

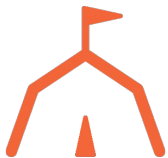
# **.NET C# Bootcamp**



# Frontend Technologies



# HTML

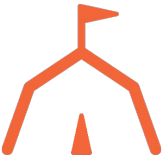


# **The Website as Communication**



# The Website as Communication

Every website on the internet uses HTML & CSS



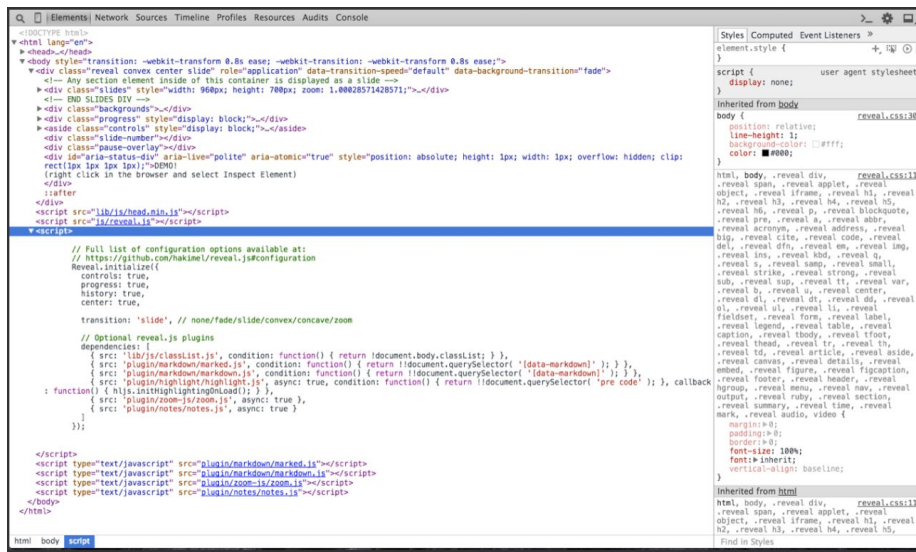
# The Website as Communication

...most of them use JavaScript as well, in one form or another.



# Demo!

## Right click in the browser and select Inspect Element



```
<!DOCTYPE html>
<html lang="en">
  <head>
    <body style="transition: ~webkit-transform 0.8s ease; ~webkit-transition: ~webkit-transform 0.8s ease;">
      <div class="reveal convex center slide" role="application" data-transition-speed="default" data-background-transition="fade">
        <!-- Any section element inside of this container is displayed as a slide -->
        <div class="slides" style="width: 960px; height: 700px; zoom: 1.0028571428571;">
          <!-- END SLIDES DIV -->
          <div class="background">
            <div class="progress" style="display: block;">
              <div class="control" style="display: block;">
                <div class="slide-number">
                  <div class="pause-overlay">
                    <div id="aria-status-div" aria-live="polite" aria-atomic="true" style="position: absolute; height: 1px; width: 1px; overflow: hidden; clip: rect(1px 1px 1px 1px);">
                      (right click in the browser and select Inspect Element)
                    </div>
                  </div>
                </div>
              </div>
            </div>
          </div>
          <script src="lib/js/head.min.js">
            <script src="js/reveal.js">
              <script>
                // Full list of configuration options available at:
                // https://github.com/hakimel/reveal.js#configuration
                Reveal.initialize({
                  controls: true,
                  progress: true,
                  history: true,
                  center: true,

                  transition: 'slide', // none/fade/slide/convex/concave/zoom

                  // Optional reveal.js plugins
                  dependencies: [
                    { src: 'lib/js/classList.js', condition: function() { return !document.body.classList; } },
                    { src: 'plugin/markdown/marked.js', condition: function() { return !document.querySelector('data-markdown'); } },
                    { src: 'plugin/markdown/highlight.js', condition: function() { return !document.querySelector('pre code'); } },
                    { src: 'plugin/notes/notes.js', condition: function() { return !document.querySelector('pre code'); } },
                    { src: 'plugin/zoom-js/zoom.js', condition: function() { return !document.querySelector('pre code'); } },
                    { src: 'plugin/notes/notes.js', condition: function() { return !document.querySelector('pre code'); } }
                  ]
                });
              </script>
            <script type="text/javascript" src="plugin/markdown/marked.js">
            <script type="text/javascript" src="plugin/markdown/highlight.js">
            <script type="text/javascript" src="plugin/zoom-js/zoom.js">
            <script type="text/javascript" src="plugin/notes/notes.js">
          </body>
        </html>
      </div>
    </body>
  </html>
```

Styles Computed Event Listeners

element.style {

script {

display: none;

user agent stylesheet

Inherited from body

body {

position: relative;

line-height: 1;

background-color: #fff;

color: #000;

Inherited from html

html, body, .reveal div, .reveal span, .reveal applet, .reveal object, .reveal iframe, .reveal h1, .reveal h2, .reveal h3, .reveal h4, .reveal h5, .reveal h6, .reveal p, .reveal blockquote, .reveal pre, .reveal a, .reveal abbr, .reveal acronym, .reveal address, .reveal big, .reveal cite, .reveal code, .reveal del, .reveal dfn, .reveal em, .reveal img, .reveal ins, .reveal kbd, .reveal q, .reveal s, .reveal samp, .reveal small, .reveal strike, .reveal strong, .reveal sub, .reveal sup, .reveal tt, .reveal var, .reveal w, .reveal x, .reveal y, .reveal z, .reveal dl, .reveal dt, .reveal dd, .reveal ul, .reveal ol, .reveal li, .reveal fieldset, .reveal form, .reveal label, .reveal legend, .reveal table, .reveal caption, .reveal tbody, .reveal tfoot, .reveal thead, .reveal tr, .reveal th, .reveal td, .reveal article, .reveal aside, .reveal canvas, .reveal details, .reveal embed, .reveal figure, .reveal figcaption, .reveal footer, .reveal header, .reveal hgroup, .reveal menu, .reveal nav, .reveal output, .reveal ruby, .reveal section, .reveal summary, .reveal time, .reveal mark, .reveal audio, video {

margin: 0;

padding: 0;

border: 0;

font-size: 100%;

font: inherit;

vertical-align: baseline;

Inherited from body

html, body, .reveal div, .reveal span, .reveal applet, .reveal object, .reveal iframe, .reveal h1, .reveal h2, .reveal h3, .reveal h4, .reveal h5, .reveal h6, .reveal p, .reveal blockquote, .reveal pre, .reveal a, .reveal abbr, .reveal acronym, .reveal address, .reveal big, .reveal cite, .reveal code, .reveal del, .reveal dfn, .reveal em, .reveal img, .reveal ins, .reveal kbd, .reveal q, .reveal s, .reveal samp, .reveal small, .reveal strike, .reveal strong, .reveal sub, .reveal sup, .reveal tt, .reveal var, .reveal w, .reveal x, .reveal y, .reveal z, .reveal dl, .reveal dt, .reveal dd, .reveal ul, .reveal ol, .reveal li, .reveal fieldset, .reveal form, .reveal label, .reveal legend, .reveal table, .reveal caption, .reveal tbody, .reveal tfoot, .reveal thead, .reveal tr, .reveal th, .reveal td, .reveal article, .reveal aside, .reveal canvas, .reveal details, .reveal embed, .reveal figure, .reveal figcaption, .reveal footer, .reveal header, .reveal hgroup, .reveal menu, .reveal nav, .reveal output, .reveal ruby, .reveal section, .reveal summary, .reveal time, .reveal mark, .reveal audio, video {

margin: 0;

padding: 0;

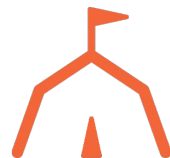
border: 0;

font-size: 100%;

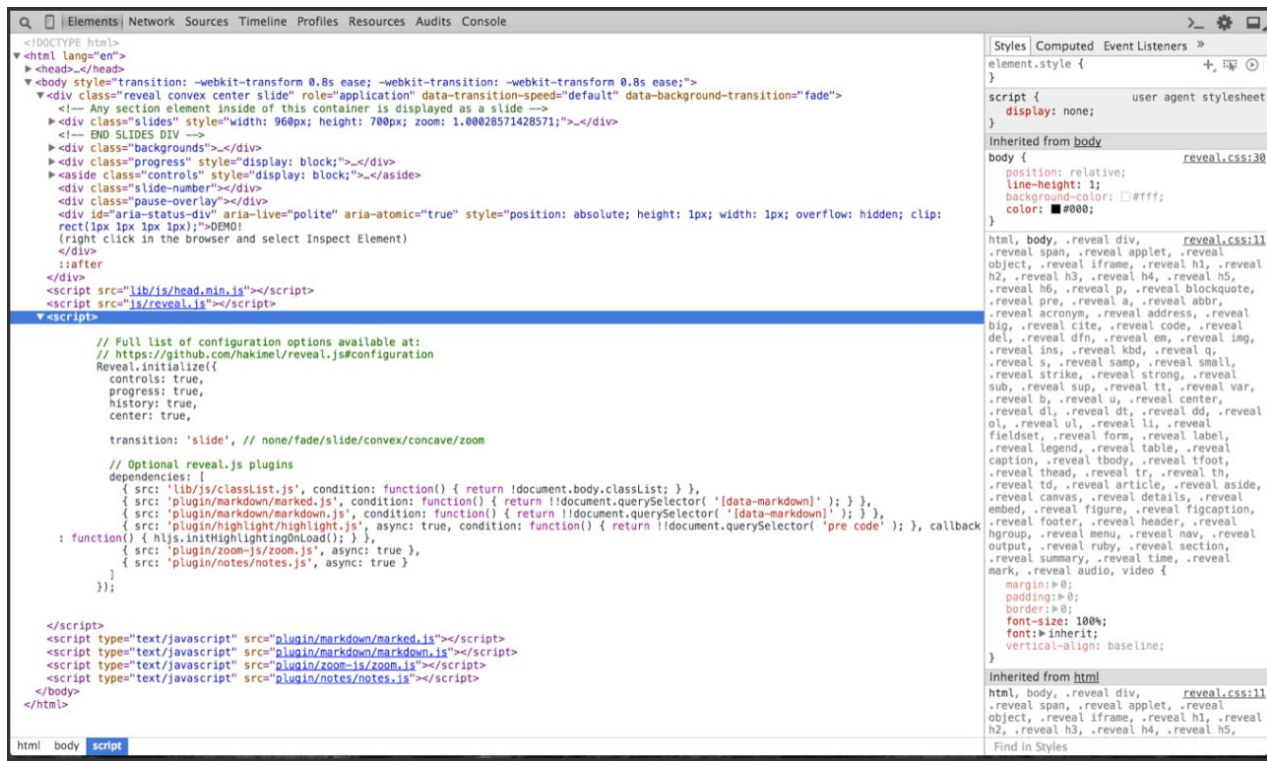
font: inherit;

vertical-align: baseline;

Find in Styles



# Demo!



The screenshot displays the browser's developer tools with the 'Elements' panel on the left and the 'Styles' panel on the right. The HTML structure shows a Reveal.js presentation with a main container, slides, backgrounds, and controls. The 'script' element is selected, showing the Reveal.js configuration and plugins. The 'Styles' panel shows the default styles for the selected element, including position, line-height, background-color, and color.

```
<!DOCTYPE html>
<html lang="en">
<head></head>
<body style="transition: -webkit-transform 0.8s ease; -webkit-transition: -webkit-transform 0.8s ease;">
  <div class="reveal convex center slide" role="application" data-transition-speed="default" data-background-transition="fade">
    <!-- Any section element inside of this container is displayed as a slide -->
    <div class="slides" style="width: 960px; height: 700px; zoom: 1.00028571428571;"></div>
    <!-- END SLIDES DIV -->
    <div class="backgrounds"></div>
    <div class="progress" style="display: block;"></div>
    <aside class="controls" style="display: block;"></aside>
    <div class="slide-number"></div>
    <div class="pause-overlay"></div>
    <div id="aria-status-div" aria-live="polite" aria-atomic="true" style="position: absolute; height: 1px; width: 1px; overflow: hidden; clip: rect(1px 1px 1px 1px);">DEMO!
    (right click in the browser and select Inspect Element)
    </div>
    :after
  </div>
  <script src="lib/js/head.min.js"></script>
  <script src="js/reveal.js"></script>
</script>

// Full list of configuration options available at:
// https://github.com/hakimel/reveal.js#configuration
Reveal.initialize({
  controls: true,
  progress: true,
  history: true,
  center: true,

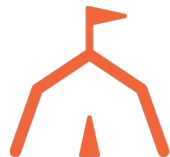
  transition: 'slide', // none/fade/slide/convex/concave/zoom

  // Optional reveal.js plugins
  dependencies: [
    { src: 'lib/js/classList.js', condition: function() { return !document.body.classList; } },
    { src: 'plugin/markdown/markdown.js', condition: function() { return !document.querySelector( '[data-markdown]' ); } },
    { src: 'plugin/markdown/markdown.js', condition: function() { return !document.querySelector( '[data-markdown]' ); } },
    { src: 'plugin/highlight/highlight.js', async: true, condition: function() { return !document.querySelector( 'pre code' ); } }, callback
    : function() { hljs.initHighlightingOnLoad(); } },
    { src: 'plugin/zoom-js/zoom.js', async: true },
    { src: 'plugin/notes/notes.js', async: true }
  ]
});

</script>
<script type="text/javascript" src="plugin/markdown/markdown.js"></script>
<script type="text/javascript" src="plugin/markdown/markdown.js"></script>
<script type="text/javascript" src="plugin/zoom-js/zoom.js"></script>
<script type="text/javascript" src="plugin/notes/notes.js"></script>
</body>
</html>
```

The 'Styles' panel shows the following styles for the selected element:

- element.style {  
}
- script {  
 display: none;  
 user agent stylesheet  
}
- Inherited from body  
body {  
 position: relative;  
 line-height: 1;  
 background-color: #fff;  
 color: #000;  
}
- Inherited from html  
html, body, .reveal div, .reveal span, .reveal applet, .reveal object, .reveal iframe, .reveal h1, .reveal h2, .reveal h3, .reveal h4, .reveal h5, .reveal h6, .reveal p, .reveal blockquote, .reveal pre, .reveal a, .reveal abbr, .reveal acronym, .reveal address, .reveal big, .reveal cite, .reveal code, .reveal del, .reveal dfn, .reveal em, .reveal img, .reveal ins, .reveal kbd, .reveal q, .reveal s, .reveal samp, .reveal small, .reveal strike, .reveal strong, .reveal sub, .reveal sup, .reveal tt, .reveal var, .reveal b, .reveal u, .reveal center, .reveal dl, .reveal dt, .reveal dd, .reveal ol, .reveal ul, .reveal li, .reveal fieldset, .reveal form, .reveal label, .reveal legend, .reveal table, .reveal caption, .reveal tbody, .reveal tfoot, .reveal thead, .reveal tr, .reveal th, .reveal td, .reveal article, .reveal aside, .reveal canvas, .reveal details, .reveal embed, .reveal figure, .reveal figcaption, .reveal footer, .reveal header, .reveal hgroup, .reveal menu, .reveal nav, .reveal output, .reveal ruby, .reveal section, .reveal summary, .reveal time, .reveal mark, .reveal audio, video {  
 margin: 0;  
 padding: 0;  
 border: 0;  
 font-size: 100%;  
 font: inherit;  
 vertical-align: baseline;  
}





# Terminology



# Terminology

Some key terms or phrases that are used as a matter of course in the software industry. You may know some or all of these, or you may have heard the terms but be unclear about their actual meaning. They are common jargon among developers



# Web Development

Web development is a broad term for the work involved in developing a website for the Internet. In industry parlance, 'web development' usually refers to the more code-related tasks such as programming JavaScript, coding HTML and CSS. It can even extend to tasks related to the back end server infrastructure such as creating web services and handling business logic for a company or product.



# Web Design

The process of planning & structuring a web site; specifically, the visual aspects and assets for the site. Recently, this job description has also begun to include interaction design. That is, designing the user experience (UX), information architecture, and the flow of the application or site.



# Web Site

A largely informational web page. While they may include dynamic elements and react to user inputs. The general purpose of a web site is to provide information about a person, business, product, or service.



# Web Application

A more recent term to indicate a web site whose sole purpose is not just informational, but rather functional. Web applications have become robust enough to do everything from our taxes, manage our personal calendars, or even do standard desktop publishing tasks.



# HTML 101



# Hypertext Markup Language

- Developed by Tim Berners-Lee
- Developed in 1989 at CERN
- Originally developed as a way to share documents





# HTML Structure

```
<html>
  <head>
    <title>My First Website!</title>
  </head>
  <body>

    <p>Look at the amazing content of my awesome website.</p>

  </body>
</html>
```



# CSS Presentation

```
body {  
  background-color: white;  
  font-size: 18px;  
  color: blue;  
}  
  
p {  
  text-align: center;  
  font-weight: bold;  
}  
  
img {  
  border: 1px solid black;  
  padding: 50px;  
}
```



# **HTML**

# **Hypertext Markup Language**



# HTML Elements

An element is an individual component of HTML



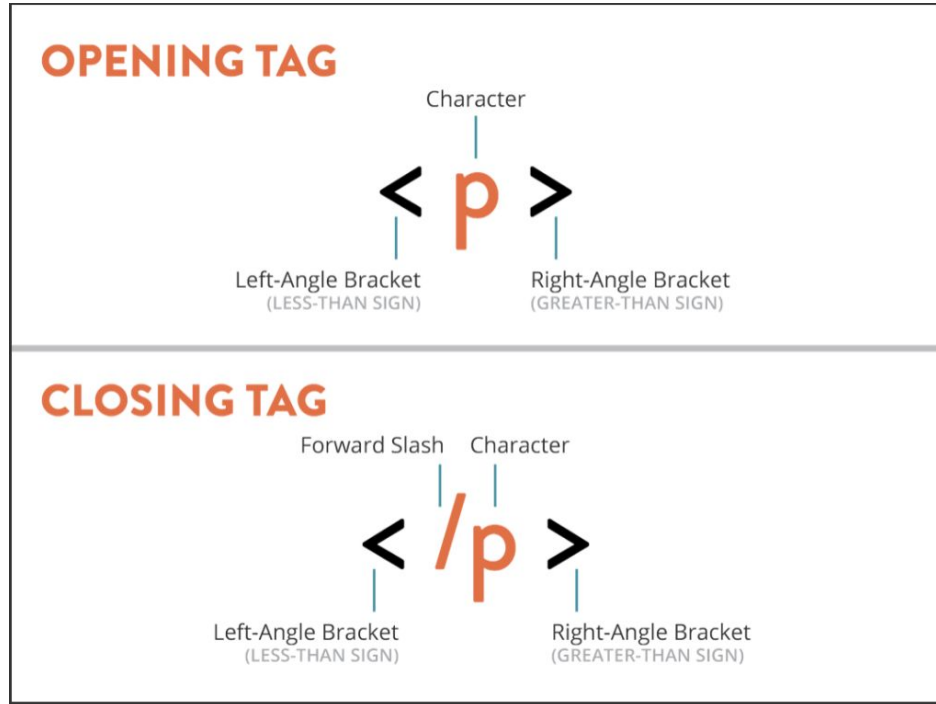
# HTML Tags

A tag marks the beginning and end of an element.

*Tags are containers.* They tell you something about the content between the opening & closing tags.



# Anatomy of an HTML Element



# Example

```
<p> This is a paragraph </p>
```

A paragraph element consists of an opening `<p>` tag and a closing `</p>` tag and the content between the tags.



# Different Kinds of Elements

Elements can be either container elements (they hold content) or stand-alone elements, sometimes called self-closing elements.





# Different Kinds of Elements

Paragraph elements are containers

```
<p>Hi, I contain content</p>
```

Image elements are stand-alone

```

```



# Attributes

1. Provide additional information about HTML elements
2. Attribute tags include `class`, `id`, `style`, `language`, and `src` (source)
3. Attributes are positioned inside the opening tag, before the right bracket



# Attributes

4. Attributes are paired with values.

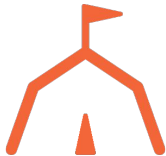
Key/value pairs are an important concept in programming.

5. Selected from a predefined set of possible attributes depending on the element.



# Values

1. Values are assigned to attributes
2. Values must be contained inside quotation marks



# Example Attributes

`<p id="trademark"> An inline  
element<p>`



# Example Attributes

```
<div class="container">  
  A bunch of stuff!  
</div>
```



# Doctype

The first element on every HTML page. It tells the browser to expect HTML and what version to use.



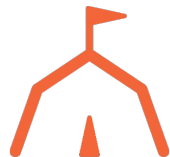
# Doctype

HTML 5

```
<!doctype html>
```

HTML 4

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01  
Transitional//EN"  
"http://www.w3.org/TR/html4/loose.dtd" >
```





# HTML Tag

After the doctype, all page content must be contained in the `<html>` tags



# Main HTML Tags

Tag	Description
<code>head</code>	Contains the page title and metadata
<code>body</code>	Contains all of the visible content
<code>title</code>	Optional tag. This is the name of your page. Nested in the <code>head</code> tag



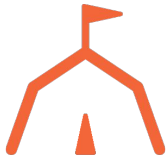
# Nesting

HTML elements 'nest' inside one another. The element that opens first closes last





# Content Tags



# Common Content Tags

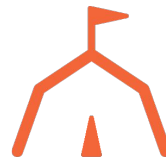
Tag	Description
<code>div</code>	Defacto container element
<code>p</code>	Used for body copy
<code>h1 thru h6</code>	Designating titles/subtitles
<code>ol</code>	Create a numbered list
<code>ul</code>	Create a bulleted list
<code>li</code>	List elements



# Sample HTML Page

```
<!DOCTYPE html>
<html>
<head>
  <title>My Totally Rad Website!</title>
</head>
<body>
  <h1>HELLO I AM JAMES AND THIS MY WEBSITE</h1>
  <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit,
    sed do eiusmod tempor incididunt ut labore et dolore magna
    aliqua. Ut enim ad minim veniam, quis nostrud exercitation
    ullamco laboris nisi ut aliquip ex ea commodo</p>

  <h4>My favorite things</h4>
  <ol>
    <li>Coding</li>
    <li>Board Games</li>
    <li>Quadcopters</li>
  </ol>
</body>
</html>
```



# Links





# Anchor Element

Links to other sites on the web (or within your project) are created using this element.

```
<a href="http://facebook.com">Facebook</a>
```

```
<a href="about.html">About Me</a>
```

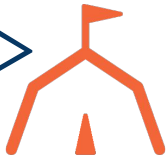


# Link Element

Unlike the anchor element. The `<link>` specifies relationships between the current document and an external resource.

Most often this manifests as how CSS files are included with an HTML file.

```
<link src="main.css" rel='stylesheet' />
```



# HTML Comments

Like any other good coding language, HTML offers comments. They operate like comments in any other language. They are ignored by the browser engine.

```
<!-- Hello, I am a comment. -->
```



# Tables



# Table Element

Tables are a way to represent complex information in a grid format. They are made of rows and columns.

Tables are compound elements (similar to lists).



# Table Element

Element	Description
<code>&lt;table&gt;</code>	Table element
<code>&lt;tr&gt;</code>	Table row
<code>&lt;td&gt;</code>	Table cell
<code>&lt;th&gt;</code>	Table header cell (optional)



# Table Example

```
<table>
  <tr>
    <th>Business</th>
    <th>Numbers</th>
    <th>Synergy</th>
  </tr>
  <tr>
    <td>Cell 1</td>
    <td>Cell 2</td>
    <td>Cell 3</td>
  </tr>
  <tr>
    <td>Cell 4</td>
    <td>Cell 5</td>
    <td>Cell 6</td>
  </tr>
</table>
```

Business	Numbers	Synergy
Cell 1	Cell 2	Cell 3
Cell 4	Cell 5	Cell 6



# Styled Tables

<b>Id</b> ▾	<b>Name</b> ▾	<b>Email</b> ▾	<b>Investments</b> ▾	<b>Action</b>
261	Alfred Alan	<a href="mailto:aaalan@gmail.com">aaalan@gmail.com</a>	Stocks	 
227	Alison Smart	<a href="mailto:asmart@biztalk.com">asmart@biztalk.com</a>	Residential Property	 
246	Ally Emery	<a href="mailto:allye@easymail.com">allye@easymail.com</a>	Stocks	 
212	Andrew Phips	<a href="mailto:andyp@mycorp.com">andyp@mycorp.com</a>	Stocks	 
218	Andy Mitchel	<a href="mailto:andym@hotmail.com">andym@hotmail.com</a>	Stocks	 
221	Ann Melan	<a href="mailto:ann_melan@iinet.com">ann_melan@iinet.com</a>	Residential Property	 
243	Ben Bessel	<a href="mailto:benb@hotmail.com">benb@hotmail.com</a>	Stocks	 
232	Bensen Romanolf	<a href="mailto:benr@albert.net">benr@albert.net</a>	Bonds	 
233	Brad Cole	<a href="mailto:bradc@hotmail.com">bradc@hotmail.com</a>	Stocks	 
241	Catherine Benchman	<a href="mailto:cathb@hotmail.com">cathb@hotmail.com</a>	Stocks	 
74 items found, displaying 1 to 10. [First/Prev] <a href="#">1</a> , <a href="#">2</a> , <a href="#">3</a> , <a href="#">4</a> , <a href="#">5</a> , <a href="#">6</a> , <a href="#">7</a> , <a href="#">8</a> [Next/Last]				

Table caption					
Header	Header	Header	Header	Header	Header
sample data	sample data	sample data	sample data	sample data	sample data
sample data	sample data	sample data	sample data	sample data	sample data
sample data	sample data	sample data	sample data	sample data	sample data
sample data	sample data	sample data	sample data	sample data	sample data
sample data	sample data	sample data	sample data	sample data	sample data
sample data	sample data	sample data	sample data	sample data	sample data
sample data	sample data	sample data	sample data	sample data	sample data





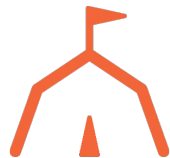
# A Note on Tables

You may be thinking it now or you may think later that tables would be a great way to position content for site...



# A Note on Tables

Don't do that. Just... don't. Trust me on this.  
It's not a good idea and it will give me an  
ulcer.



# What's Wrong With This Code?

Look at the following examples and tell me what is wrong with the code



```
<html>  
  <head>  
  <body>  
  </head>  
  
  </body>  
</html>
```



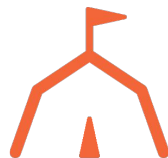
```
<html>
  <head>
    <title>The Best Site Evar!!
  </head>
  <body>

    <p>Check out this riveting content!</p>

  </body>
</html>
```



```
<p style=hotStuff>Check out this riveting content!</p>
```



# Folder Structure

This stuff is not exciting but its *important*.



# Making Folders

Let's make sure you know how to make a folder in your operating system. This might seem like a silly thing to be sure we all know how to do but it's important because you'll making lots of folders in this class.





# Folder Structure

In the beginning our projects will be very simple but as our projects grow in complexity it will be beneficial to organize them. Different devs have different takes on organization. Also, many frameworks, like .Net will dictate your folder structure.



# Folder Structure Continued

If we're doing a pure HTML/CSS site, beyond about 15 files, it's best practice to sort files like so: Each file type other than html gets its own folder. It's not uncommon to have the following folders: CSS, JS, and images



# Recap

You should understand and be able to use:

- HTML elements
- Proper nesting
- HTML comments
- Correct folder structure



# Code Along



# Let's Make Our First Website

1. Make a folder, name is `FirstWebsite`
2. Make a file, name it `index.html`
3. Follow along



# CSS



# Part Two: CSS



# Goals For Today

- CSS styles
- Linking HTML and CSS
- Inline and block elements
- Character codes





# CSS 101



# Cascading Style Sheets

Remember that 'cascading' part



# CSS in the Wild

```
.reveal.page .slides section {  
  padding: 30px;  
  min-height: 700px;  
  -moz-box-sizing: border-box;  
  box-sizing: border-box; }  
  
.reveal.page .slides section.past {  
  z-index: 12; }  
  
.reveal.page .slides section:not(.stack):before {  
  content: '';  
  position: absolute;  
  display: block;  
  width: 100%;  
  height: 100%;  
  left: 0;  
  top: 0;  
  background: rgba(0, 0, 0, 0.1);  
  -webkit-transform: translateZ(-20px);  
  transform: translateZ(-20px); }
```



# CSS Rules

Individual components of CSS are called rules



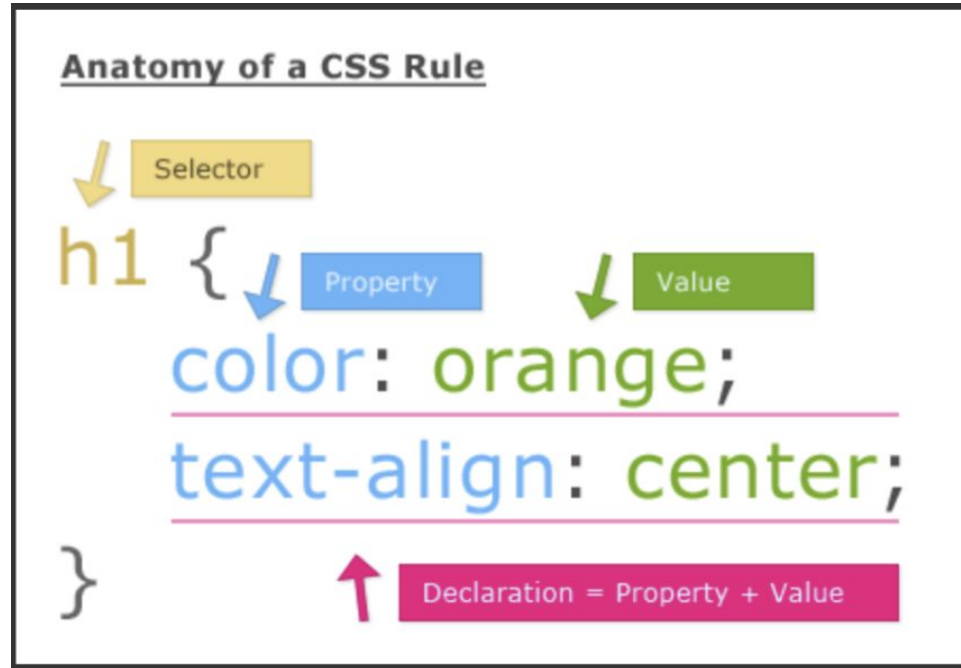
# CSS Rules

CSS rules are made up of two parts:

- One or more selectors
- One or more declarations
- The declaration must be inside curly braces that follows the selector

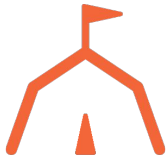


# CSS Syntax



# CSS Comments

Just like HTML, CSS offers comments:



# CSS Declarations

- They can be grouped together so that more than one declaration may be applied to a selected element.
- Declaration groups must be surrounded by curly brackets.





# CSS Declarations

- Declaration groups must be surrounded by curly brackets.
- Declarations must end in a semicolon.

```
.selector {  
  background-color: red;  
  color: white;  
  border: 1px solid black;  
  border-radius: 5px;  
}
```



# CSS Selector

The selector instructs the browser to search the page for any HTML element that matches the given criteria. It applies any applicable declarations to that element.



# CSS Selector- Element

Elements can be selected by their element name. In this case, all elements of that element type will be selected and have the styles applied.



# CSS Selector- Element

```
p {  
  position: absolute;  
  top: 0px;  
  left: -100px;  
}
```



# CSS Selector- Class

Elements can be selected based on HTML attributes such as class. In this case all elements that have a matching class attribute will be selected.



# CSS Selector- Class

```
.timer {  
    position: absolute;  
    top: 0px;  
    left: -100px;  
}
```



# CSS Selector - ID

Elements can also be selected based on HTML attribute ID. In this case only one element would be selected, as HTML IDs are intended to be unique.



# CSS Selector- ID

```
#fluffy {  
    position: absolute;  
    top: 0px;  
    left: -100px;  
}
```





# CSS Selector- Descendent Selectors

Selectors can be combined to become more specific. This example selects searches for any paragraph tag that is nested inside a div tag.

```
▼ div p {  
  position: absolute;  
  top: 0px;  
  left: -100px;  
}
```



# CSS Selector- Multiple

In addition a set of declarations can be applied to more than one selector by listing a number of comma-separated selectors.

```
.timer, img, div p, #kitty {  
    position: absolute;  
    top: 0px;  
    left: -100px;  
}
```



# Pop Quiz Hot Shot

Look at the following examples and tell me which Elements (if any) would be returned by the following selectors.



# Pop Quiz

```
p {  
  /* blah blah */  
}
```



# Pop Quiz

```
#fuzzy {  
    /* blah blah */  
}
```



# Pop Quiz

```
.blue {  
    /* blah blah */  
}
```



# Pop Quiz

```
.blue, #fuzzy {  
    /* blah blah */  
}
```



# Pop Quiz

```
div p {  
    /* blah blah */  
}
```





# Questions?



# CSS Properties



# CSS Properties

There are literally hundreds of CSS properties that are available for use. We don't have time to go over more than just a few. We will go over a few of the most common. However, the best strategy is to google for styling options as you're working.



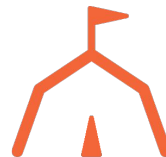
# Common CSS Properties

Property	Description
<code>background-color</code>	background color for an element
<code>color</code>	color of the text in an element
<code>font-family</code>	typeface for text
<code>font-size</code>	size for text (px,%,em,pt)
<code>font-weight</code>	used to bold text (if possible)
<code>text-decoration</code>	used for underline (mostly)
<code>height</code>	specifies the height of an element
<code>width</code>	specifies the width of an element



# Color in CSS

Method	Syntax	Description
Color name	<code>white</code>	A list of 140 predefined colors
hexidecimal	<code>#FF0000</code>	RGB values in hex 00-FF (0-255)
RGB	<code>rgb(255,0,187)</code>	RGB values in decimal numbers (0-255)
RGBA	<code>rgba(255,0,187,0.5)</code>	RGB values with an added alpha (opacity) value



# CSS Units

Method	Syntax	Description
em	1em	Scalable unit based on font size
pixels	16px	Fixed number of pixels
percent	120%	Percent value based on font size



# Folder Structure

- Still not exciting
- Still important



# Project Setup

Wherever you are going to keep your project work:

- Create a new folder called CSS-Exercise
- Inside that folder create two files called `index.html` & `main.css`





# Setup- Linking HTML & CSS Files

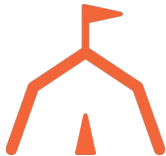
Add this element to your HTML page's head

```
<link href="main.css" rel="stylesheet">
```



# **Code Along**

## **Styling an HTML Page**



# Setup

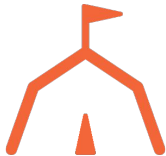
1. Make a new folder
2. Create new files `index.html` and `styles.css`
3. Setup a basic HTML page (doctype, html, head, body)
4. `link` the css file to the html page
5. Test the linkage
6. Add some HTML content (tags, classes, and ids)



# Simple CSS Styles

Include at least one of each:

- class selector
- id selector
- child selector
- multiple selectors



# Simple CSS Styles

Apply the following CSS properties:

- background-color
- color
- font-family
- font-size
- one or more additional properties of your choice



# CSS Positioning

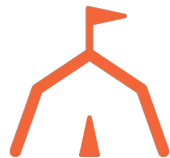


# Block vs. Inline Elements



# Block Elements

Block elements each appear on a new line of a web page, like paragraphs. Spatially, what is happening is that the block element takes up all of the horizontal space it can. It stretches to fill all the space to the left and right of the element within its parent container.



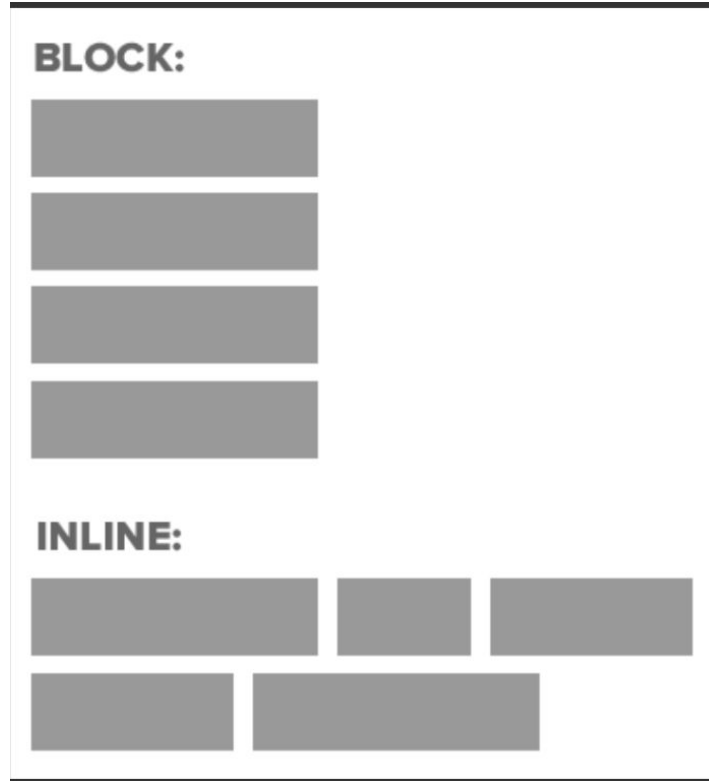


# Inline Elements

Inline elements are rendered without starting a new line. They appear side by side until reaching the edge of its parent container. Then it will start a new line.



# Block vs. Inline Elements



# Block vs. Inline Elements

- *Inline element:* `<a>` `<img>` `<span>`
- *Block element:* `<div>` `<p>` `<ul>` `<li>`  
`<table>`...and just about everything else.

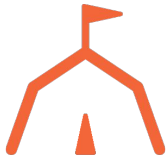


# Riddle Me This

How could you make a list of anchor tags (inline element), display on a page as a vertical list?



# JavaScript



# What is JavaScript



# What is JavaScript

JavaScript is a light-weight and object-oriented scripting language.

JS is a dynamic, cross-platform language originally used for creating interactive websites.



# What is JavaScript

JS is now popular server-side and client side-side.

JavaScript runs in all browsers and most modern browsers have a JavaScript console that can be used for writing, running, and debugging code.





# What is JavaScript

JavaScript is a loosely typed or dynamic language, so you don't have to declare a variable type ahead of time.



# Client-Side JavaScript

Client-side JavaScript extends the core language by supplying objects to control a browser and its Document Object Model (DOM).

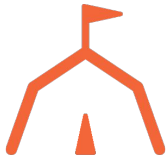


# Server-Side JavaScript

Server-side JavaScript extends the core language by supplying objects relevant to running JavaScript on a server.

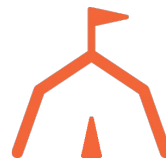


# JavaScript Compared to C#



# JavaScript Compared to C#

JavaScript	C#
Object-oriented. No distinction between types of objects. Inheritance is through the prototype mechanism, and properties, functions, and methods can be added to any object dynamically.	Class based. Objects are divided into classes and instances with all inheritance through the class hierarchy. Classes and instances cannot have properties or methods added dynamically.
Variable data types are not declared (dynamic typing).	Variable data types must be declared (static typing).
Cannot automatically write to hard disk.	Can automatically write to hard disk.



# Console Output



# Printing to the Console

When programming in JavaScript, much of the output is tracked via the console.

```
console.log("The statement you want to output.");
```



# Variables





# Variables

You must first declare a variable then initialize the variable. These can be done in one line or separate statements. The equals sign is used to assign a value.



# Loose Type

Because JavaScript is loosely typed, there are two ways of declaring a variable. You can use the keyword `let` or you can simply assign a value to a variable name.



# Variable Naming

Must begin with a letter, `_`, or `$`.

Must contain letters, numbers, `_`, or `$`.

As with Java and other languages,  
camel-casing and similar conventions are  
good practice.



# Let vs Var

In modern Javascript, it is best practice to use `let` to define your variables, as `let` uses similar Scoping rules to Java and C#.

`Var` will show up in older code snippets, but know its scoping rules are complex and it leads to many hard to track errors.



# Declare A Variable

```
let myVariable;  
myVariable = value;
```



# Initialize A Variable

```
let myVariable = value;
```



# Data Types



# Data Types

JavaScript has six primitive data types: string, number, and boolean, null, undefined, and symbol. The other data type is known as Object. The value of a variable differentiates its type.





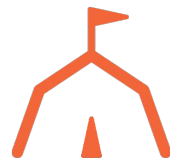
# Data Types

```
let myVariable = 25;           //number
let myVariable = "word";       //string
let myVariable = "25";         //string
let myVariable = true;         //boolean
let myVariable = null;         //null -- no value
let myVariable;                //undefined
```



# Keywords

break	continue	debugger
do...while	for	function
if..else	return	switch
try...catch	var	



# Operators



# Arithmetic

Addition	+
Subtraction	-
Division	/
Multiplication	*
Modulus	%
Increment	++
Decrement	--



# Assignment

standard assignment	=
plus equals	+=
minus equals	-=
assignment by multiplication	*=
assignment by division	/=



# Comparison

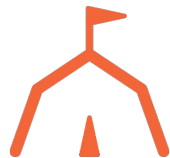
Equality	===
Inequality	!==
Greater than	>
Greater than or equal to	>=
Less than	<
Less than or equal to	<=



# Double Equals

- 'Shallow' equals ==
- 'Shallow' inequals !=

Performs a type coercion before checking equality.



# Type Coercion

Don't use double equals.

The equivalent to Java's `==` is `===`.

```
true == "true" // > false
true === "true" // > false
"1" == 1 // > true
"1" === 1 // > false
"1" != 1 // > false
```





# If/Else



# If/Else

```
if (condition) {  
    do something;  
} else {  
    do something else;  
}
```



# If/Else

```
if (condition) {  
    statement 1;  
} else if {  
    statement 2;  
} else if {  
    statement 3;  
} else {  
    statement 4;  
}
```



# Functions



# Function Declaration

In JavaScript, a function can be written for reusable blocks of code. You must declare a function, give the function some instructions, and call the function.



# Function Declaration

```
function myFunction() {  
    // do something;  
}  
myFunction();  
  
function myFunction(parameters) {  
    // do something;  
}  
myFunction(parameters);
```



# Function Declaration

```
function myFunction(x, y) {  
  return x * y;  
}  
myFunction(3, 4); // returns 12
```



# Part Two: DOM



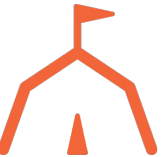


# Goals For Today

1. Objects in JavaScript
2. The Document Object Model (DOM)



# Interactivity



# **prompt()**

prompt() is a DOM method that allows a program to (you guessed it) prompt a user for input.



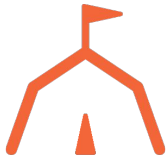
# prompt()

If the method is just called by itself, it simply returns the user's input in the form of a string. But the return from the method can also be stored to a variable and used. i.e.

```
let name = prompt("Please Enter your name");
```



# Code Challenge



# Setup

Go to [jsbin.com](https://jsbin.com)



# Step 1

## Work in Pairs

1. Write a simple program that accepts user input from a prompt. In the prompt, say “Please enter your name”.
2. Store the result in a variable.
3. Use an alert() function to print “Hello [name]” where [name] is the input entered by the user.



## Step 2

Simple Error Checking What happens if the user types some kind of unusable input (i.e. only one letter for their name). No one has a name that short! (For this example, assume this is true)





## Step 2

Add some logic to your function that checks the length of the input (ex. shorter than 2 characters) and alerts an error if their name is too short. Bonus: If the user input is not accepted, have the program ask again until the input is acceptable.



# Sample Solution

```
let name = "";
let done = false;
function ask() {
    return prompt("Please enter your name: ");
}

function helloName() {
    while (!done) {
        userInput = ask();
        if (isTooShort(userinput)) {
            alert("Your name is too short. \n do it again!");
        }
        else {
            alert("Hello " + userInput);
            done = true;
        }
    }
}
```



# Sample Solution Continued

```
function isTooShort(inputName){  
    return inputName.length < 2;  
}  
helloName(name);
```



# DOM



# Document Object Model

The document object model (DOM) is an interface which allows programs and scripts to dynamically access and update the content, style and structure of a document



# Document Object Model

The DOM is a W3C (World Wide Web Consortium) standard and includes the Core DOM, XML DOM, and HTML DOM



# Document Object Model

The HTML DOM is a standard model for HTML documents and defines how to get, change, add, or delete HTML elements



# Document Object Model

In order to integrate services like Twitter, Facebook, etc. into our own applications, we have to use their APIs (application programming interfaces). The HTML DOM is an API for manipulating HTML documents.





# Document Object Model

The DOM has a tree structure and identifies objects using nodes

```
let byNode = document.body;  
let htmlNode = document.body.parentNode;  
let childNodes = document.body.childNodes;
```

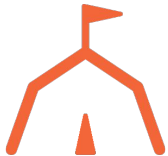


# Document Object Model

Watch what happens when you put this command in your js file.

```
document.write("JavaScript all the things!");
```

For extra fun, try that in the console on any website. In the DOM, your HTML page is known as the 'document'



# Document Object Model

It's possible to find elements on the HTML page by parent, sibling, or child node, but that's time-consuming and will make you crazy.



# Document Object Model

HTML elements can be located by tag

```
<li class="food">Pizza</li>  
<li class="food">Sushi</li>  
<li class="food">Hummus</li>
```

```
let listItems = document.getElementsByTagName('li');  
  
for (let i = 0; i < listItems.length; i++) {  
  let listItem = listItems[i];  
  //Do stuff with listItem  
}
```



# Document Object Model

... or by id or class

```
<p id="happy">A happy paragraph!</p>
```

```
var paragraph = document.getElementById('happy');
```



# Document Object Model

You can also select anything(id, tag, or class) but using `querySelectorAll`.

```
let paragraphs = document.querySelectorAll("p");  
let divID = document.querySelectorAll("#heading");  
let divClass = document.querySelectorAll(".news");
```



# Document Object Model

The attributes of HTML elements can also be accessed and modified through the DOM

```
let img = document.getElementById("myImage");  
img.getAttribute('src');  
img.setAttribute("src", "../images/newImage");
```



# Document Object Model

DOM nodes have a property called `innerHTML` which lets you access and modify the contents of the node





# Document Object Model

## Setup a simple HTML file

```
<!DOCTYPE html>
<html>
  <head>
    <title>JavaScript Demo</title>
    <script src="sample.js"></script>
  </head>
  <body>
    <p id="happy">A happy paragraph!</p>
  </body>
</html>
```



# Document Object Model

Setup sample.js like this. What happens when you load the page?

```
document.body.innerHTML = '<p>I changed the whole page!</p>'
```



# Document Object Model

Now change sample.js to this (remove all previous statements)

What happens? To make sure you notice, add border to all tags via CSS

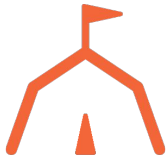
```
let paragraph = document.createElement('p');  
paragraph.innerText = "I made a new tag!";  
document.body.appendChild(paragraph);
```



# Questions



# jQuery



# Goals For Today

1. Introduction to jQuery
2. jQuery 101



# Introduction to jQuery



# What is a Library

A software library is a collection of functions. When you include a library in your code, you have access to all the functions contained in the library. jQuery is a big honkin' library packed full with functions intended to help you 'write less, do more' when working with the DOM





# What Does jQuery Help With?

jQuery simplifies DOM scripting and other common JavaScript tasks including: HTML element selection and manipulation, CSS manipulation, and JavaScript events and animations



# Facts About jQuery

- The most popular JavaScript library
- Extensive Docs
- Numerous tutorials online
- Used by 20 million websites



# 1 Fact

jQuery is still just JavaScript. If you understand JavaScript, you can understand jQuery.



# How Do I Get jQuery?

Two ways:

1. Download the library and store it locally in your project folder.
  2. Link to a live version of the library via CDN.
- Bonus: Display each object with a statement using concatenation. Ex: Sparky the dog.  
Coco the cat.



# Which Way is Best?

Each has pros and cons. Live code can change but jQuery is popular enough that there are stable versions of each major release of jQuery out there. Keeping it local is stable but bloats the overall size of your site.



# Which Way is Best? (Con't)

For this class, download it. The network might flake and then your site will not work. Plus a call will be made to get the whole library every time you refresh (which will be a lot).



# Which Way is Best? (Con't)

For active development it's fine to link. For production sites, I tend to lean toward sourcing it locally with your project.



# How To Download jQuery For This Class

1. Go to [jquery.com/download](https://jquery.com/download)
2. Click the link Download the uncompressed, development jQuery 1.11.1. This takes you to a raw text file.
3. Select All > copy > paste into empty file > save As 'jquery-1.11.1.js'
4. Place the saved file into a 'lib' folder in your site project





# How To Download jQuery For This Class

Or just link to a hosted CDN in a script tag on your page. Both script tags link to a working jQuery file.

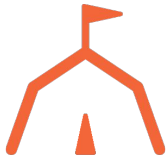
```
>head<
  <script src="lib/jquery-1.11.1.js"></script>
  <script src="http://code.jquery.com/jquery-1.11.0.min.js"></script>
>/head<
```



# jQuery Selectors

Does that selector syntax look familiar?

```
var divs = $("div");  
// All divs on the page  
var happyThing = $("#happy");  
// Element with id "happy"  
var rounds = $(".roundedCorner");  
// Elements with class "roundedCorner"
```



# jQuery Selectors

jQuery has hundreds of action that can be performed on any element

All actions are functions (or methods).

```
$(selector).action();
```



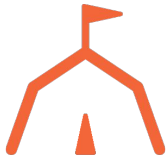
# jQuery Selectors

jQuery makes it simple to accomplish a huge number of common tasks such as updating element's attributes or CSS (and pretty much anything else)

```
var img = $('#myPicture');  
img.attr('src');  
img.attr('src', 'http://www.myPictureLivesHere.com');  
  
img.css('width');  
img.css('width', '200px');
```



# Questions



# jQuery 101



# jQuery

If you haven't done so yet, please download the example project named 'jquery-demo.zip' and 'jquery101-exercises.zip' from Slack.



# jQuery

jQuery interacts with the DOM to access and modify HTML elements.

`$()` is a jQuery function which turns whatever is inside the parentheses into a jQuery object.





# jQuery

.ready() is a jQuery function which tells a script to wait to execute until an HTML document has loaded.

```
$(document).ready();
```



# jQuery

The argument for the `.ready()` function is an anonymous function which will be executed as soon as the HTML document loads.



# jQuery

We can put other jQuery functions inside this outer function to make sure our code waits for the whole document to load before executing.



# jQuery

```
$(document).ready(function() {  
    $("header").slideDown('slow')  
});
```

```
$('img').fadeOut("slow", 0.5);  
var $img = $("img");  
$img.fadeTo("slow", 0.5);
```



# jQuery

In addition to elements, jQuery can use class and id selectors.

Any element that can be targeted with CSS can also be selected and modified using jQuery.

jQuery can also be used to modify an element's CSS styles.



# The .CSS Method

The .css() method is used to dynamically modify CSS styles.

```
$("#first").click(function() {  
    $("#first").css({  
        'background-color': 'blue'  
    });  
});
```



# jQuery



# Effects

In jQuery, functions like `fadeOut()` and `slideDown()` that add animations to a web page are called effects.





# Effects

In jQuery, functions like `click()` and `hover()` that refer to user interactions with the browser are called events.



# Questions



# Exercise



# In Your Project:

- Find the img tag in the page.
- In your JavaScript file add the following code (some of it is already there).

```
$(document).ready(function() {  
    $("img").mouseenter(function() {  
        $("img").fadeTo("fast", 0.5);  
    });  
});
```



# In Your Project (Con't)

- Save and reload the page. Mouse over the image, what happens?
- Can you figure out how to make the img return to full opacity when your mouse is no longer inside the element?



```
$(document).ready(function() {  
    $("img").mouseenter(function() {  
        $("img").fadeTo("fast", 0.5);  
    });  
    $("img").mouseleave(function() {  
        $("img").fadeTo("fast", 1.0);  
    });  
});
```



# jQuery



# **.click()**

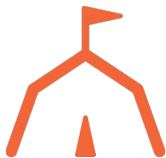
.click() is one of the most frequently-used event handlers in jQuery.

.click() captures a click on the selected element.





```
$("#div").click(function() {  
    // Do stuff when I click a div!  
});  
$("#img").click(function() {  
    // Do stuff when I click an img!  
});
```



# .click()

this is a keyword which tells jQuery, "I only want to do stuff to the specific element with which I'm interacting".

```
$(document).ready(function() {  
    $("div").click(function() {  
        $(this).fadeOut("slow");  
    });  
});
```



# jQuery Effects



# .hide()

.hide() hides a visible element

```
$("#myPicture").hide();
```



# .show()

.show() hides a visible element

```
$("#myPicture").show();
```



# **.toggle()**

.toggle() hides or shows an element depending on its current state (hides a visible element and shows a hidden element).



# **.fadeIn()**

.fadeIn() displays an element by fading it to full opacity.



# **.fadeOut()**

.fadeOut() hides an element by fading it to full transparency.





# .fadeto()

.fadeTo() adjusts the opacity of an element according to the opacity specified in the arguments.

```
$("button").click(function(){  
    $("div").fadeTo(0.25);  
});
```



# **.fadetoggle()**

.fadeToggle() hides or shows an element depending on its current state by animating its opacity.



# **.slideUp()**

.slideUp() hides an element with an upward sliding motion.



# **.slideDown()**

.slideDown() displays an element with a downward sliding motion.



# **.slidetoggle()**

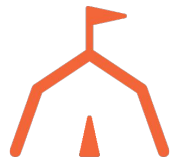
.slideToggle() hides or shows an element with a sliding motion depending on its current state.



# .delay()

.delay() specifies the time between each event queued on an element.

```
$("#button").click(function() {  
    $("#div.first").slideUp(300).delay(800).fadeIn(400);  
    $("#div.first").slideUp(300).delay(800).fadeIn(400);  
});
```



# Questions



# jQuery Events





# jQuery Events

An event refers to any kind of action a web page can recognize, including mouse clicks, hovering, scrolling, page loading, and resizing windows.



# jQuery Events

jQuery uses event handlers to define specific behavior which should occur in the web page when certain events are triggered.

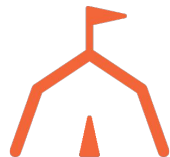
What are some real-world examples of events and the types of responses they trigger?



# .click()

.click() captures a click on the selected element.

```
$("#div").click(function() {  
    //Do stuff when I click a  
    !  
});  
$("#img").click(function() {  
    //Do stuff when I click an <img>!  
});
```



# **.dblclick()**

.dblclick() captures a double-click on the selected element.



# **.mouseenter()**

.mouseenter() is triggered when the mouse is moved over a selected element.



# **.mouseleave()**

.mouseleave() is triggered when the mouse is moved out of the area of a selected element.



# **.mousedown()**

.mousedown() is triggered when the mouse is moved over a selected element and the mouse button is pressed.



# **.mouseup()**

.mouseup() is triggered when the mouse is moved over the selected element and the mouse button is released.





# **.mousemove()**

.mousemove() is triggered when the mouse is moved within the area of a selected element.



# .load()

The `.load()` event is triggered on an element when the element and all its sub-elements have loaded.

```
$("#book").load(function() {  
    // Do stuff!  
});
```



# .unload()

The .unload() event is triggered when a user navigates away from the page (don't do that).

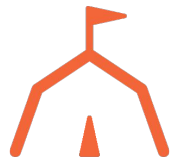
```
$(window).unload(function() {  
    alert("Thanks for visiting");  
});
```



# .error()

The `.error()` event is triggered when an element is not loaded correctly (for example, an image isn't found).

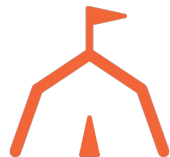
```
$("img").error(function() {  
    $(this).hide();  
}).attr("src", "missing.png");
```



# .resize()

The .resize() event is sent to the window when the size of the browser window changes.

```
$(window).resize(function() {  
    console.log("You resized the browser window");  
});
```



# .scroll()

The .scroll() event is sent to an element when the user scrolls to a different place in the element.

```
$(window).scroll(function() {  
    $("#pictureDiv").fadeOut("slow");  
});
```



# Questions



# Lab





# Rockin' With jQuery

Create a website that lists and demonstrates the jQuery topics we've covered so far. Include:

- Selectors
- Effects
  - Slide actions
  - Fade actions
  - Show / hide
- Events
  - click
  - hover
  - dblClick

