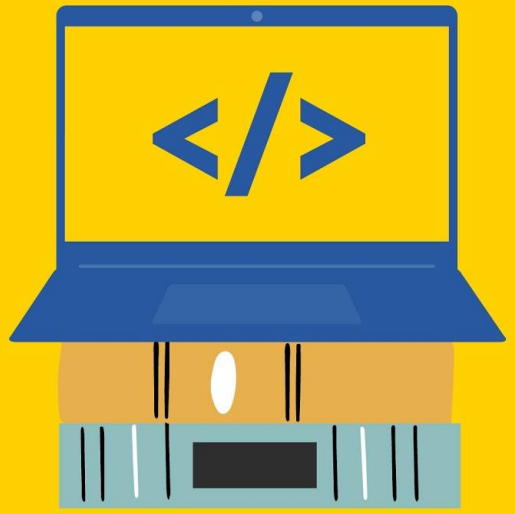


Front End Development

Session 6

Laying Our Foundations With ***JavaScript***


WELCOME

Learning Goals

In today's session, we will:

01 Define basic programming concepts in JavaScript, including Variables, Arrays, Conditionals, Loops

02 Practice applying variables, logging, and if/else statements

03 Explain the concept of browser events

04 Create example HTML buttons that run JS code when clicked.

Let’s review

Q. What kind of code are we learning in this course?

Q. Can you name a role, job, or industry that uses this kind of code?

Q. What three “languages” do we use to create websites?

Q. What does "HTML" stand for?

Q. How can we examine website code on our computers?

Q. In HTML, what's the difference between the <head> and the <body>?

What is the difference between Git and Github?

What does the CLI stand for? What would we use it for?

How do you format a link in HTML?

How do we save our work with Git and Github?

What does a tag do?

What does CSS stand for? Why do we use CSS?

What is the difference between *block* and *inline*?

What does “float” do?

What are the five positioning properties?

What is the difference between *block* and *inline*?

What does “float” do?

What are the five positioning properties?

What is a framework?

What does “open-source” mean?

What is responsive design?

Get Ready to Program!

Nothing Comes Easy

Learning to code requires two things:

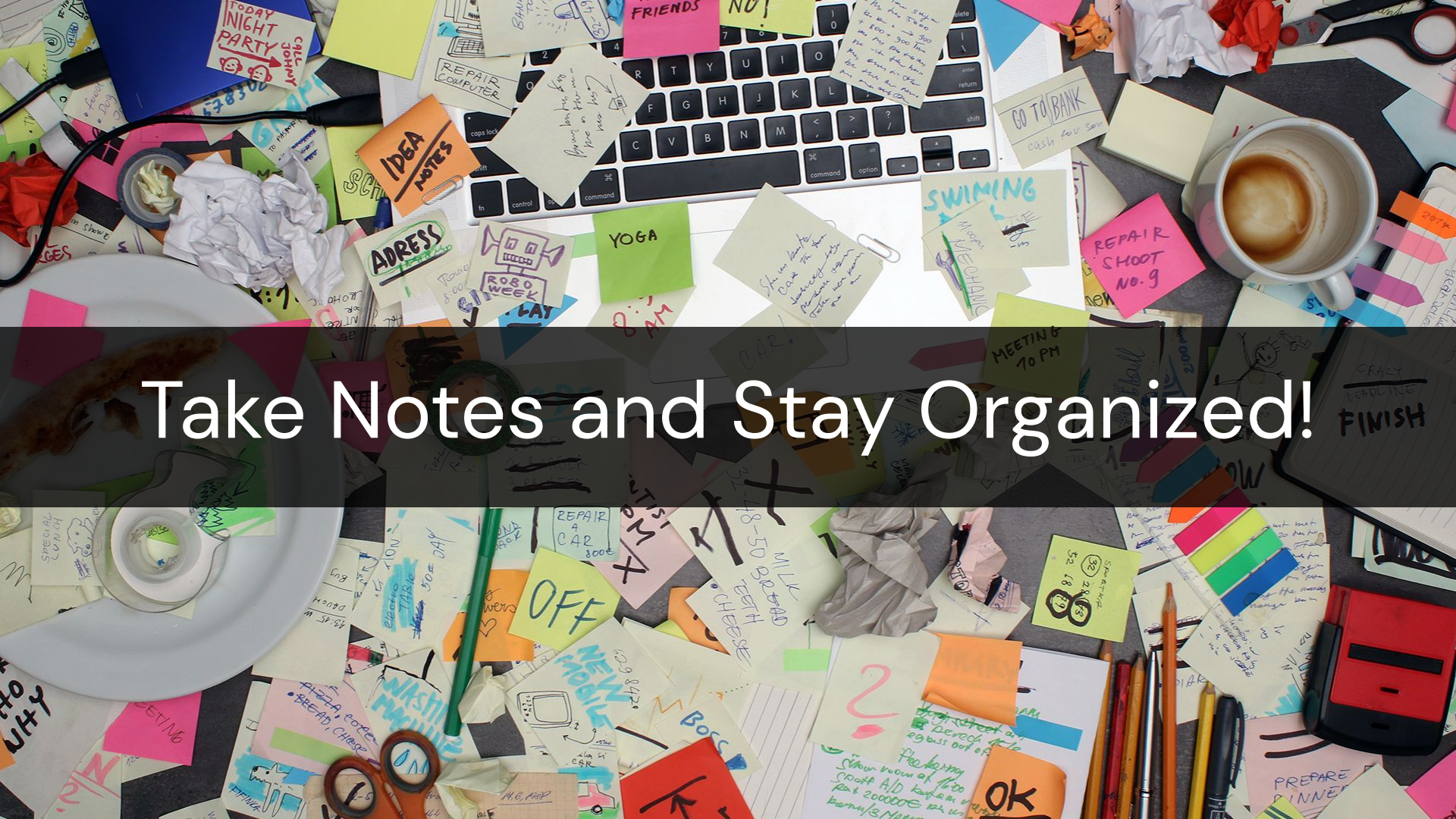
Persisting in the face of something that feels incredibly hard and confusing.

Maintaining the self-confidence necessary to believe that **YOU CAN DO THIS.**

You can’t tell whether you’re learning something when you’re learning it—in fact, learning feels a lot more like frustration.

—Jeff Dickey, author of Write Modern Web Apps

Your Brain on JavaScript



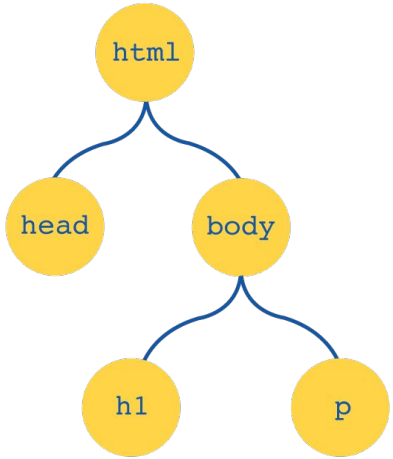
Take Notes and Stay Organized!

JavaScript

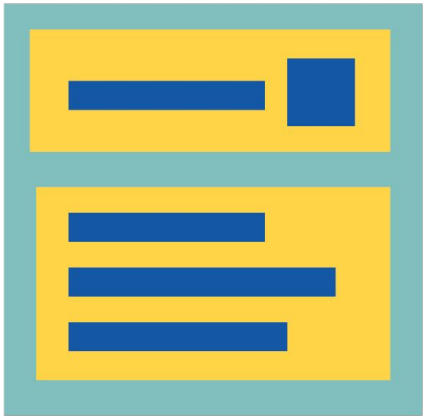
Prepare to become true coders!

Introducing: JavaScript

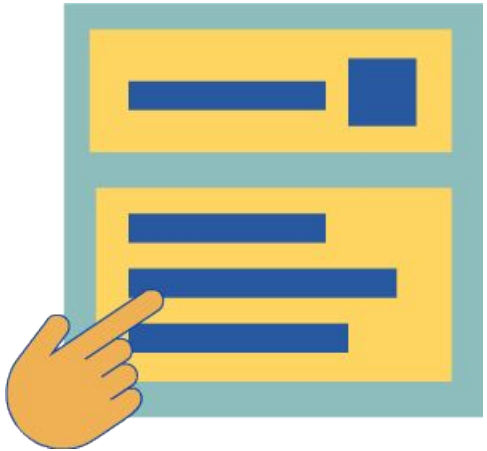
What Makes a Website?



HTML



CSS



JavaScript

JavaScript lets us host and interact with dynamic content on the web, like data or multimedia.

JavaScript Definition

JavaScript is one of ***three*** modern web programming languages:

HTML	CSS	JavaScript
Write content.	Format content.	Create interactive applications that log user input, change what's displayed, animate elements, and more!

Variables

The most basic building block in JavaScript are variables.

Variables are used to store and retrieve data.

A **variable** is a named location for a value that gets stored in the browser's memory when a program is run.

Variable Basics

Variables store information, so the best way to make our code easier to read and write is to give our variables **descriptive names** - names that tell us what each variable should be doing!

var	playerName	=	"Layla"	;

Variable Syntax

Var keyword	Variable name	Assignment	Value	Termination
var	playerName	=	"Layla"	;

JavaScript Data Types: *String*

“**String**”: fancy word for written text

*Notice the quotes - this tells our variable that the entry “Snow White” is a **String value** - e.g. a piece of text.*

```
var name = "Snow White";
var dwarfCount = 7;
var isSleeping = true;
```

JavaScript Data Types: *Integer*

“**Integer**”: fancy word for numeric data

Notice that we gave our variable a descriptive name to help us remember what information we are storing!

```
var name = "Snow White";
var dwarfCount = 7;
var isSleeping = true;
```

JavaScript Data Types: Booleans

“**Booleans**”: fancy word for TRUE / FALSE statements

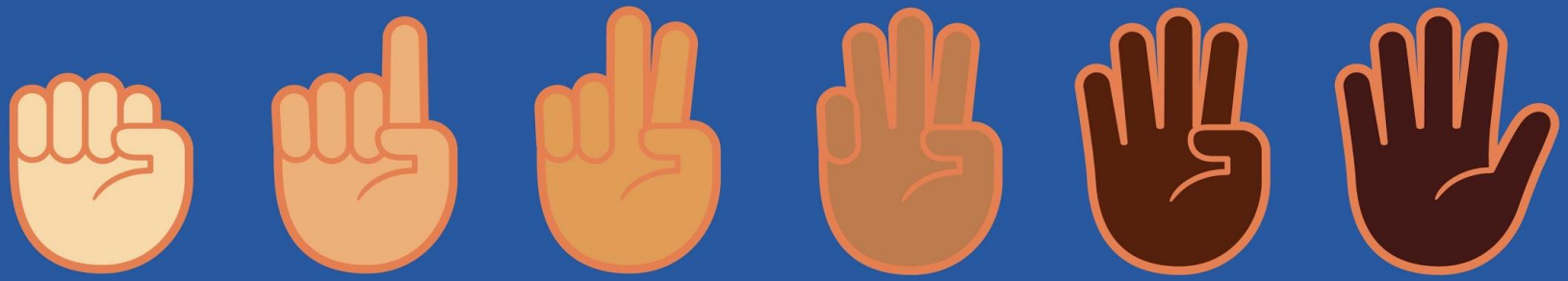
We use Booleans to store simple “binary” information about a subject. What information are we storing here?

```
var name = "Snow White";
var dwarfCount = 7;
var isSleeping = true;
```

Instructor Demonstration:

Variables and *Comments*

Suggested Time:
5 minutes



Fist to five

Let’s

Activity: Pizza Variables

1. Open our instructions in Canvas
2. Add your code.
3. When you are done, open your file in Chrome and check it out!

Suggested Time:
15 minutes

Let’s review

Questions?

Working with the Console

`console.log` is a simple function that prints a result to our console so we can “debug” our code.

How do you comfort a **JavaScript bug**?

You "console" it!

Console.log

`console.log` can be viewed in the browser's console and is very useful during development!

```
var quick = "Fox";
var slow = "Turtle";
var numbers = 121;

// The console.log() method is used to display data in the the browser's console.
// We can log strings, variables, and even equations.

console.log("Teacher");
console.log(quick);
console.log(slow);
console.log(numbers + 15);
```

Instructor Demonstration:

console.log

Suggested Time:
5 minutes

`console.log` is a `function` that prints a message which can be viewed in our browser's JavaScript engine.

A **function** is a predefined action that can be "called" or invoked later in your code

Function

`function` keyword

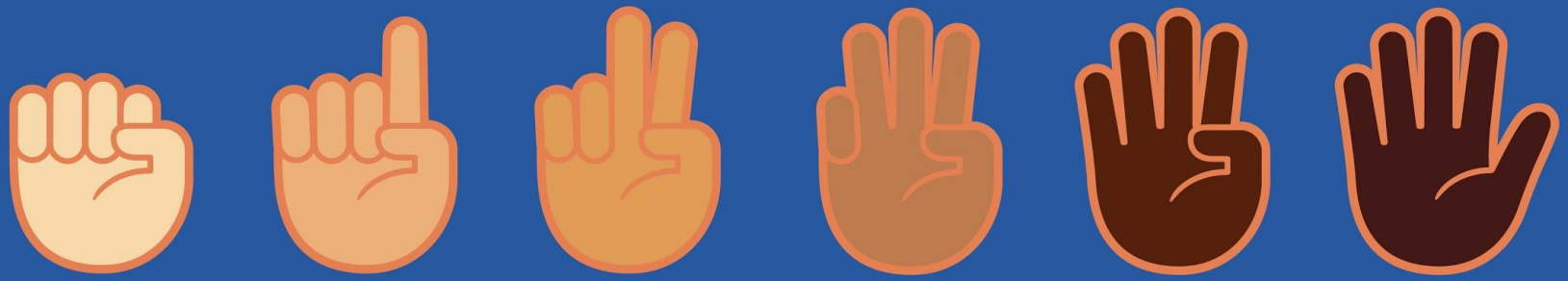
Function parentheses

Function name

Function block opening brace

```
function fight() {
  window.alert("The fight has begun!");
}
```

Function block closing brace



Fist to five

Let’s

Activity: Pizza Console

1. Break into pairs
2. Open instructions in Canvas
3. Modify the code.
4. Share your work with a partner!

Suggested Time:
10 minutes

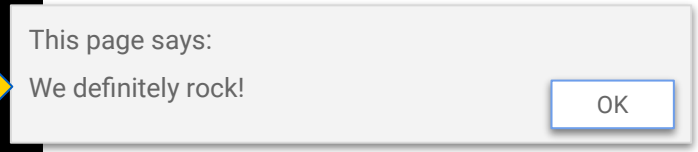
Let’s review

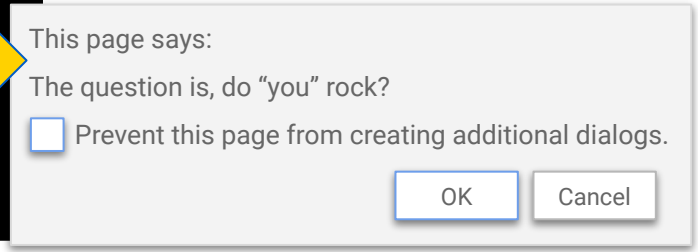
JavaScript Alerts

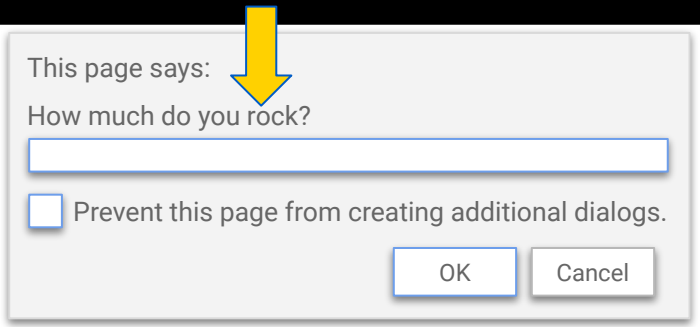
Alerts, **prompts**, and **confirms** create popup windows in the browser when run.

Examples: Alerts, Prompts, Confirms

```
// Alert
alert("We definitely rock!");

// Confirm
var doYouRock = confirm("The question is, do *you* rock?");

// Prompt
var howMuchRock = prompt("How much do you rock?");
```



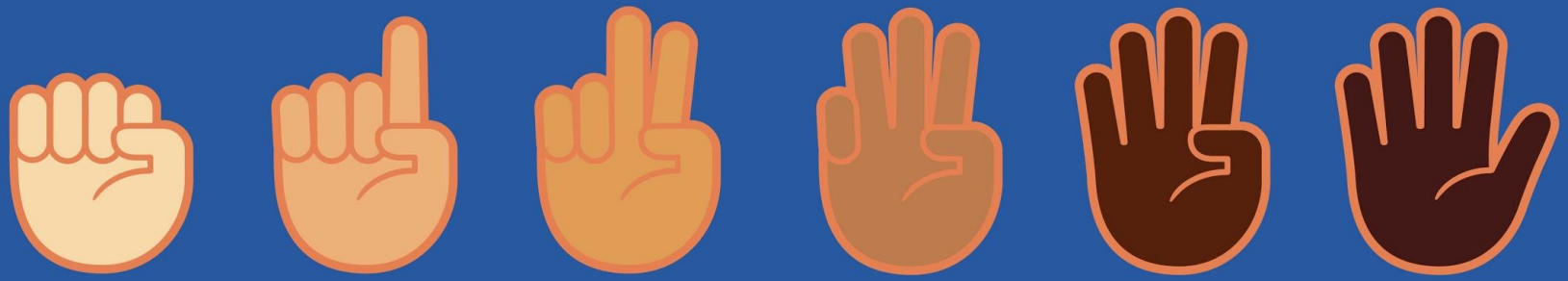




Instructor Demonstration:

Alerts, Prompts, and Confirms

Suggested Time:
5 minutes



Fist to five

Let’s

Activity: Prompt Sushi

1. Open instructions in Canvas
2. Open up the `prompt-sushi` file
3. Write your code.
4. When you are done, share with your partner!

Suggested Time:
15 minutes

Let’s review

Alert vs. Console.log

Our `alert()` and `console.log()` functions provide a form of output, but they can be rather inconvenient - they're either **too loud** or **too quiet**, and *neither of them stick around very long*!

```
// Alert
alert("We definitely rock!");
```

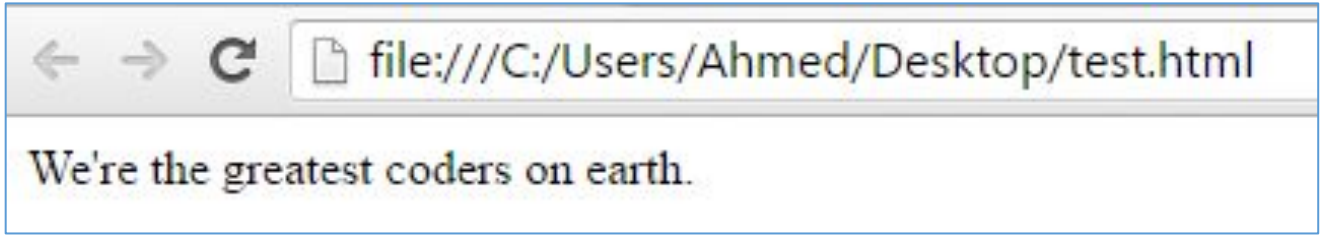
This page says:

We definitely rock!

Writing to HTML: `document.write()`

This function takes in a string and then prints it to the browser, even allowing developers to add HTML tags into the string so that the text may be formatted in some way.

```
<script type="text/javascript">
    document.write("We're the greatest coders on earth.");
</script>
```



The `document.write()` function on its own will always replace the content on the webpage.

Overwriting HTML with `document.write()`

`document.write()` is often used for testing purposes or site design -- ***not*** website deployment.

Questions?

Conditionals and Control Flow

State refers to the data in our application at a specific point in time.


After exercising, we might check on our own “state” in order to make a decision: hydrate, stretch, or take a shower

Conditions check on the status of our code and help the program determine what to do

Conditional Statements: IF

The **`if`** keyword allows us to specify a block of code that should be run ***if*** a specified condition is met first.

```
var playerHealth = 100;

// check to see if the value of the playerHealth variable is greater
than 0
if (playerHealth > 0) {
  console.log("Your player is still alive!");
 }
```

Conditional Statements: IF

The condition appears between the parentheses

```
var playerHealth = 100;

// check to see if the value of the playerHealth variable is greater
than 0
if (playerHealth > 0) {
  console.log("Your player is still alive!");
 }
```

Conditional Statements: IF

If the condition is considered "true” then the code between the curly braces executes.

```
var playerHealth = 100;

// check to see if the value of the playerHealth variable is greater
than 0
if (playerHealth > 0) {
  console.log("Your player is still alive!");
 }
```

Conditional Statements: IF

If the condition is ***not*** considered "true," then the code between the curly braces will ***not*** execute.

```
var playerHealth = 0

// check to see if the value of the playerHealth variable is greater
than 0
if (playerHealth > 0) {
  console.log("Your player is still alive!");
 }
```

Conditional Statements: IF

The **`if`** keyword allows us to specify a block of code that should be run ***if*** a specified condition is met first.

```
var playerHealth = 0;

// check to see if the value of the playerHealth variable is greater
than 0
if (playerHealth > 0) {
  console.log("Your player is still alive!");
 }
```

Conditional Statements: IF, ELSE

We use **`if, else`** to specify an *alternative* block of code to be executed, **if** our first condition is **not met**.

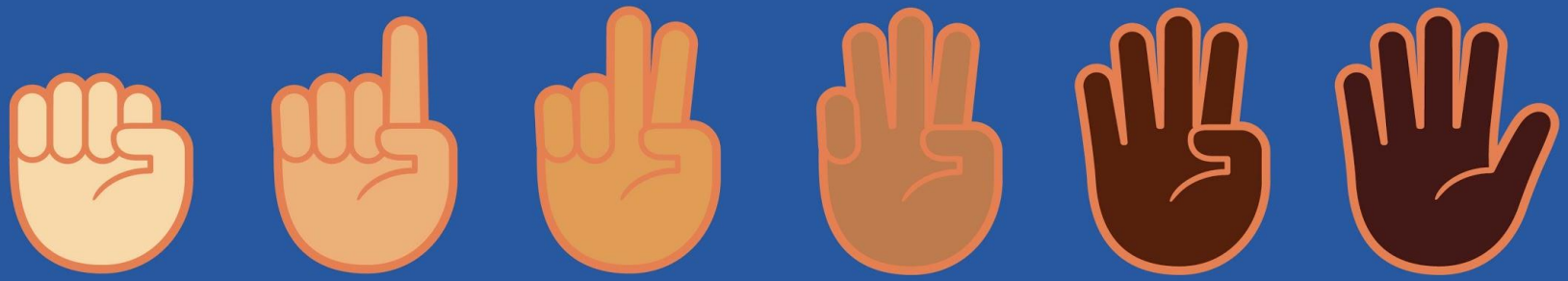
```
var playerHealth = 0;

if (playerHealth > 0) {
  console.log("Your player is still alive!");
 }

else {
  console.log("Uh-oh, your player is dead!”);
 }
```

Conditional logic allows our code to take a certain path depending on the conditions that we define.

Instructor Demonstration:

Conditional Demo

Suggested Time:
5 minutes



Fist to five

Let’s

Activity: Steak Conditionals

1. Open instructions in Canvas
2. Modify our takeout order html page.
3. When you are done, share with a partner!

Suggested Time:
10 minutes

Let’s review

Questions?

Arrays

Single Variables vs *Arrays*

So far, we've defined variables with single data values; however, **arrays** let us assign *multiple* values to the same variable!

```
var car = "Volvo";

var cars = ["Saab", "Volvo", "BMW"];
// arrays let us store additional data within our variables
```

Arrays let us store multiple values within a single variable (at the same time).

Accessing Data From An Array

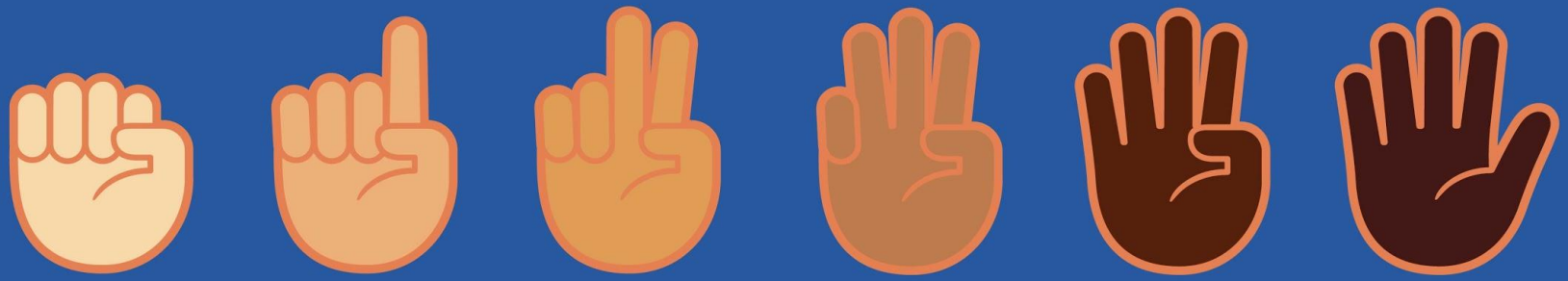
Array data can be accessed with an **index**, but here's a quirky thing: the count starts at *zero*!

```
var cars = ["Saab", "Volvo", "BMW"];
console.log(cars[2])

// This will return our third entry
```

Instructor Demonstration:

Arrays

Suggested Time:
5 minutes



Fist to five

Let’s

Activity: Animals Array

1. Pair with a partner
2. Review files and instructions in Canvas
3. Add comments to your code!

Suggested Time:
10 minutes

Let’s review

Questions?

More Practice!

Let’s

Activity: JavaScript Bands

1. Get into pairs or small groups
2. Open instructions
3. Be prepared to share your work!

Hint: you may need to do a bit of research to complete this one!

Suggested Time:
10 minutes

Let’s review

Questions?

Time to Recap

Learning Goals

Our objectives for today's session:

01 Define basic programming concepts in JavaScript, including Variables, Arrays, Conditionals, Loops

02 Practice applying variables, logging, and if/else statements

03 Explain the concept of browser events

04 Create example HTML buttons that run JS code when clicked.

Reflection

What was your favorite part of today's session?

What was the most interesting thing we covered today?

What do you still have questions about?

Sneak Preview

Tomorrow we'll be practicing our JavaScript by creating a game that can run in our web browser! Afterward, you'll be able to show these off and **share them in your portfolio!**

Questions?

The End