JavaScript

For the browser… and beyond!

A few things about me…

- Oldest of 3 siblings
- She/her pronouns
- Grew up in Indianapolis
- Took a few programming courses in college
  - Java, Python, and C++
- Interned in Detroit after graduation
  - C# and .NET
- Currently a Software Engineer at LegitScript
  - Python, Ruby, Typescript, and the occasional JavaScript ☺
- Two dependents: a 12-month-old daughter and a 9-month-old sourdough starter

Why learn JavaScript?

- It's a must for web development
  - JavaScript is the basis of popular front end frameworks like Angular and React
- It's useful in almost all software engineering and development jobs
  - With the rise of Node.js, JavaScript (and its superset TypeScript) have become go-to choices for full-stack development.
- It's easy to get started (as we're about to do !)
  - The tools you need are already downloaded in your favorite browser!

Start programming in 3 steps

- On any web page, hit the F12 key to see developer tools
- Find the "console"
- Copy and paste the following code into the console and hit enter:

```
alert("Welcome to JavaScript!");
```

Where in the web is JavaScript?

https://www.w3schools.com/js/js_whereto.asp

- Embedded directly in HTML in <script></script> tags
- Referenced in external files
- Called from within HTML elements

HTML embedded JavaScript

```
<html>
<head>
    <script>
        function getGreeting() {
            alert("Hello! ");
        };
    </script>
</head>
<body>
    <button id="submit" onclick="getGreeting() ">Say hi!</button>
</body>
</html>
```

External file reference

index.html

```
<html>
<head>
    <script src="myScript.js"></script>
</head>
<body>
    <button id="greet" onclick="getGreeting() ">Say hi!</button>
</body>
</html>
```

myScript.js

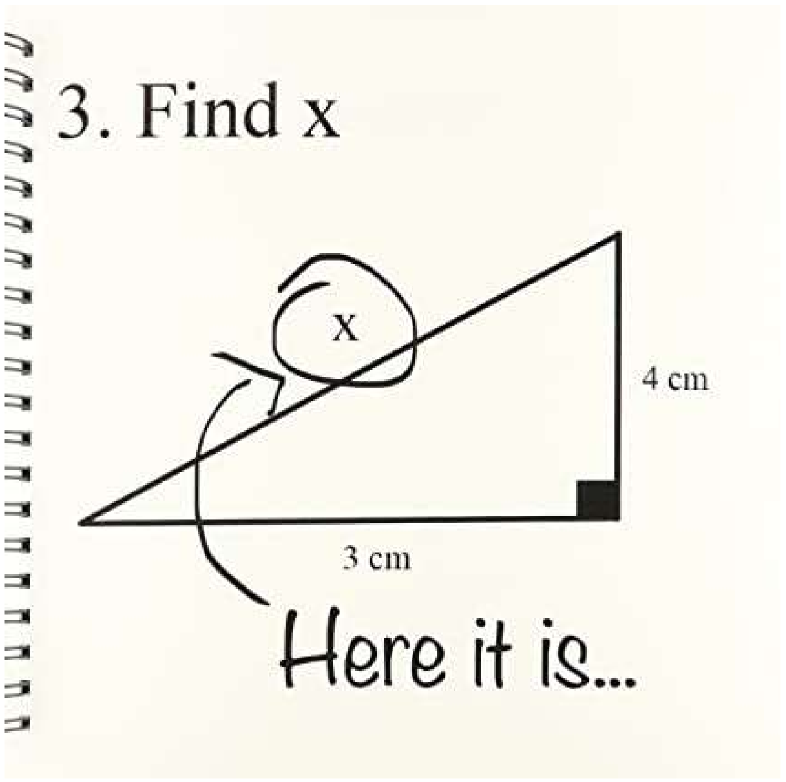
```
function getGreeting() {
    alert("Hello! ");
}
```

Within html elements

```
<html>
<head>
</head>
<body>
    <button id="submit" onclick="alert('Hello!');">submit</button>
</body>
</html>
```

Variables

https://www.w3schools.com/js/js_variables.asp



Variables

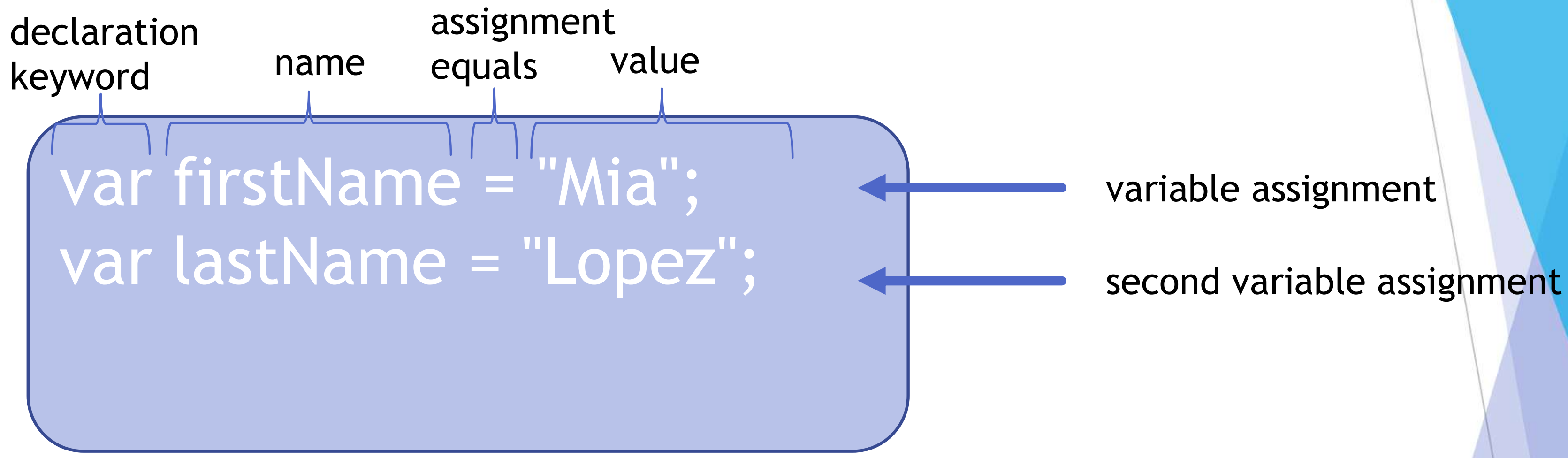
https://www.w3schools.com/js/js_variables.asp

- Variables represent values.
- Variables are a lot like the "nouns" of the language.
- Variables are used in statements or expressions.
- Types of variables:
  - String
  - Number
  - Boolean
  - Array
  - Many others (but we won't get into them today)

Variable declaration and assignment

- Variables are declared with a keyword
- There are three keywords that can be used:
  - let
  - const
  - var
- The differences are a bit nuanced for today's lesson, so we will only be using the "var" keyword.
- Variables can be assigned with an "assignment equals". This is often done in the same statement as declaration.
- Variables declared with the "var" keyword can be re-assigned as many times as needed.

Variable declaration and assignment

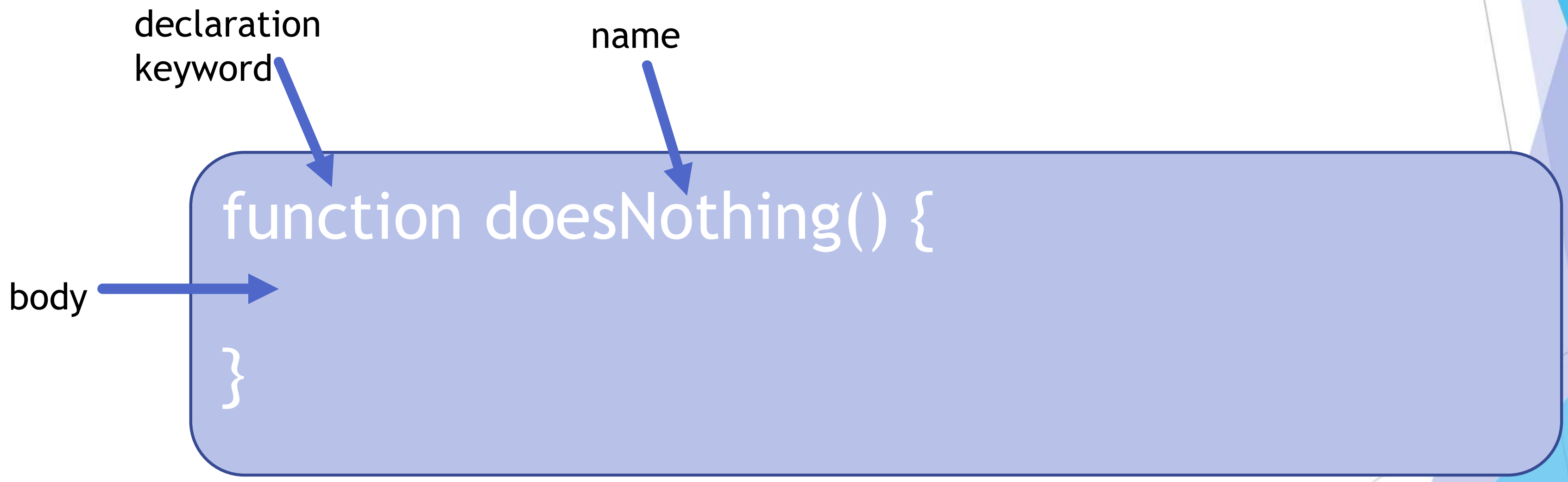


```
var firstName = "Mia";
var lastName = "Lopez";
```

Functions

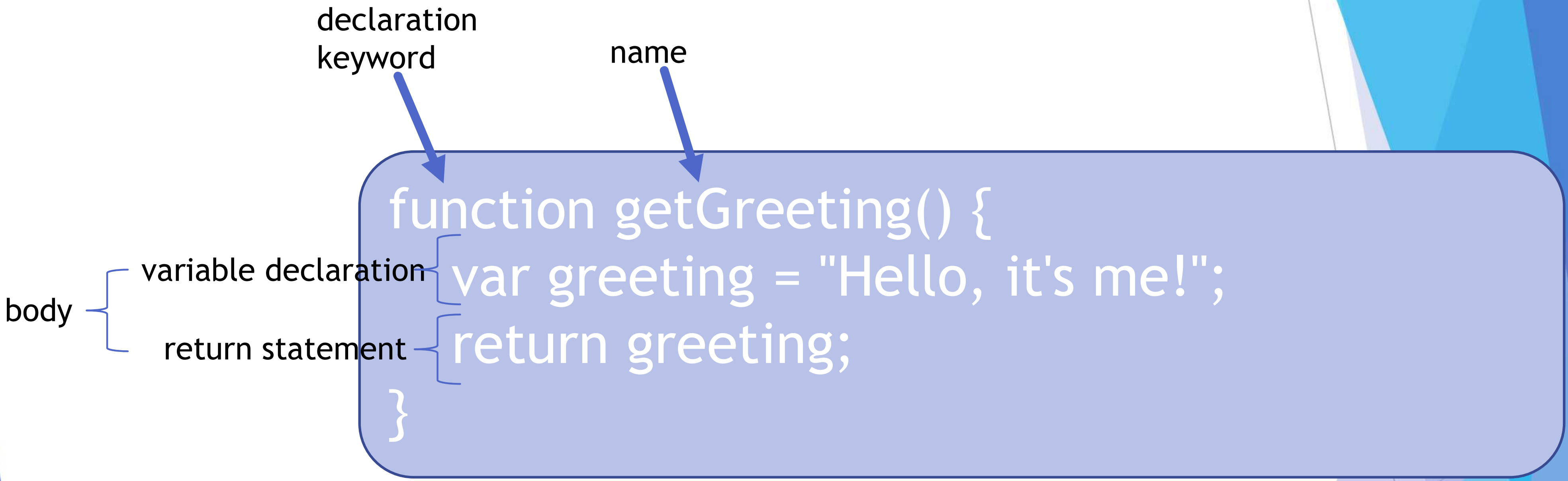
https://www.w3schools.com/js/js_functions.asp

Functions are subsets of code that perform a dedicated task. They are a lot like the "verbs of the language.



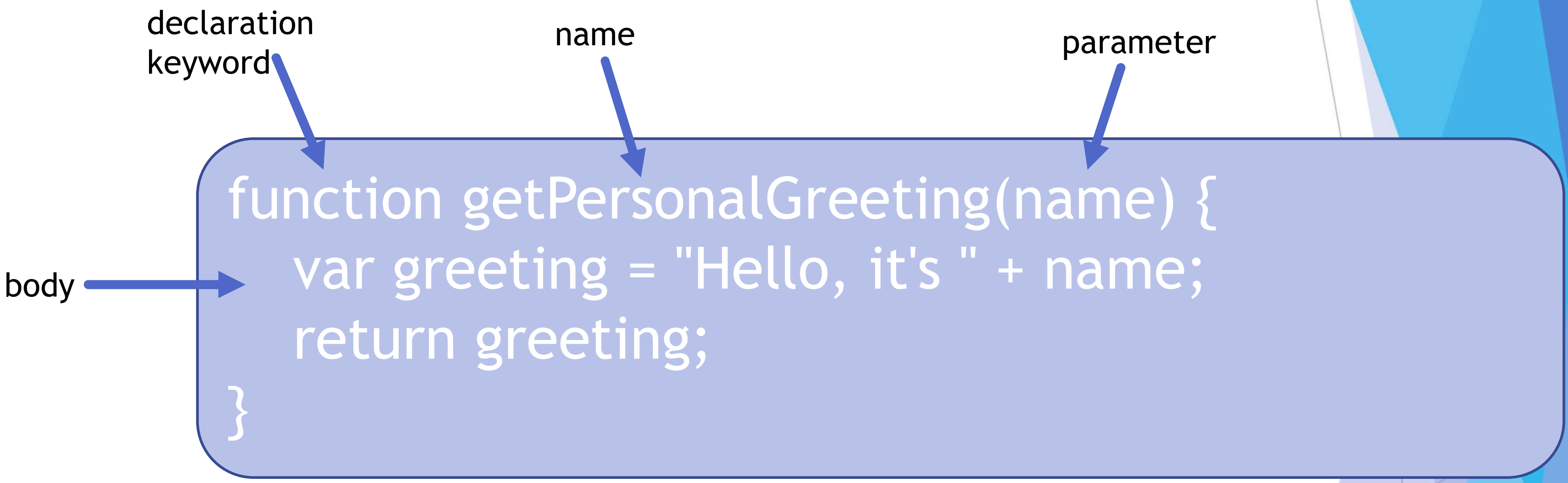
```
function doesNothing() {

}
```

Functions can return output



```
function getGreeting() {
  var greeting = "Hello, it's me!";
  return greeting;
}
```

Functions can accept input



```
function getPersonalGreeting(name) {
    var greeting = "Hello, it's " + name;
    return greeting;
}
```

Functions can take multiple parameters

```
function getGreeting(firstName, lastName) {
    var greeting = "Hello, it's " + firstName + " " + lastName;
    return greeting;
}
```

More on functions

- Functions are "called" from other blocks of code by their "name".
- Functions can have 0, 1, or many inputs (called parameters).
- Functions may or may not have output.
  - In JavaScript, if an output, or "return value" is not defined, it will return "undefined".

Semantics to make life easier

- JavaScript functions and variables are normally named in "camel case": the first letter of the first word in the name is lower case, and the first letter of each subsequent word is capitalized. Examples:
  - setFireToTheRain
  - chasePavements
  - helloItsMe
- Indentation is paramount for keeping code readable!
- For concision, it's always good practice to terminate a statement with a semicolon.

Starting our project

- Find and download the code to start your project on GitHub:
  - https://github.com/j-f-zhang/dln-encoder

Checkpoint #1

Open your project in VS Code (or text editor of choice)

Checkpoint #2

Open your project page in a browser!

Checkpoint #3

An alert that reads "Welcome to Javascript!" shows up after the "submit" button is clicked.

Strings
https://www.w3schools.com/js/js_strings.asp

- A string is a type of variable, surrounded by double quotes, single quotes, or backticks.
- There are several operators and functions that can be performed on strings
- One common operator is the plus operator, +
  - We can concatenate string variables and string literals to create new strings

```
var firstName = "Mia";
var lastName = "Lopez";
```

variable assignment

variable assignment

$$\texttt{var fullName = }\underbrace{\texttt{firstName}}_{\text{variable}}\texttt{ + }\underbrace{\texttt{" "}}_{\text{String literal}}\texttt{ + }\underbrace{\texttt{lastName}}_{\text{variable}}\texttt{;}$$

String methods

https://www.w3schools.com/js/js_string_methods.asp

- There are many string methods to do things such as:
  - Get the length of a string
  - Find a string within a string
  - Extract a part of a string
  - Convert to upper and lower case
  - Access a character at a given position

String method examples

```
var lastName = "Patel";

var length = lastName.length; // 5

var allCaps = lastName.toUpperCase() // "PATEL"

var firstCharacter = lastName[0]; // "P"

var positionOfT = lastName.indexOf("t"); // 2

var lastThreeCharacters = lastName.substring(2, 5); // "tel"
```

Checkpoint #4

An alert that reads "L****FMYY0MD" shows up after the "submit" button is clicked.

Checkpoint #5

Implement the getFirstInitial function!

Arrays

https://www.w3schools.com/js/js_arrays.asp

Arrays

- Arrays are a single variable made up of multiple values
- Like strings, values within an array are zero-indexed and can be accessed by position using square brackets []

```
var penguin1 = "Emperor";
var penguin2 = "King";
var penguin3 = "Galapagos";

var penguinSpecies = ["Emperor", "King", "Galapagos"];

penguinSpecies[0]; // returns "Emperor"
penguinSpecies[1]; // returns "King"
penguinSpecies[2]; // returns "Galapagos"
```

Operators

https://www.w3schools.com/js/js_operators.asp

Operators

```
// assignment operators:
   var x = 1; // assignment equals

// arithmetic operators:
   var x = 1 + 1; // addition
   var x = 1 - 1; // subtraction
   var x = 1 * 4; // multiplication
   var x = 1 / 4; // division
   var x = 1 % 1; // modulo
```

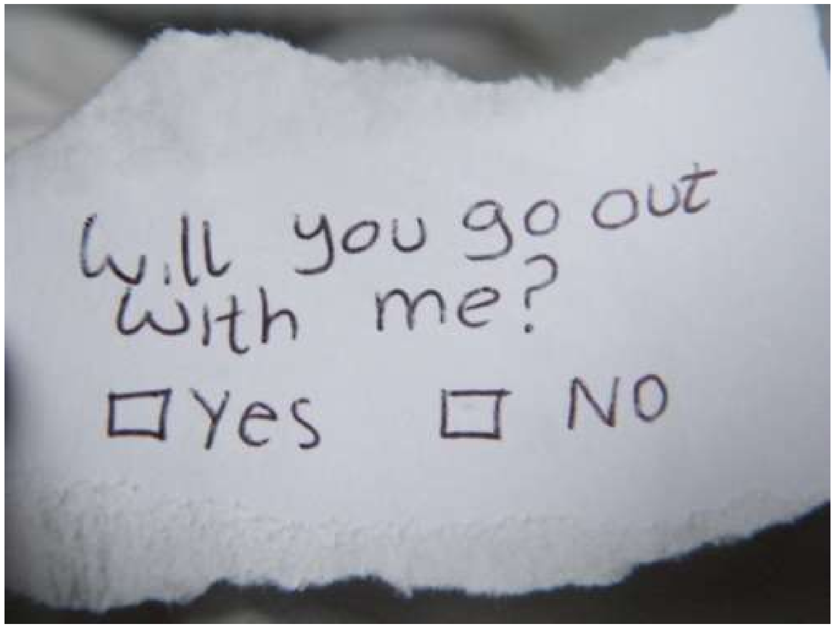
Modulo operator

- Using a modulo operator will return the remainder when the first number is divided by the second.
- This is a built-in operator in most programming languages, designated by the percent symbol %

```
var numerator = 13;
var denominator = 10;
var remainder = 13 % 10; // remainder's value is 3
```

Booleans

https://www.w3schools.com/js/js_booleans.asp



Will you go out with me?

☐ Yes ☐ NO

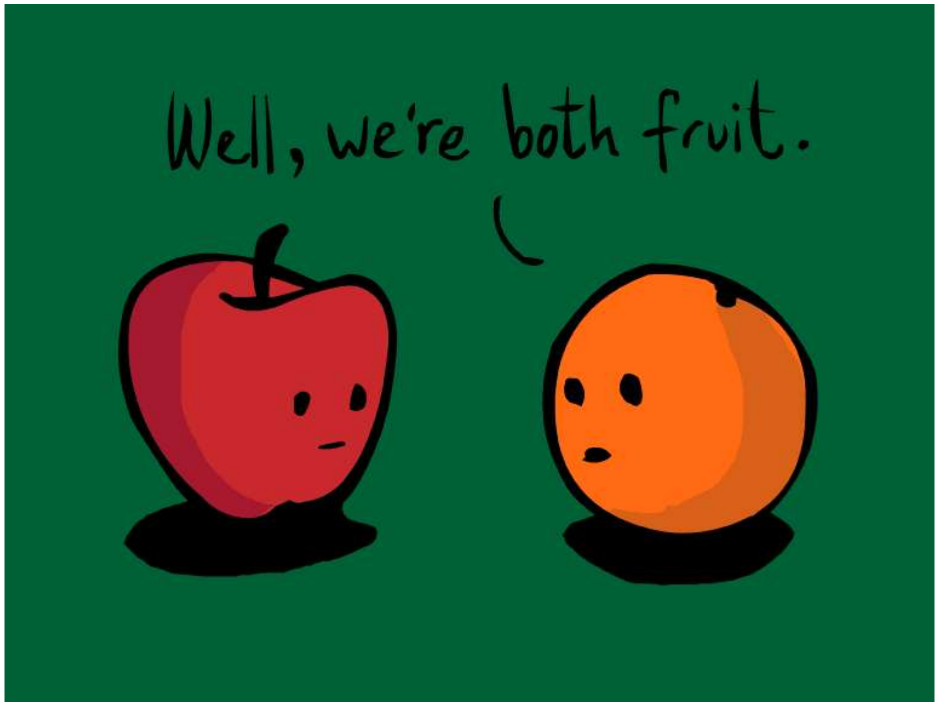
Booleans

- A Boolean is a type of variable that can either by true or false.
- We can create Booleans with true/false literals or operators: ==, >, <

```
var isTheSkyBlue = true;
var isTheGrassGreen = true;
var canOstrichesFly = false;
var canPenguinsFly = false;

(isTheSkyBlue == isTheGrassGreen); // this statement returns true
(canOstrichesFly == canPenguinsFly); // this statement also returns true!
(10 > 9); // this statement returns true
(10 > 13); // this statement returns false
```

Conditions and comparisons

https://www.w3schools.com/js/js_comparisons.asp
https://www.w3schools.com/js/js_if_else.asp



Comparison operators

```
var x = 5;

x == 8; // returns false
x == 5; // returns true
x == "5"; // true

x != 8; // returns true
x != 5; // returns false
x != "5"; // false

x > 8; // returns false
x < 8; // returns true
x >= 8; // returns false
x <= 8; // returns true


x < 5; // returns false
x <= 5; // returns true
x >= 5; // returns true
```

if and else

- If statements execute blocks of code if the specified condition is true
- Else statements execute a block of code if the specified condition is false

```
var isTheSkyBlue = true;

if(isTheSkyBlue){
        // this code block will be executed
        alert("The sky is blue!");
}

if(isTheSkyBlue){
        // this code block will be executed
        alert("The sky is blue!");
} else {
        // this code block will be ignored
        alert("The sky is not blue!");
}

if(10 < 9){
        // this code block will be ignored
        alert("Ten is less than nine!");
} else {
        // this code block will be executed
        alert("Ten is not less than nine!");
}
```

Logical operators

https://www.w3schools.com/js/js_comparisons.asp

- We can combine or negate Booleans with logical operators
  - And &&
  - Or ||
  - Not !

```
var isTheSkyBlue = true;
var isTheSkyGreen = false;

if(isTheSkyBlue && isTheSkyGreen){
        // this code block will be executed
        alert("The sky is blue and green!");
} else {
        // this code block will be ignored
        alert("The sky is not both blue and green!");
}

if(isTheSkyBlue || isTheSkyGreen){
        // this code block will be executed
        alert("The sky is blue or green!");
} else {
        // this code block will be ignored
        alert("The sky is neither blue nor green!");

}
```

While loops

https://www.w3schools.com/js/js_loop_while.asp

While loops

- Loops come in handy when we need to perform a block of code multiple times
- A while loop repeats a block of code as long as a condition we specify is true
- This is very handy when we know how many times we want to iterate

```
var lastName = "Patel";

var i = 0;

while(i < lastName.length){ // our condition is specified in the parentheses
      alert(lastName[i]); // alert the character at position i
      i++; // increment i (very important!)
}
```

Going beyond…

- Go back through the w3schools tutorials linked to in these slides – they're a great resource for you! https://www.w3schools.com/js/default.asp
- If you feel ready, implement the rest of the functions in your project with your new JavaScript skills!
- Get a GitHub account – upload and show off your great work! https://guides.github.com/activities/hello-world/
- Search for open source projects that interest you and start contributing!

Thanks!