a href=”https://www.google.com”>
     Link Example
</a>

<img src=”my_image.jpg” alt=”Super Cool
Image”>

<p style=”color:red”>Example paragraph
text</p>
```

Ordered and Unordered lists

- Unordered lists are tags that render bullet points on the web.
- They start with a `<ul>` tag and have `<li>` tags within them. Each `<li>` tag is one bullet point.
- Ordered lists are tags that render a numbered list starting at 1.
- They are similar to unordered lists except they start with a `<ol>` tag instead of a `<ul>`.

```
<ul>
  <li>Coffee</li>
  <li>Water</li>
  <li>More Coffee</li>
</ul>

<ol>
  <li>Coffee</li>
  <li>Water</li>
  <li>More Coffee</li>
</ol>
```

```
 ●   Coffee
 ●   Water
 ●   More Coffee


1.   Coffee
2.   Water
3.   More Coffee
```

Other Tags

- `<h1>`, this is a heading tag. There are different size heading tags 1-6. `<h6>`
- `<p>`, paragraph tag, used for grouping of text.
- `<a>`, anchor tag, used for links. Should be paired with a `href` attribute.
- `<img>`, image tag, used for displaying images. Should be paired with a `src` attribute.
- `<hr>`, horizontal rule, draws a line across the screen.
- `<br>`, break tag, enters down a new line.
- `<div>`, division tag, this is the most commonly used tag in html. We use this to break our site into sections to be structured and styled.
- `<b>`, bold text. Deprecated
- `<strong>`, important text.
- `<i>`, italic text. Deprecated
- `<em>`, emphasized text.
- `<mark>`, marked/highlighted text.
- `<small>`, small text.
- `<sub>`, subscripted text.
- `<sup>`, superscripted text.
- `<ins>`, inserted text.
- `<del>`, deleted text.
- more......

Comments

- Comments are started with `<!--` and ended with `-->`
- Comments can be multiline

```
<!-- This is a comment -->

<!--
    Multi
    line
    Comment
-->
```

Exercise

Create your resume using html elements.

- Name and contact info
- Summary about yourself
- Work Experience
- Skills
- Education
- etc…

CSS

What is CSS?

- CSS stands for Cascading Style Sheet.
- CSS describes how web pages are to be displayed on the screen.
- CSS is an important part of websites, without it we'd have bland black and white HTML elements rendered on the screen.
- There are three ways to add CSS to HTML elements.
  - Inline, using the `style` attribute.
  - Internal, using a `<style>` tag in the `<head>` tag. (more on this later)
  - External, using an external css file. (more on this later)
- The most common and best practice is to use an external file to add css to your elements.

CSS styles

- There are many styles you can apply to elements, but there are some common ones you can use to get started.
- The `color` style is used to change the font color of text within an HTML element.
- The `background-color` style is used to change the background-color of an HTML element.
- You can add multiple styles to the same HTML element. They are separated by a semi-colon(;).

```
<p style=”color:blue”>This text will be
colored blue</p>
```

```
<p style=”background-color:#0000FF;color:white”>
     The background will render blue and the
     font will be white
</p>
```

```
/* Psst this is a comment in css */
```

Inline Styles

- Inline styles, are added by directly accessing the `style` attribute on html tags.
- Every html tag can be styled, but not every css style will modify html tags the same way. For instance you cannot change the font color of a `<br>` tag.

```
<div style=”background-color:blue”>
  Hello World
</div>

<p style=”color:red”>foo bar</p>

<div style=”text-align:center”>
  <h1>
    My Custom Heading
  </h1>
</div>
```

Colors

- In css, colors can be specified by using RGB, HEX, HSL, RGBA, and HSLA values.
- RGB stands for red, green, blue and is specify like so: `rgb(255,0,0)` is `red`. Values are from 0-255
- HEX, uses a similar approach as rgb except it is specified from 0-F and prefixed with a # symbol, #rrggbb. It is used like so: `#ff0000` is `red`.
- HSL, RGBA and HSLA are more custom ways of manipulating colors. The first two are probably the most common

Style A Basic Web Page

- Create a web page with the basic HTML elements and add some inline styles to them.
- Add 3 different types of HTML tags. Heading tags, paragraph tags, division tags, etc.
- Color the background different for each of them.
- Color the font different for each of them.
- Use a different font types.

Selectors

- A selector points to an individual html element you would like to style.
- There are many different types of selectors.
- Selectors are made up of 3 components.
  - The selector itself
  - The Declaration block
  - The Declaration
- The declaration block can contain many declarations and are surrounded by curly braces.
- Each declaration is made up of a name and a value, separated by a colon and ending with a semicolon.

```
selector {
  name:value;
  name2:value2;
}
```

Element Selector

- An element selector selects an element based on the name of the tag.
- For example you can make all `<p>` tags red or make all `<h1>` tags background color green if you wanted.

```css
p {
  color: #ff0000;
}
h1 {
  background-color: #00ff00;
}
```

Id Selector

- The id selector uses the id attribute of an HTML element to apply css to it.
- The id of an HTML element should be unique on the page.
- You can select an element in css by id with a hash(#) then the id name.
- Note: An id name cannot start with a number.

```
/* html code */
<div id=”helloWorld”>
  Hello World!
</div>

/* CSS using an id selector */
#helloWorld {
  text-align: center;
  color: rgb(255,0,0);
}
```

Class Selector

- The class selector is similar to the id selector in that it uses a tag attribute to select the tags.
- The class selector uses the class attribute to select all tags that match that class. So it will apply to any tags with that class.
- Just as the class will apply to multiple tags. Tags can also contain multiple classes. They are separated by a space.
- The class selector uses a period(.) followed by the name of the class.
- Note: A class name cannot start with a number.

```
/* html for the class selector */
<div class=”right”>
  Hello
</div>
<div class=”right red”>
  World!
</div>

/* css for class selector */
.right {
  text-align: right;
}
.red {
  color: red;
}
```

Internal Styling

- Styles can be specified in the head tag using the `<style>` tag.
- This allows you to style anything in the body that matches the selectors.
- Note: You can also select the `<body>` tag to be styled as well.

```
<html>
  <head>
    <style>
      h1 {
        color: red;
      }
    </style>
  </head>
  <body>
    <h1>Hello World</h1>
  </body>
</html
```

Adding an External CSS File

- In order to add an external css file you will need to specify a `<link>` tag in the `<head>` tag.
- The `<link>` tag needs two attributes, first what type of link it is(`rel`) and where it is located including the file name(`href`).
- The `type` is the type of content in the file.

```
<!DOCTYPE html>
<html>
  <head>
    <link rel=”stylesheet” type=”text/css”
href=”mystyle.css”>
  </head>
  <body>
    <h1>Heading Text</h1>
  </body>
</html>
```

Exercise

- First, you will need to go to github and clone this repo: https://github.com/cooler09/css-practice.git
- Within the repo you should see a couple different files. We'll be using the "css.css" file for this exercise.
- Please complete all of the TODO's inside the css file.
- You can check your work each step by opening the "index.html" file in your browser of choice.
- If you would like to see what the final website should look like, or validate whether you're done, please open the image "0781_lrg.jpg".

Order Matters!

We've now seen 3 different ways of adding css to your webpage, but let's say you have both an external style and an internal style. What will happen to the `<h1>` tag? Will it be red font or blue font?

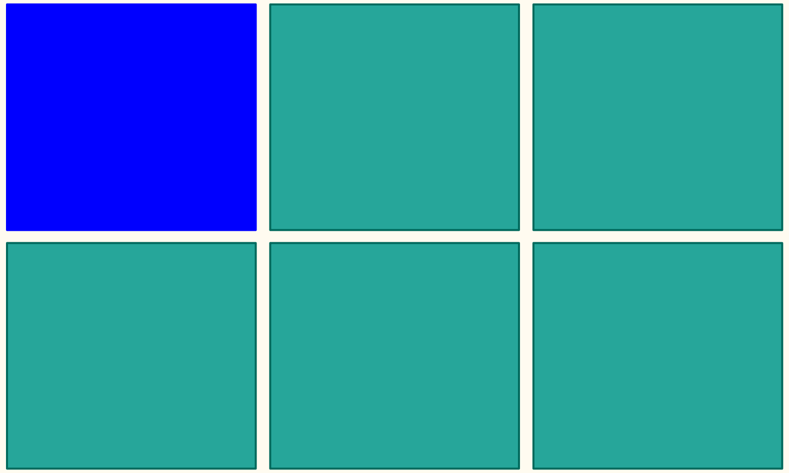
mystyle.css

```
h1 {
  color:blue;
}
```

```
<html>
  <head>
    <link rel=”stylesheet” type=”text/css”
href=”mystyle.css”>
    <script>
      h1 {
        color:red;
      }
    </script>
  </head>
  <body>
    <h1>Foo bar</h1>
  </body>
</html>
```

Group Exercise

- Create 6 tiles in the shape of a 2 by 3 grid.
- Each tile should be 250px by 250px.
- You may color the tiles any color you wish.
- The tiles should change colors when you mouse over them.



JavaScript

What is JavaScript?

- JavaScript is a object-oriented scripting/programming language.
- It is commonly used for creating interactive effects within web browsers.
- JavaScript is used for web development primarily because it can be stored on client side, as opposed to communicating to the server every time you need functionality.
- Note: Java and JavaScript are different languages.

How Do I Add JavaScript To A Page?

- Much like css, you can add JavaScript to your page 3 different ways.
  - Inline
  - Internal
  - External
- Just as css uses the `<link>` and `<style>` tags. JavaScript only uses the `<script>` tag to add functionality to a page.

```
// Comment in JavaScript

/*
Also a
Comment
*/
```

- Internal
    - JavaScript can be specified in the head or body tag using the `<script>` tag.
- External
    - Just as css can be loaded from an external file. So can javascript, but we only need the `<script>` tag and the `src` attribute.
- Inline
    - You can sometimes add JavaScript directly to certain HTML tags. For example the button has an `onclick` attribute that accepts JavaScript.

```
//Internal Example
<script>
  // JavaScript goes here
</script>

//External Example
<head>
  <script src="script.js"></script>
</head>


//Inline Example
<button onclick="myJSFunction()">
     Click me!
</button>
```

Order Matters in JavaScript!

JavaScript is a scripting language and runs from top down. If you have your script run before your HTML elements have rendered your code won't be able to find them.

In this Example the script gets executed before the `<h1>` tag has been rendered on the screen. There are a couple solutions to this, one of them being move the script below the elements, and another being use JavaScript code that waits until all the HTML elements have been rendered before executing your code.

```
<html>
  <head>
    <script>
      document.getElementById("foo");
    </script>
  </head>
  <body>
    <h1 id="foo"></h1>
    <!-- Move Script here -->
  </body>
</html>
```

Accessing an HTML Element

- Most of what you do in JavaScript you will need the ability to access an element to change how it renders or behaves.
- You can access an element in HTML multiple ways but probably the most useful and common way is the `getElementById()` function.
- In order to call this function we need to refer to the `document` object.
- The `document` object represents your webpage.
- If you want to access any element in an HTML page, you always start with the `document` object.

```
<html>
  <body>
    <div id=”myId”></div>
    <script>
      document.getElementById(“myId”);
    </script>
  </body>
</html>
```

Writing Output Data

- There are a couple ways of outputting data from JavaScript
    - `innerHTML`, writes into an HTML element.
    - `document.write()`, writes into the HTML output.
    - `window.alert()`, writes data to an alert box.
    - `console.log()`, writes data into the browser console.
- All have different purposes and use cases.

```
<script>
  //inner HTML example
  var element = document.getElementById(“myId”);
  element.innerHTML = “foo bar”;

  //document.write() example
  document.write(“foo bar”);

  //window.alert() example
  window.alert(“foo bar”);

  //console.log example
  console.log(“foo bar”);




</script>
```

Modifying Element Styles

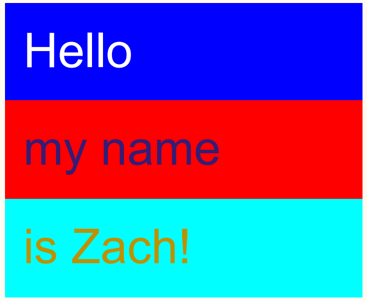
- You can modify element css styles through JavaScript. As with most things in JavaScript, there are many ways to do this.
- One way is by using the `style` property after selecting a specific HTML element.
- Once you do this you can directly modify or add css styles using the `style` property.
- Another way requires you to still select the element, but after you've done that call the function `setAttribute(attribute,value)`.

```
<script>
  //using the style property
  var ele = document.getElementById(“myId”);
  ele.style.color = “white”;
  ele.style.backgroundColor = “blue”;

  //using the setAttribute function
  ele.setAttribute(“style”,
      ”background-color:blue;color:white;”);
</script>
```

Exercise

- Add 3 different divs to your webpage and give them unique ids.
- Using JavaScript, set the background color and font color of 3 different div tags, to 3 different colors.
- Using JavaScript, set the `innerHTML` of each div tag to text of your choice.


Hello
my name
is Zach!

Let’s Take a Look at Some JavaScript Functionality!

Events

- HTML elements can fire events when certain actions occur.
- You can hook onto these events and execute JavaScript when said event occurs.
- There are a couple ways to add events, some ways you can add them is through HTML attributes or using JavaScript

```
<div onclick=”console.log(‘hello world’)”>
</div>

<div onclick=”changeMyInnerHTML(this)”>
</div>

<script>
  function changeMyInnerHTML(ele){
    ele.innerHTML = “foo bar”;
  }

  // hooking up the event onclick through javascript
  document.getElementById(“id”).onclick = helloWorld;

  function helloWorld(){
    console.log(“Hello World!”);
  }
</script>
```

Some Examples

- When a user click the mouse
- When a web page has loaded
- When an image has been loaded
- When the mouse moves over an element
- When an input field is changed
- When an HTML form is submitted
- When a user strokes a key
- etc...

Event Listeners

- You can attach an event listener to an element by using `addEventListener()` function.
- You can add many event handlers to one element.
- You can add many event handlers of the same type to one element. (like two “click” events)
- You can add event listeners to any DOM object not just HTML elements. (like the window object)
- You can remove an event listener by using the `removeEventListener()` function.
- The first parameter is the type of event(“click” or “mousedown”).
- The second parameter is the function we want to call when the event occurs.
- The third parameter is a boolean value specifying whether to use event bubbling or event capturing. This is optional

```
// example syntax
ele.addEventListener(event,function,useCapture);



var ele = document.getElementById(“foo”);

ele.addEventListener(“click”, function(){
  console.log(“clicked!”);
});

ele.addEventListener(“click”, myFunction);

function myFunction(){
  console.log(“clicked!”);
}
```