FUNCTIONS

LEARNING OBJECTIVES

- Practice using the console to debug code and find errors.
- Describe why functions are useful
- Describe how parameters relate to functions
- Given a function and a set of arguments, predict the output of a function
- Define and call a function
- Compare global and local scope

REVIEW

JAVASCRIPT — UPDATING THE VALUE OF A VARIABLE

Declaring a variable:

```
var champion = "Sarah";
```

Update the value of the variable:

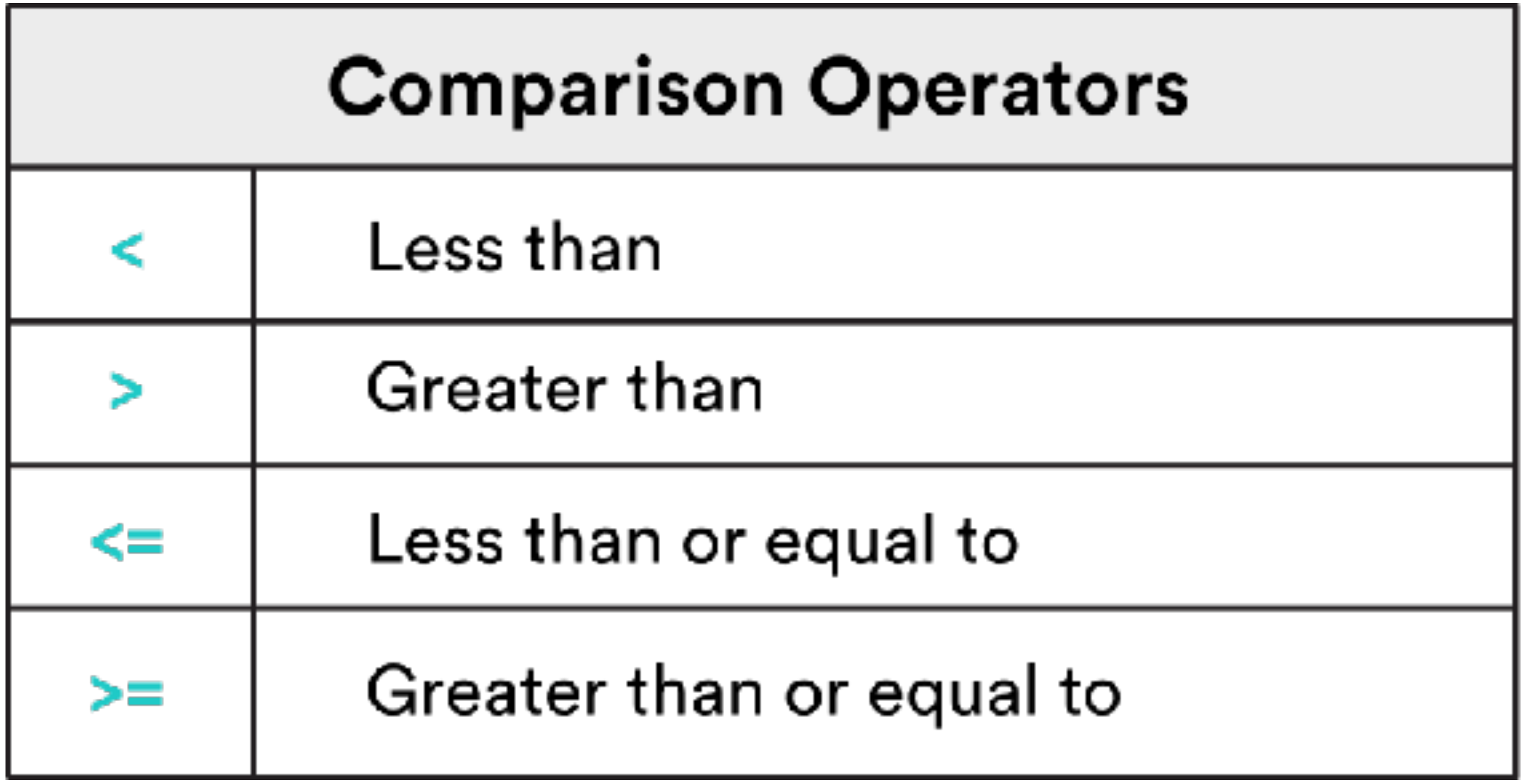
```
champion = "Madeline";
```

STRING CONCATENATION

- To take two strings (or a combination of strings and variables) and stick them together, use the **+** operator.
- This is called string concatenation.

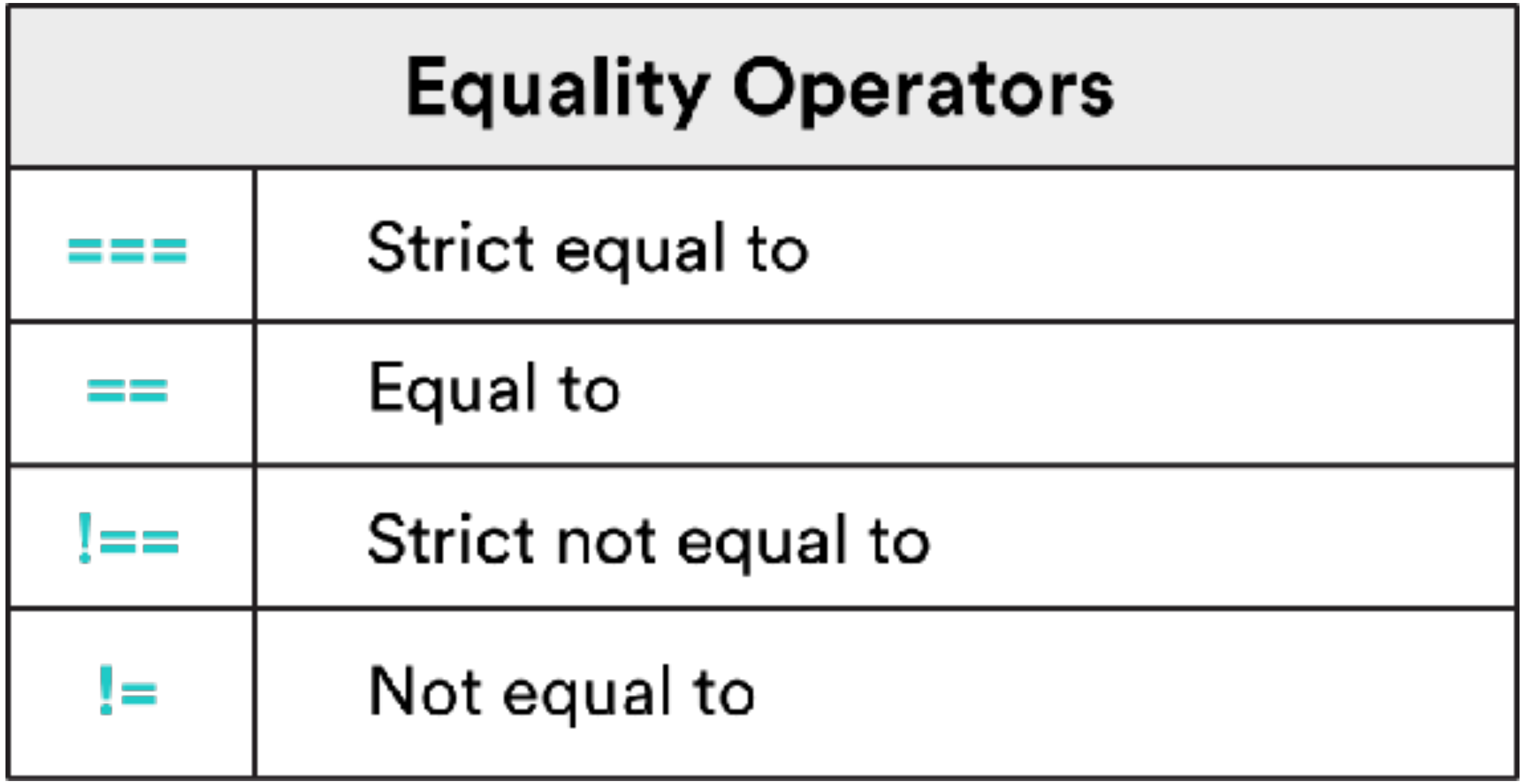
```
var name = "Suzie Smith";
var greeting = "Hello " + name;
// greeting will be: "Hello Suzie Smith"
```

COMPARISON OPERATORS



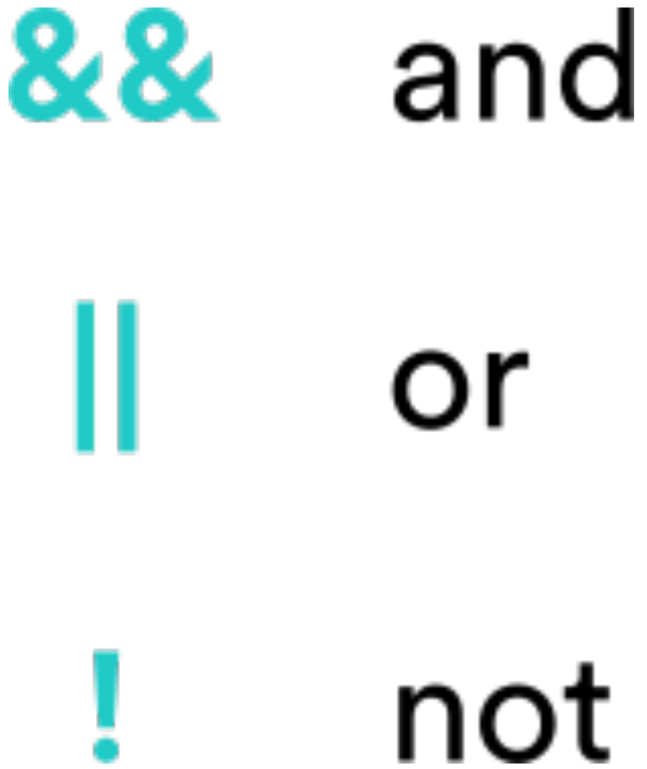
Comparison Operators	
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to

COMPARISON OPERATORS



Equality Operators	
===	Strict equal to
==	Equal to
!==	Strict not equal to
!=	Not equal to

MULTIPLE CONDITIONS



&&	and
||	or
!	not

IF STATEMENTS

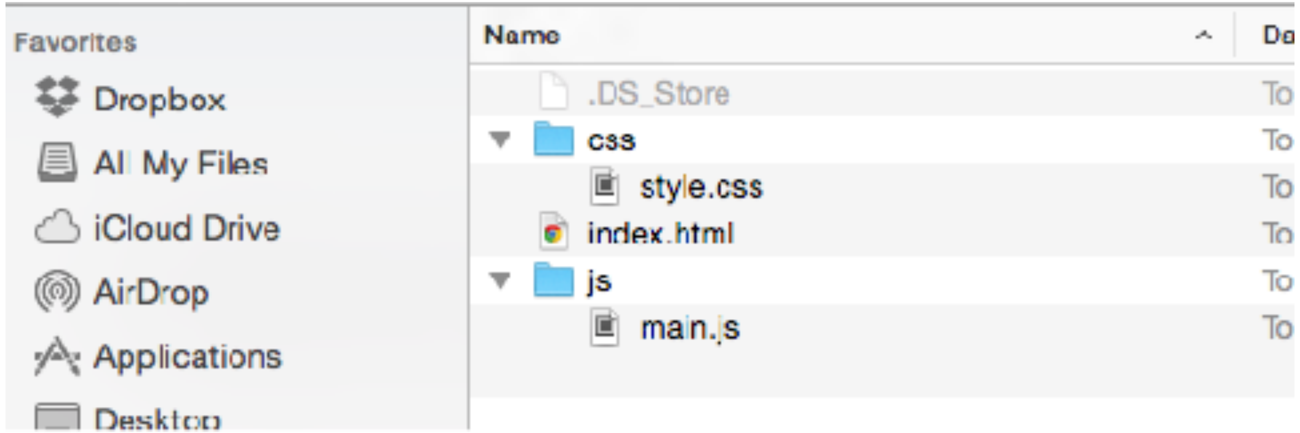
```
if (age > 65) {
    console.log("Senior Discount Applied");

} else if (age < 18) {
    console.log("Student Discount Applied");

} else {
      console.log("Sorry, you don't qualify for a discount");
}
```

KEEP IT ON THE UP AND UP!

- It is considered best practice to keep Javascript files organized in one folder.
- Usually people name this folder scripts, js, or javascript.

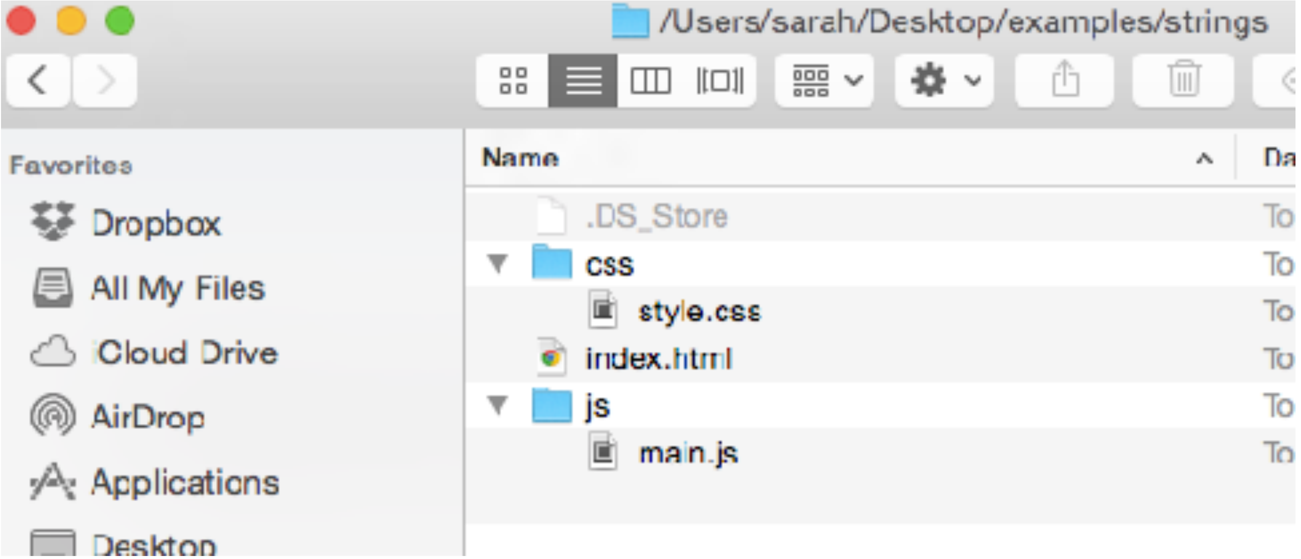


Remember - use an underscore or dash between words in folder names instead of a space. And try to avoid characters/symbols in file names (really_cool_page.html or really-cool-page.html).

ADDING JAVASCRIPT TO YOUR PROJECT

KEEP IT ON THE UP AND UP!

- It is considered best practice to keep Javascript files organized in one folder.
- Usually people name this folder scripts, js, or javascript.



Remember - use an underscore or dash between words in folder names instead of a space. And try to avoid characters/symbols in file names (really_cool_page.html or really-cool-page.html).

ADD A JAVASCRIPT FILE TO YOUR PROJECT

1. Create a Javascript file. This process will be similar to creating an HTML or CSS file, but this time the file should have a .js extension (example: main.js)
2. Link to the Javascript file from your HTML page using a script element. Add this right before the closing body tag.

```
<body>
  <!-- HTML content here -->
  <script src="js/main.js"></script>
</body>
```

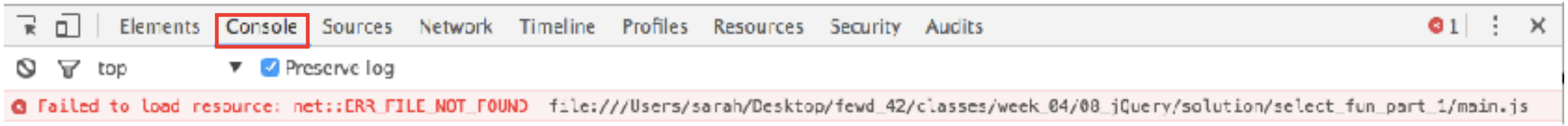
ORDER IS IMPORTANT!!!!

MAKE SURE YOUR JS IS HOOKED UP PROPERLY

- Method 1: Add an alert to the top of your JS file. When you open the page in your browser, an alert will pop up if your JavaScript file is properly hooked up.

```
alert('Hello from JS!);
```

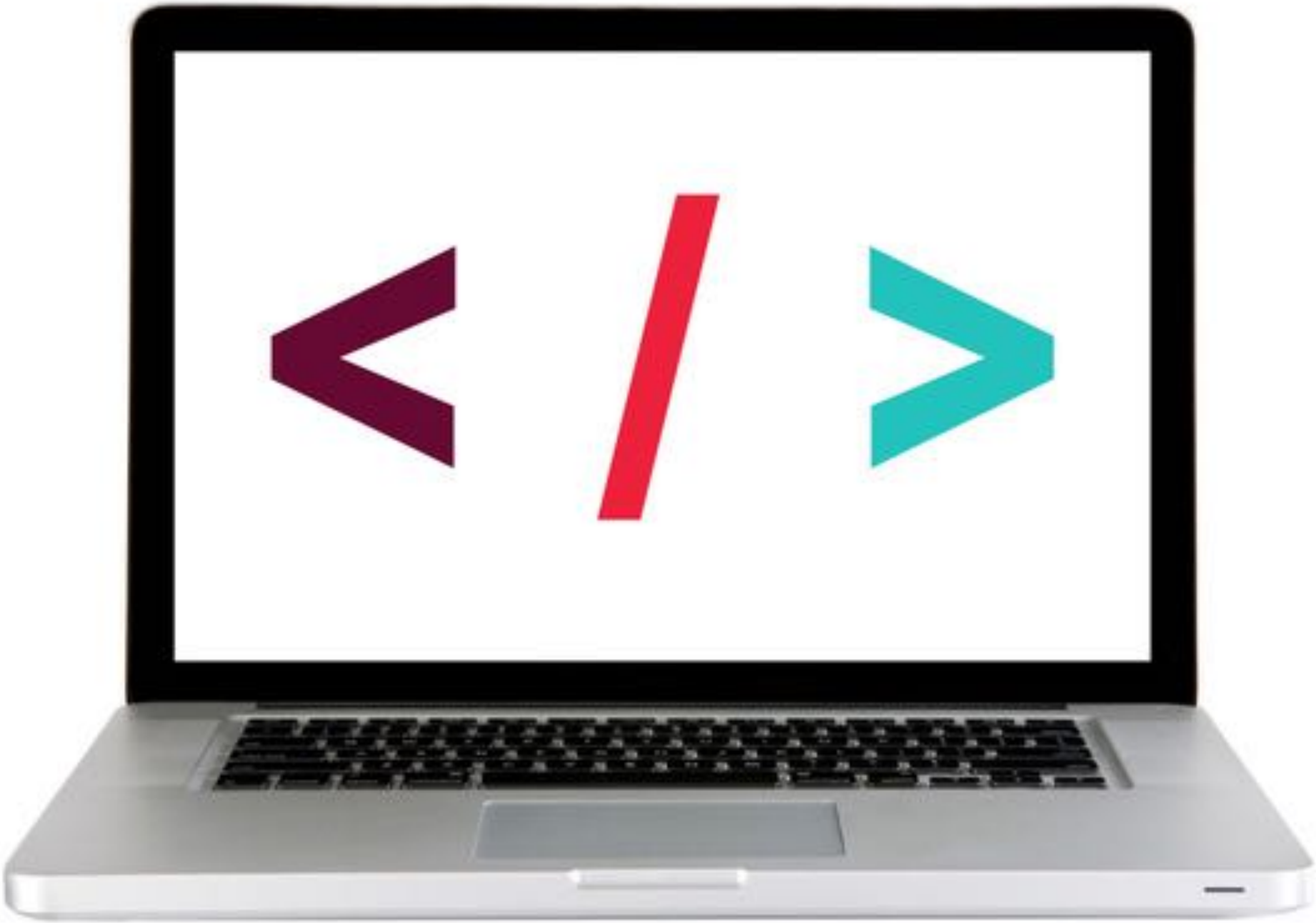
- Method 2 (preferred): Open the page in Chrome. Go to view > developer > developer tools. Click on the console tab and make sure there are no errors.



```
Failed to load resource: net::ERR_FILE_NOT_FOUND  file:///Users/sarah/Desktop/fewd_42/classes/week_04/08_jQuery/solution/select_fun_part_1/main.js
```

This error means the file can't be found. Check your url in your script tag. Make sure the file exists.

LET'S TAKE A CLOSER LOOK



DEBUGGING — LEVEL 1

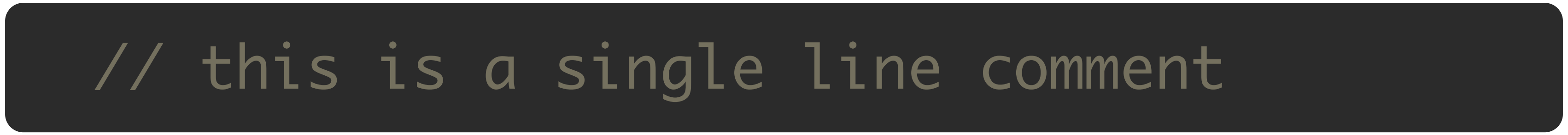
JS SYNTAX

Syntax: Spelling and grammar rules of a programming language.

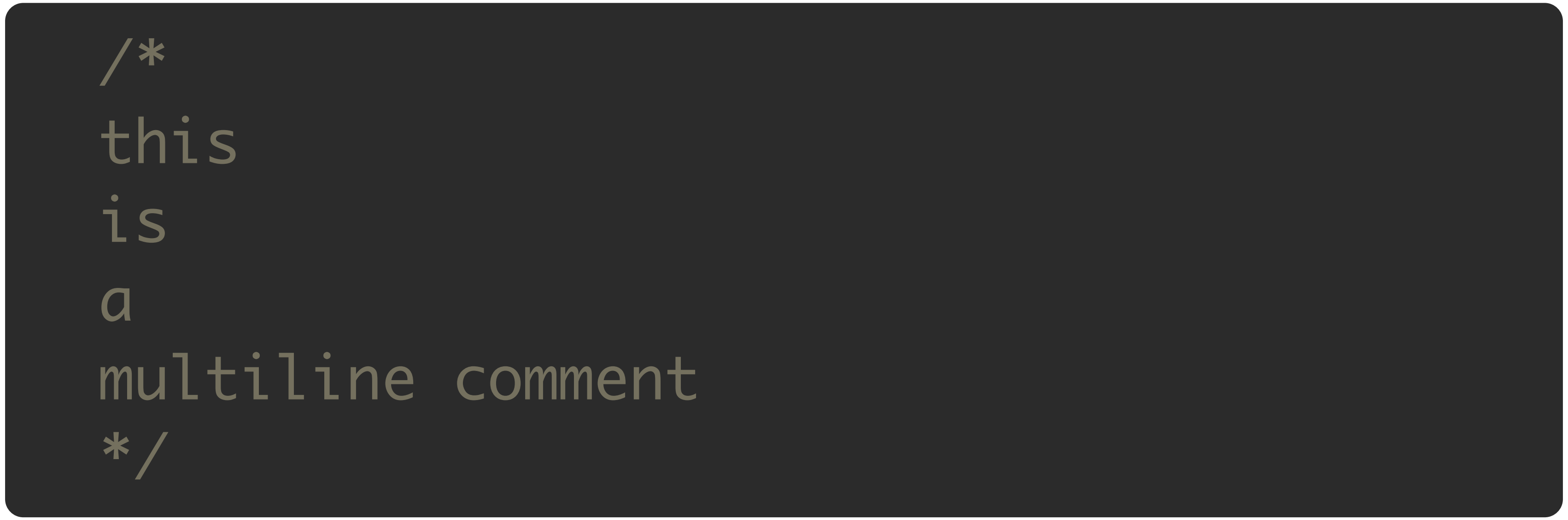
; { } ( ) “ ”

Like any language, there are formal rules around how to write Javascript. This is the syntax.

COMMENTS



```
// this is a single line comment
```



```
/*
this
is
a
multiline comment
*/
```

shortcut: 1) Highlight what you want to comment 2) command + /

YOUR RESPONSIBILITIES

Don't feel like you have to sit down and memorize the syntax!

Focus on understanding
the key concepts

Be resourceful. Google
is your best friend

Practice, practice,
practice

DEBUGGING

To access debugging console:

PC: CTRL+SHIFT+J

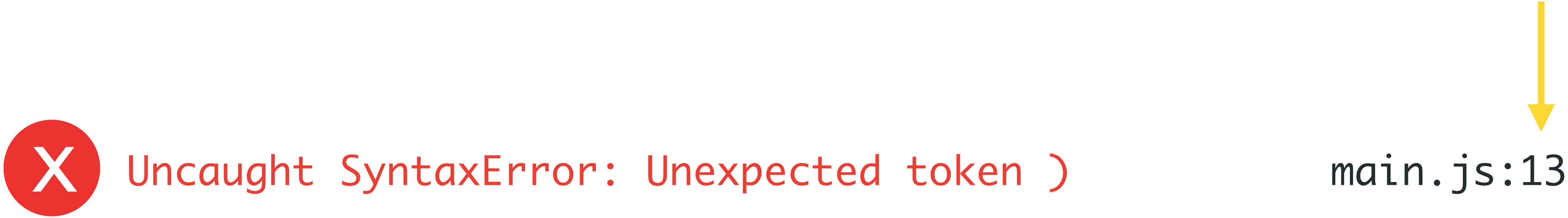
Mac: COMMAND+OPTION+J

Click the error

DEBUGGING — LEVEL 1

Check for errors in console

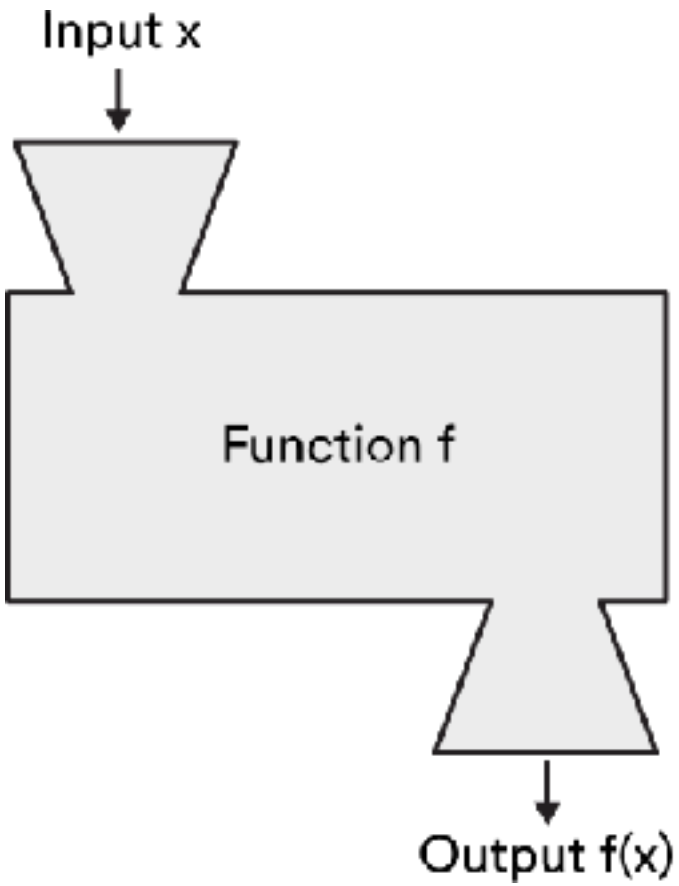
- The line number to the right may not be correct but is a good place to start.
- The error message may not pinpoint the error, but it might give you a clue



```
Uncaught SyntaxError: Unexpected token )                main.js:13
```

FUNCTIONS

FUNCTIONS



x	f(x)
-1	-2
0	0
1	2
2	4
3	6

EXERCISE — FUNCTIONS INTRO

EXERCIS
E

KEY OBJECTIVE

- Practice reading code and guessing what's happening.

TYPE OF EXERCISE

- Groups of 3 - 4

LOCATION

- starter code > functions_intro

EXECUTION

8 min

1. Walk through the questions at the top of main.js and discuss.
2. Walk through the questions in part 2 and discuss.

FUNCTIONS

FUNCTIONS

GROUP STEPS

Allow us to group a series of statements together to perform a specific task

REUSABLE

We can use the same function multiple times

STORE STEPS

Not always executed when a page loads. Provide us with a way to 'store' the steps needed to achieve a task.

DRY — DON'T REPEAT YOURSELF

DRY — WHY SHOULD WE AVOID REPETITION?

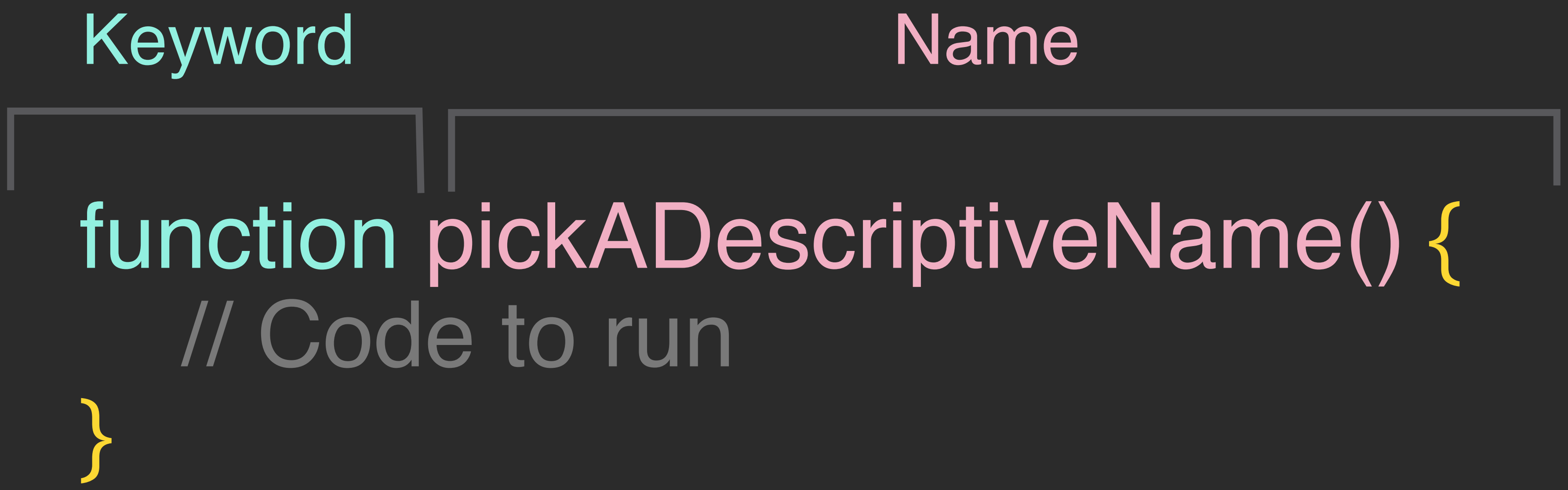
Why should we avoid repetition?

1. **Performance:** Having repeated code will lead to longer scripts. Longer scripts take up more memory and will take more time to load, which can make a website seem slow.
2. **Maintainability:** Imagine that we have the same line of code repeated 10 times in our program. If we ever want to change that functionality, we would have to search for all 10 places where that code is repeated and make 10 individual changes.

FUNCTIONS

SYNTAX

SYNTAX — DECLARING A FUNCTION



Keyword

Name

```
function pickADescriptiveName() {
    // Code to run
}
```

SYNTAX — CALLING A FUNCTION

```
function pickADescriptiveName () {
    // Code to run
}
```

To run the function, we need to *call* it. We can do so like this:

```
pickADescriptiveName();
```

Function name + parentheses

FUNCTIONS — LET'S PLAY SOME MUSIC!

```
function playSong () {
    var song = "This Must be the Place";
    var artist = "Talking Heads";

    $('#nowPlaying').html('Now playing: ' + song + ' by ' + artist);
}
```

FUNCTIONS — LET'S PLAY SOME MUSIC!

```
function playSong () {
    var song = "This Must be the Place";
    var artist = "Talking Heads";

    $('#nowPlaying').html('Now playing: ' + song + ' by ' + artist);
}

playSong();
```

EXERCISE — WRITING FUNCTIONS

EXERCIS
E

KEY OBJECTIVE

- Practice defining and executing functions

TYPE OF EXERCISE

- Individual/paired

LOCATION

- starter code > functions_practice (part 1)

EXECUTION

4 min

1. Follow the instructions under Part 1

SYNTAX — DECLARING A FUNCTION (WITH PARAMETERS)

Parameters

```
function multiply (param1, param2) {
   console.log(param1 * param2);
}
```

We can use these parameters like variables from within our function

Arguments

```
multiply(350, 140)
```

FUNCTIONS — LET'S PLAY SOME MUSIC!

```
function playSong (song, artist) {
    $('#nowPlaying').html('Now playing: ' + song + ' by ' + artist);
}
```

FUNCTIONS — LET'S PLAY SOME MUSIC!

```
function playSong (song, artist) {
    $('#nowPlaying').html('Now playing: ' + song + ' by ' + artist);
}

playSong("This Must be the Place", "Talking Heads");
```

```
function sayHello (name) {
    $('.greeting').html("Hello " + name);
}
```

What will be the html that gets added to .greeting we call the function using these arguments:

```
sayHello("Drake");
```

```
function sayHello (name) {
    $('.greeting').html("Hello " + name);
}
```

How about now?

```
sayHello("Sally");
```

EXERCISE — READING FUNCTIONS

EXERCIS
E

KEY OBJECTIVE

- Given a function and a set of arguments, predict the output of a function

TYPE OF EXERCISE

- Individual

LOCATION

- starter code > functions_practice (part 2)

EXECUTION

8 min

1. Complete the steps under Part 2
2. Optional: Also complete the bonus step

EXERCISE — FUNCTIONS

EXERCIS
E

KEY OBJECTIVE

- Describe why functions are useful
- Describe how parameters relate to functions

TYPE OF EXERCISE

- Turn and Talk

EXECUTION

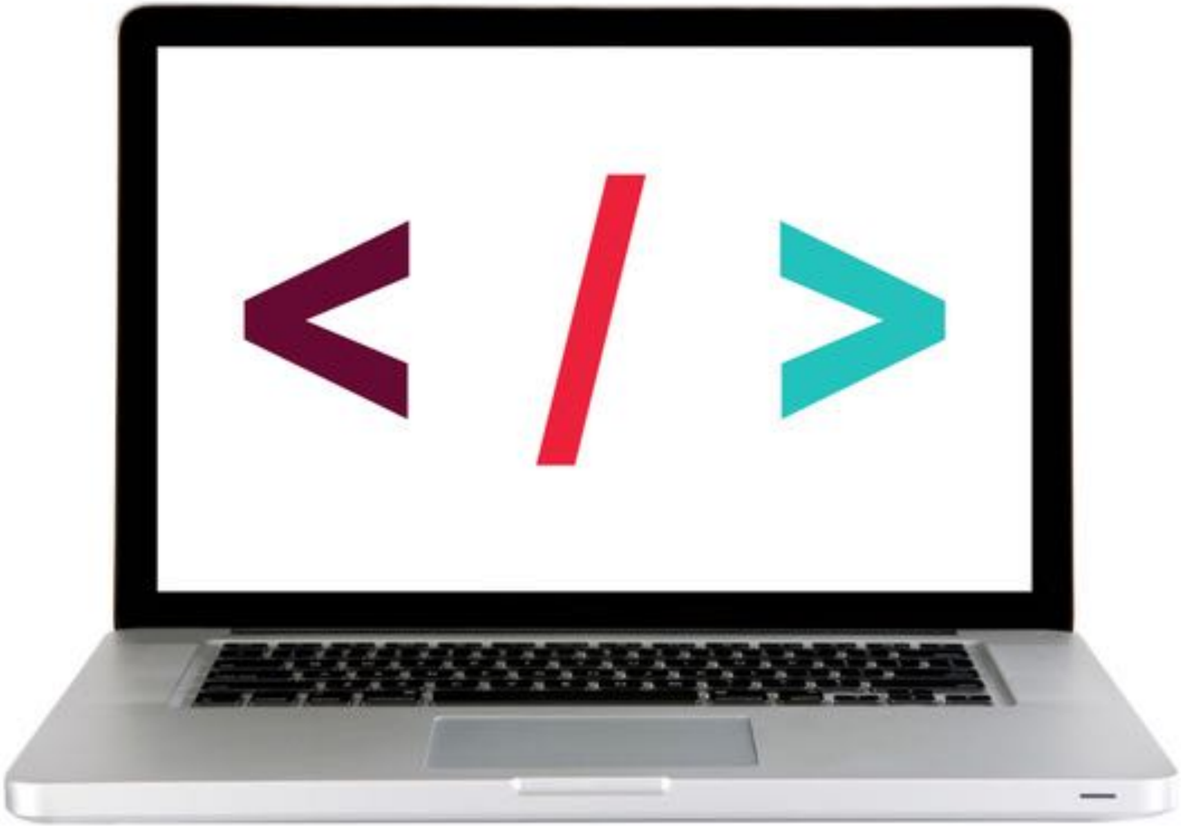
1 min

1. Summarize why we would use functions in our programs. What purpose do they serve?
2. How do parameters relate to functions?

SCOPE

LET'S TAKE A CLOSER LOOK

LET'S TAKE A CLOSER LOOK



View example in Codepen

VARIABLE SCOPE

LOCAL VARIABLES

- A local variable is a variable that is declared inside a function.
- It can only be used in that function, and cannot be accessed outside of that function

GLOBAL VARIABLES

- A global variable is a variable that is declared outside of a function.
- It can be used anywhere in the script.

EXERCISE — READING FUNCTIONS

EXERCIS
E

KEY OBJECTIVE

- Given a function and a set of arguments, predict the output of a function

TYPE OF EXERCISE

- Groups of 2 - 3

LOCATION

- starter code > functions (part 4)

EXECUTION

8 min

1. Follow the instructions under Part 4 in main.js

LAB TIME!

EXERCISE — ROCK, PAPER, SCISSORS

EXERCIS
E

KEY OBJECTIVE

- Practice writing pseudo code to create a rock, paper, scissors game.

TYPE OF EXERCISE

- Groups of 4-5

LOCATION

- Slack > Text Snippet

EXECUTION

10 min 1. Write pseudo code for a rock, paper scissors game.

EXERCISE — ROCK, PAPER, SCISSORS

EXERCIS
E

KEY OBJECTIVE

- Practice variables, conditionals, and JS functions by creating a rock, paper, scissors game.

TYPE OF EXERCISE

- Individual / Groups

LOCATION

- starter code > Rock, Paper, Scissors

EXECUTION

1. Write your code under each line of pseudo code in the main.js file.
2. Bonus: Best of 3 version
3. Challenge: Generate a random computer play each time.

LAB

LEARNING OBJECTIVES

- Practice using the console to debug code and find errors.
- Describe why functions are useful
- Describe how parameters relate to functions
- Given a function and a set of arguments, predict the output of a function
- Define and call a function
- Compare global and local scope

EXIT TICKETS