Introduction to JavaScript

What is JavaScript?

- ☞ A client-side scripting language
- ☞ Meant to run entirely on the user's browser
- ☞ Defined by the ECMAscript standard, published by the ECMA foundation
- ☞ JavaScript != Java, they're completely different languages

What is it used for?

- ☞ JavaScript allows you to interact with the DOM (Document Object Model), so you can do things like:
  - ☞ Showing and Hiding elements
  - ☞ Animating elements
  - ☞ Replacing elements with other elements
  - ☞ Making requests to the server without reloading the page
- ☞ The DOM is a programmatic representation of all of the HTML elements on the web page

Example of JS functionality

Don't worry about comprehension, just an example of what JS can do

```
function addNums(){
  num1 = document.getElementById('num1').value;
  num2 = document.getElementById('num2').value;
  document.getElementById('result').innerHTML = (parseInt(num1) + parseInt(num2));
}
```

Necessary HTML

```
Number 1: <input type="text" id="num1">
Number 2: <input type="text" id="num2">
<button id="add" onclick="addNums()">Click to add</button>
<br>
Result:
<span id="result">Result Here</span>
```

Writing and running JavaScript - in a separate file, included

Use your text editor of choice to write `.js` files (for instance: `main.js` or `script.js`)

```
// main.js
alert("Hello World");
```

Save them and include them in your HTML to run them

```
<script src="main.js"></script>
```

Writing and running JavaScript - in HTML file

You can also write JavaScript directly in your HTML:

```
<head>
  <title>My Great JS example page</title>
  <script>
    alert(‘hi!');
  </script>
</head>
```

Comments

There are two different ways to write comments in JavaScript:

```
// This is a single line comment

/* I'm
a
nifty
multi-line
comment! */
```

Comments are a great way to stay organized and leave notes in your code as to exactly what you're trying to accomplish

Script Output - Why?

As you write JavaScript code, you'll need to periodically check that what your code returns is what you actually want it to return

Script Output - How?

There are a few different ways to see the output of your script when you run it.

- ☞ **The Console** is a built-in tool in your browser where you can run JavaScript code directly. It also allows you to see the output of JavaScript you write in your editor
- ☞ **Alerts** are "pop-ups". You can see the output in a popup window as soon as the alert() function gets called
- ☞ **Logging to HTML** means to change the content of an HTML element with content of your choice, which could be script output

Using the Console

- ☞ The JavaScript console can be opened in Google Chrome by using Command + Option + J (or Ctrl + Option + J on Windows machines)
- ☞ JavaScript can be run line by line inside of the console
- ☞ You can also output to the console inside of a .js file using:

```
js
console.log("My Content");
```

Using an Alert

To display an alert, use the `alert();` function. You can do it directly in the console or in a script (a `.js` file):

```
alert("Hello World!");
```

Logging directly to HTML

You can also change the "innerHTML" of any element, meaning what is contained inside of it:

```
function change(){
  document.getElementById('el').innerHTML = "NEW TEXT";
}
```

```
<div onclick="change()" id="el">CHANGE ME</div>
```

In the above example, we're changing the content of a with the ID "el" from "CHANGE ME" to "NEW TEXT".

See the next slide for a line-by-line breakdown

Code Breakdown

```
// Line 1: Declare a function called change
function change(){
// Line 2: Inside of the function, retrieve the current
// HTML document. Inside of that document, retrieve
// the element with the ID 'el' using the getElementById
// function. Set the content inside of the element to say
// "New Text".
  document.getElementById('el').innerHTML = "NEW TEXT";
// Line 3: Close the function.
}
// In your HTML document, you'll need a <div> with the ID
// "el". The onclick attribute allows you to specify a
// function to run when the element is clicked, in this
// case "change()"
<div onclick="change()" id="el">CHANGE ME</div>
```

Your first JavaScript

- ☞ Create a JavaScript that sends an alert to the user that says "Hello World"
- ☞ If you've got this down, try inserting "Hello World" into an element on screen instead

Programming Basics

What you'll need before you get to the "fun stuff"

Data types in JavaScript

- ☞ Why are we putting any text being logged or alerted inside of quotes?
- ☞ By encapsulating our text in quotes, we are telling JavaScript the data inside is a string, a basic JavaScript data type

Basic Data Types

- ☞ String - `"Hello World"`
- ☞ Number - `5`, `5.5`, `1000` (all numbers in JS are floats)
- ☞ Boolean - `true`, `false`
- ☞ `Undefined` (no value)

Variables in JavaScript

- ☞ You may have learned about variables in a mathematical context, for instance:
  x = 0
  x + 10 = ?
  Answer: 10

- ☞ In JavaScript and in almost any programming language, a variable is simply a container for a value

Variables in JavaScript

☞ JavaScript variables can be equivalent to any data type

☞ Variables are defined like so:

```
   var name = "Zach";
var numberOfWidgets = 10;
var isCodingCool = true;
```

Basic Math

JavaScript can do all basic math. For instance, you could run this code in the developer console:

```
10 + 10;
› 20

var x = 100;
x * 40;
› 4000
```

Further data types: Arrays

- ☞ Arrays are used to hold a collection of data, of any data type. This one is full of strings:

```
["Snoopy", "Charlie Brown", "Patty", "Violet"];
```

- ☞ They can hold multiple data types:

```
[11, 15, 25, 48, 79, "elephant"];
```

- ☞ Arrays can also be stored in variables:

```
var class_names = ["Julie", "Sophie", "Rob", "John"];
```

Accessing Array items - indexes

- ☞ Once you've declared an array, you may want to retrieve the items inside of it using their indices
- ☞ The index of an item inside of an array corresponds to its position from the beginning of the array
- ☞ The first item always has an index of 0
- ☞ In this array, "charlie brown" has the index of 0 and "snoopy" has the index of 1:

```
["charlie brown", "snoopy"];
```

ARRAYCEPTION

An array can store other arrays:

```
// declare the first array
var toyotas = ["Camry", "Prius"];
// declare the second array
var porsches = ["Camero", "Boxer"];
// declare a third array that contains the first
// and second array
var cars = [toyotas, porsches];
```

This is called a multi-dimensional array

ARRAYCEPTION

To access the items nested inside of a multi-dimensional array, you'll use this syntax:

```
// declare your mult-dimensional array
var cars = [ ["Porsche", "Camero"], ["Camry", "Prius"] ];

// access the first array inside
// then the first item inside that array
console.log(cars[0][0]);

> "Porsche"

// access the first array inside
// then the first item inside that array
console.log(cars[1][0]);

> "Camry"
```

A note on semicolons

- ☞ Semicolons are traditionally used to end statements in JavaScript
- ☞ Code will still execute without them
- ☞ They should be used to indicate the end of a statement and it's good form to include them

Exercises

- ☞ Create a script with two variables assigned to two numbers. Add them together and output the result to the console
- ☞ Try to add two strings together and output the result to an alert
- ☞ Create a multi-dimensional array related to a subject that interests you. Output two of the items in sub-arrays to the console