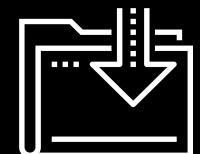


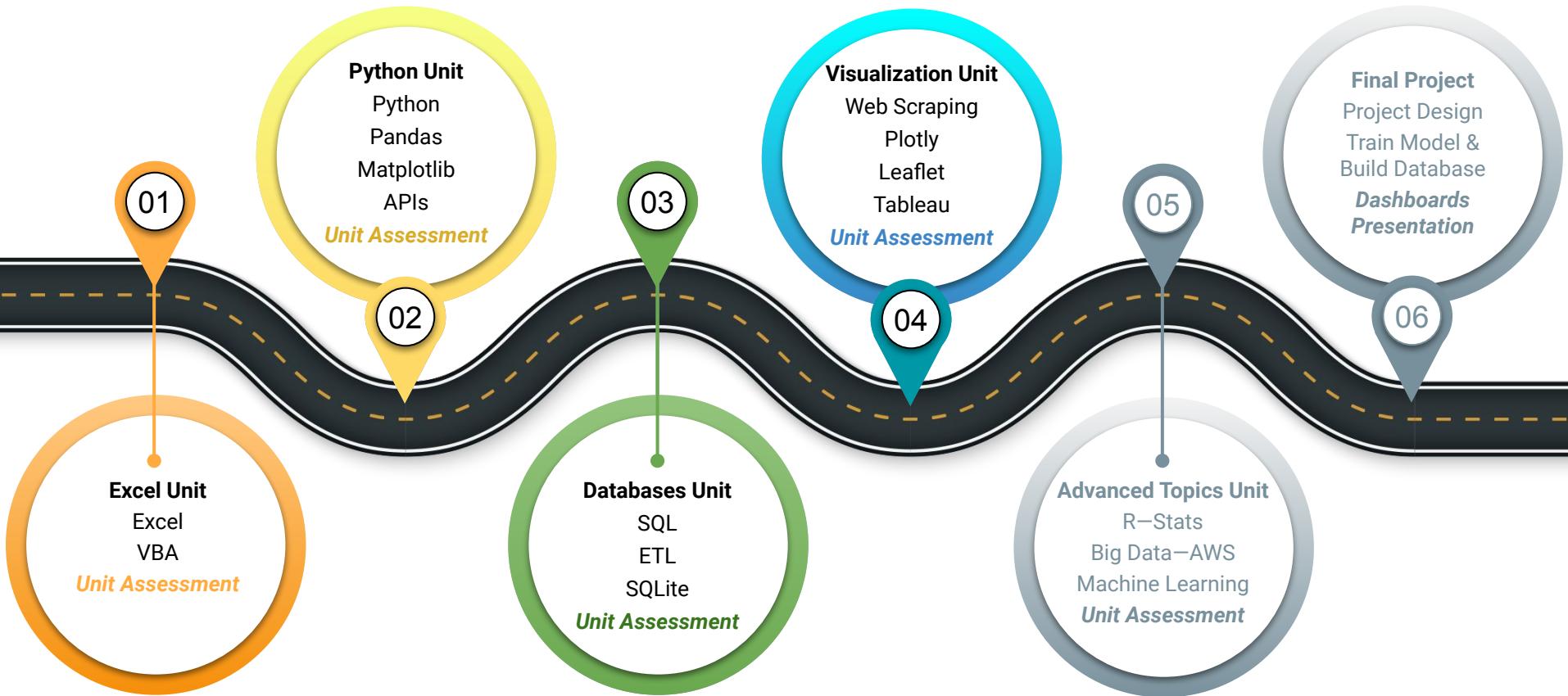
# JavaScript Fundamentals

Data Boot Camp  
Lesson 11.1



# The Big Picture

---





## Quick Tip for Success:

Consider teaming up with some of your classmates for virtual study groups as we get into this content!

Module 11

# This Week: JavaScript Fundamentals

# This Week: JavaScript Fundamentals

---

By the end of this week, you'll know how to:



Explain the strengths and weaknesses of JavaScript "standard" and JavaScript version ES6+



Describe JavaScript syntax and ideal use cases



Build and deploy JavaScript functions, including built-in functions



Convert JavaScript functions to arrow functions



Build and deploy `forEach` (JavaScript `for` loop)



Create, populate, and dynamically filter a table using JavaScript and HTML



## This Week's Challenge

Using the skills learned throughout the week, create new search parameters and new JavaScript functions for UFO data to create an updated and dynamic webpage.



## Career Connection

How will you use this module's content in your career?

Module 11

# How to Succeed This Week



## Quick Tip for Success:

Don't forget to lean on documentation whenever you get stuck on syntax. You'll be making robust visualizations in no time!

Module 11

# Today's Agenda

# Today's Agenda

---

By completing today's activities, you'll learn the following skills:

01

JavaScript variables and objects

02

JavaScript functions and arrow functions

03

Repetition and conditional statements

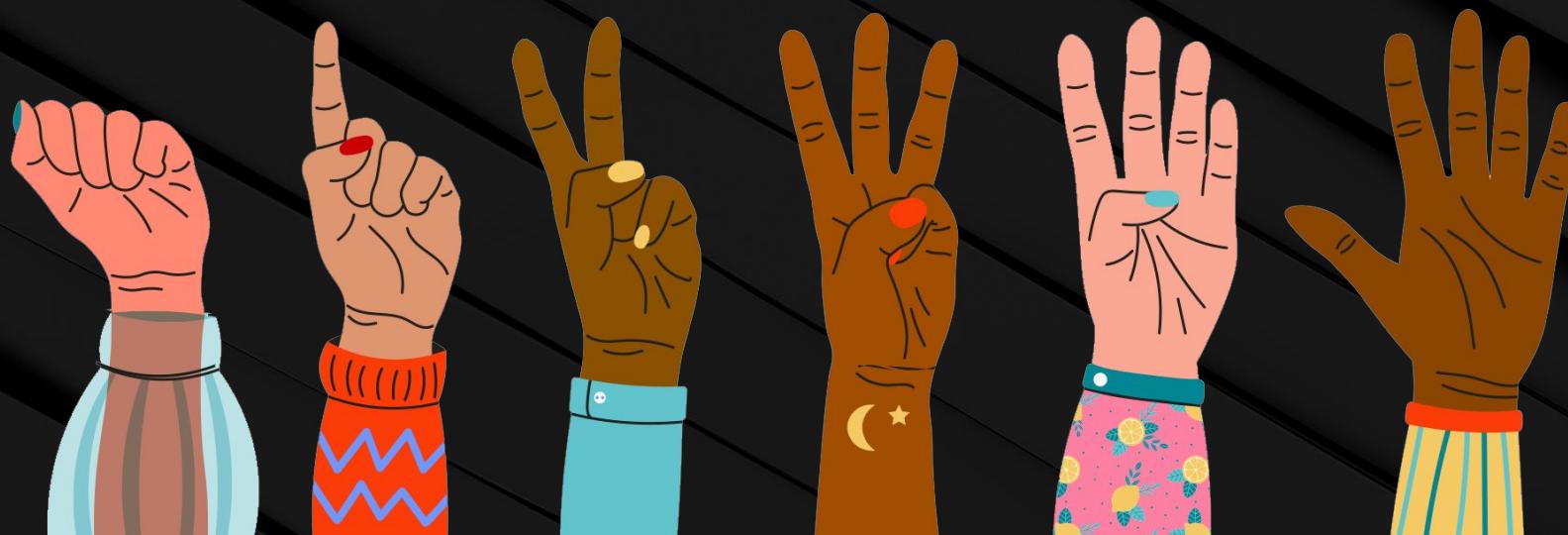


Make sure you've downloaded  
any relevant class files!

## FIST TO FIVE:

---

How comfortable do you feel with this topic?



# Introduction to JavaScript

# JavaScript

---

## JavaScript fundamentals:



Arrays



Conditionals



Loops



Functions

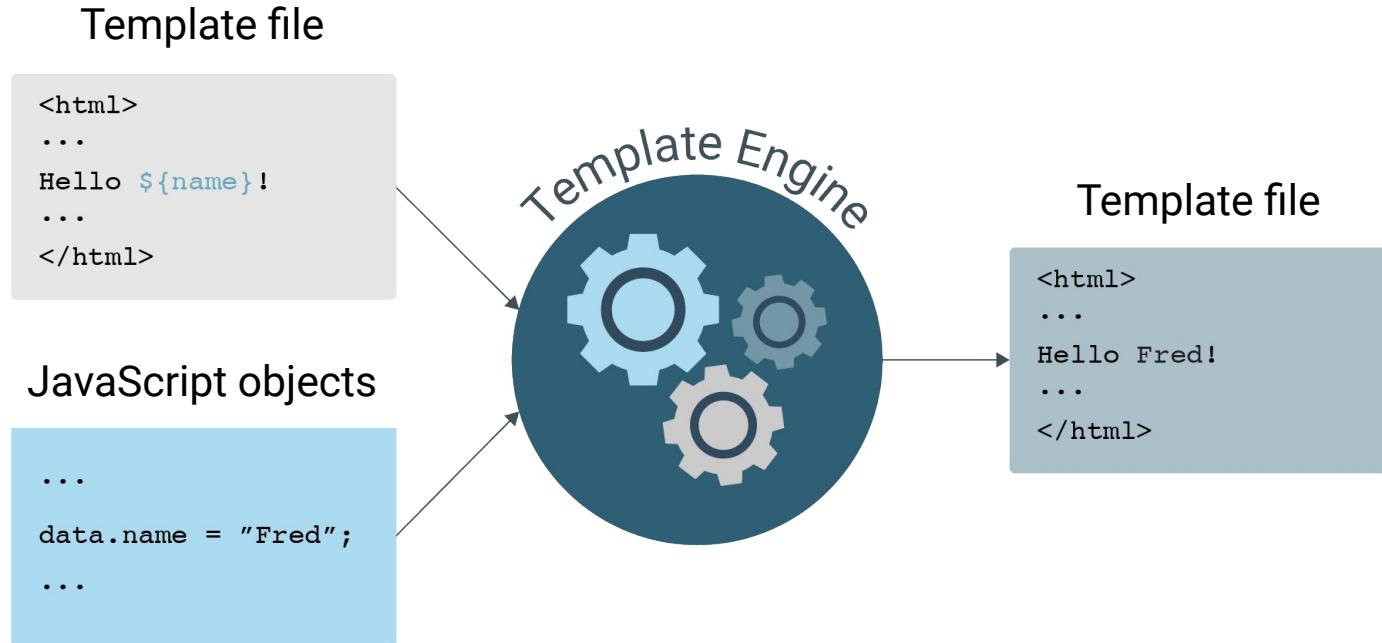


Objects

# JavaScript

---

Just as important as the ability to crunch numbers is the ability to convey information to a wider audience, and JavaScript is the language of the web.





JavaScript makes it possible  
to create interactive web  
pages and visualizations.

# JavaScript

---

Talk to us if you need extra help!

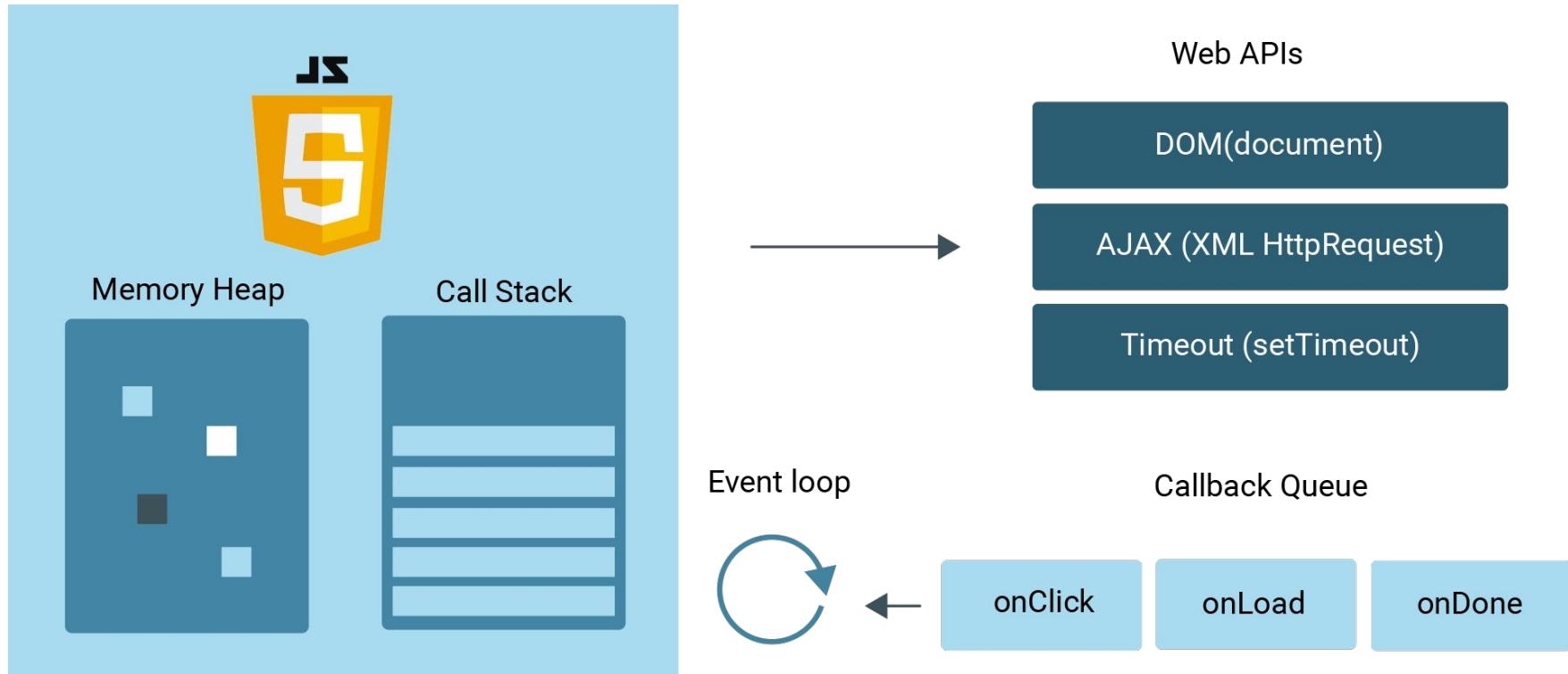




JavaScript shares many similarities with Python, but there are also a number of differences, including typical usage and syntax.

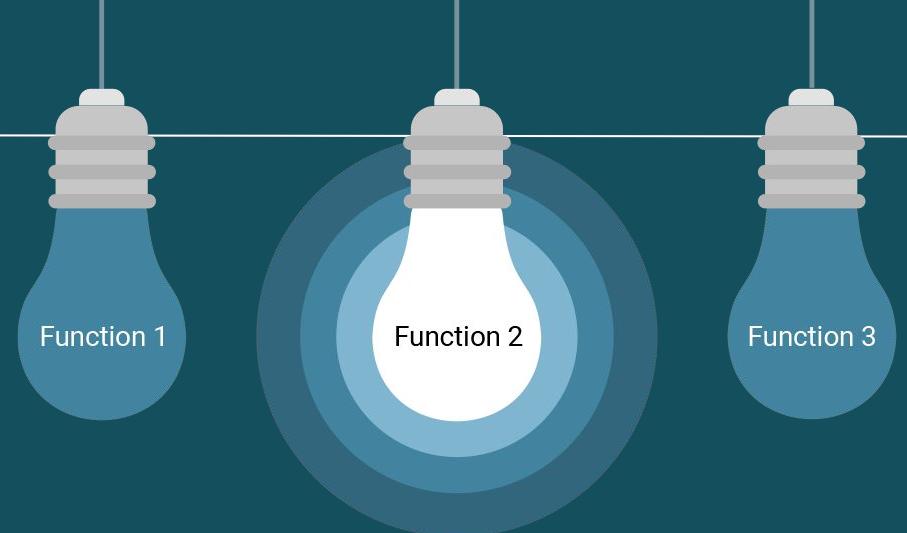
# JavaScript

JavaScript is often used to place API calls to cloud data and services.



# JavaScript

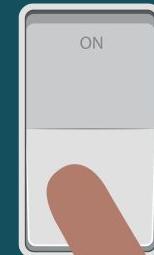
JavaScript also enables websites to send and receive data from a server, to respond to a user's actions on the page, and to dynamically modify HTML elements.



Function  
call 1



Function  
call 2



Function  
call 3





With JavaScript, it is possible  
to build interactive sites that  
do not require use of the  
command line interface

# JavaScript

JavaScript is everywhere ...

Even machine learning has been made available for the web browser

The screenshot shows the TensorFlow.js homepage. At the top, there's a navigation bar with links for Install, Learn (which is currently selected), API, Resources, Community, More, a search bar, Language, GitHub, and Sign in. Below the navigation, it says "For JavaScript". Underneath that, there are tabs for Overview (selected), Tutorials, Guide, Models, Demos, and API. The main content area features a large orange graphic on the left and a central text block. The text reads: "TensorFlow.js is a library for machine learning in JavaScript". Below this, it says: "Develop ML models in JavaScript, and use ML directly in the browser or in Node.js.". At the bottom, there are three buttons: "See tutorials", "See models", and "See demos". To the right of the text, there's a stylized illustration of a computer monitor displaying a neural network diagram with nodes and connections.

TensorFlow

Install Learn API Resources Community More

Search

Language GitHub Sign in

For JavaScript

Overview Tutorials Guide Models Demos API

TensorFlow.js is a library for machine learning in JavaScript

Develop ML models in JavaScript, and use ML directly in the browser or in Node.js.

See tutorials See models See demos

# JavaScript

JavaScript is a very marketable and in-demand skill for any data position.

**InfoWorld** UNITED STATES ▾ SOFTWARE DEVELOPMENT CLOUD COMPUTING MACHINE LEARNING ANALYTICS INSIDER ➔ 

Home > Software Development > JavaScript

## JavaScript is the most in-demand IT skill

Report based on hiring trends cites JavaScript, SQL, and Java skills as the most sought after in the past year



By **Paul Krill**  
Editor at Large, InfoWorld | JAN 21, 2020 2:29 PM PST

# How to Learn JavaScript

# Your Brain on JavaScript



# JavaScript

---

## General Tips:

<b>Review Immediately</b>	We'll be building upon these concepts quickly. The firmer your grasp is now, the better off you'll be.
<b>Redo the Class Exercises</b>	Don't just reread! Actually spend the time to redo them from scratch on your own.
<b>Get Help</b>	Come to office hours. Ask conceptual questions. Ask specific questions. Just keep asking questions!
<b>Don't Be Afraid</b>	You will get this. It will take time, but you will get this. Just keep at it. Patience will pay off.



## Instructor Demonstration

---

### Running JavaScript

# Running JavaScript

JavaScript is the language of the web, so its code is commonly found in HTML.

The code is placed between a pair of `script` tags.

The `console.log()` function prints out a message to a web browser's built-in console.



```
<!DOCTYPE html>
<html lang="en">

<head>
  <title></title>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1">
</head>

<body>

  <h1>Open the Chrome Inspector Console!</h1>

  <script type="text/javascript">
    console.log("My script is stored within the HTML!")
  </script>

</body>

</html>
```



# Running JavaScript

src="app.js"

```
<!DOCTYPE html>
<html lang="en">

  <head>
    <title></title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
  </head>

  <body>

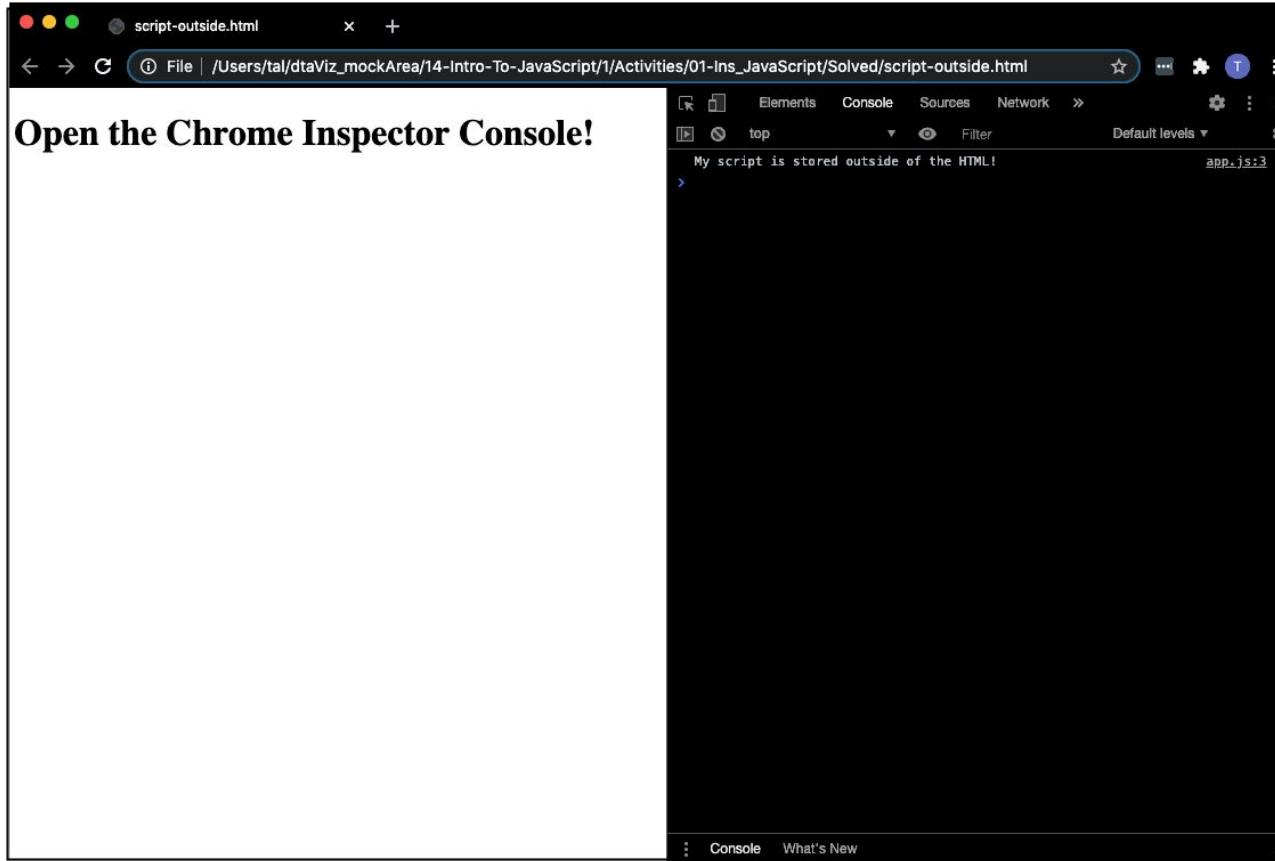
    <h1>Open the Chrome Inspector Console!</h1>
    <!-- src contains the path to the file -->
    <script src="app.js"></script>
  </body>

</html>
```

app.js

```
// This code will run when linked to in HTML
console.log("My script is stored outside of the HTML!")
```

# Running JavaScript

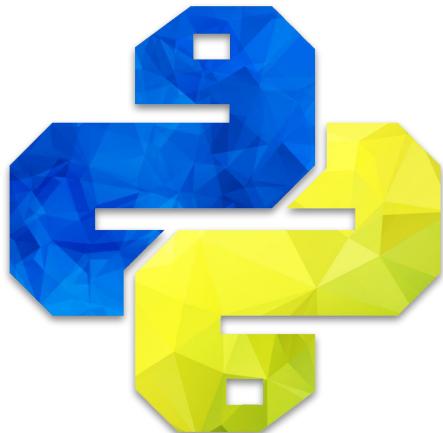


# From Python to JavaScript

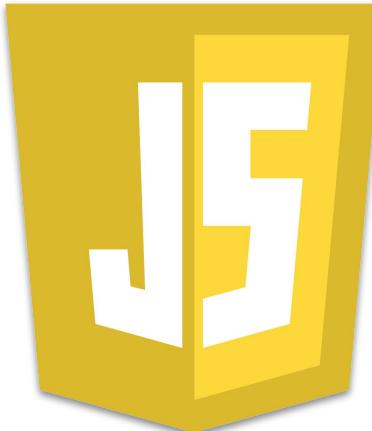
# From Python to JavaScript

---

Python and JavaScript are logically and syntactically similar.

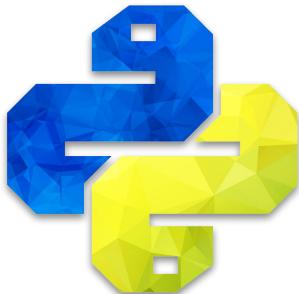


JS



# Variables

---



```
<variable Name> = <Value>
```

```
name = "Mad Max"
```



```
var <variable Name> = <Value>;
```

```
var name = "Mad Max";
```

# Booleans

---



```
<variable Name> = True
```

**satisfied = True**

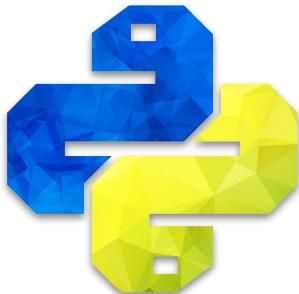


```
var <variable Name> = true;
```

**var satisfied = true;**

# String template + Note:

---



```
# Python f-string
```

```
print(f"Hello, {name}!")
```



```
// JavaScript template literal
```

```
console.log(`Hello ${name}!`);
```

# String Format Converted into Numeric Format

---



```
# int() function
```

```
weekly_wage = hourly_wage *  
int(weekly_hours)
```



```
// parseInt() function
```

```
var weeklyWage = hourlyWage *  
parseInt(weeklyHours);
```



# From Python to JavaScript

Suggested Time:

---

15 minutes

# Conditionals with Comparison Operators (== vs ===)

---



```
# Use whitespace with indentation
```

```
if x == 1:  
    print("x is equal to 1")
```

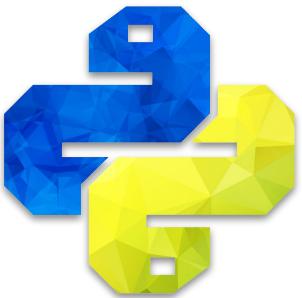


```
// Use curly brackets
```

```
if (x === 1) {  
    console.log("x is equal to 1");  
}
```

# Conditional with Logical Operators: and

---



```
# and
```

```
if x == 1 and y == 10:  
    print("Both values returned  
true")
```

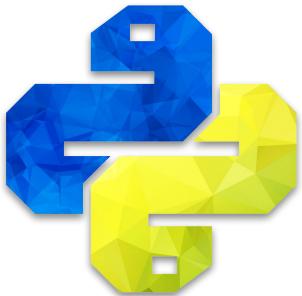


```
// &&
```

```
if (x === 1 && y === 10) {  
    console.log("Both values  
returned true");  
}
```

# Conditional with Logical Operators: or

---



```
# or
```

```
if x < 45 or y < 5:  
    print("One or the other  
statements were true")
```



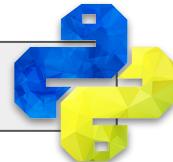
```
// ||
```

```
if (x < 45 || y < 5) {  
    console.log("One or the other  
statements were true");  
}
```

# Nested Conditionals

---

```
# if...elif...else
```



```
if x < 10:
    if y < 5:
        print("x is less than 10 and
y is less than 5")
    elif y == 5:
        print("x is less than 10 and
y is equal to 5")
    else:
        print("x is less than 10 and
y is greater than 5")
```

```
// if...else if...else
```



```
if (y < 5) {
    console.log("x is less than 10 and
y is less than 5");
}
else if (y === 5) {
    console.log("x is less than 10 and
y is equal to 5");
}
else {
    console.log("x is less than 10 and
y is greater than 5");
}
```

# Questions?





## Instructor Demonstration

---

# Variables, Arrays, and Objects



# JavaScript Arrays

Suggested Time:

---

15 Minutes



JavaScript arrays are  
similar to Python lists.

# JavaScript Arrays

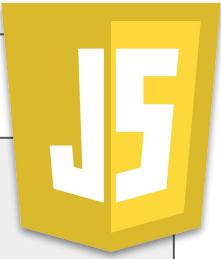
JS

JavaScript array

```
var lettersArray =  
["a", "b", "c", "d"];
```

index

```
var firstLetter =  
lettersArray[0];  
  
var secondLetter =  
lettersArray[1];  
  
a  
b
```



# JavaScript Arrays

JS

.push()

```
lettersArray.push("e");

var letterArray = ["a", "b" , "c", "d",
"e"];
```

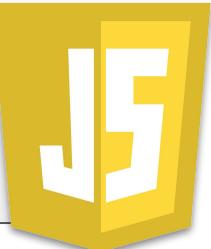


.slice()

```
var slicedArray1 =
lettersArray.slice(1);
// Return the first three items of an
array

var slicedArray2 =
lettersArray.slice(0, 3);
// Return the second and third items of
an array

var sliced Array 3 =
lettersArray.slice(1, 3);
```



# JavaScript Arrays

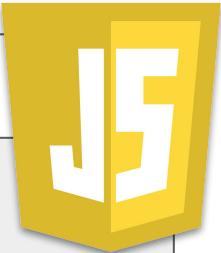
JS

.join()

```
var joinedArray =  
lettersArray.join(", ");
```

.split()

```
var soundArray =  
soundOfMusic.split(" ");
```



# Questions?





# Instructor Demonstration

---

for Loops

# for Loops in JavaScript

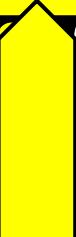
The diagram illustrates the structure of a for loop. It features three colored arrows pointing downwards at the start of the loop's initialization, condition, and update phases respectively. The first arrow is yellow and labeled 'Start'. The second arrow is cyan and labeled 'End condition'. The third arrow is green and labeled 'Increment'. Below these labels, the corresponding code segments are highlighted: 'i = 0' (yellow), 'i < 10' (cyan), and 'i++' (green). The entire loop structure is contained within a black rectangular box.

```
for (var i = 0; i < 10; i++) {
    console.log("Iteration #", i);
}
```

# for Loops on an Array in JavaScript

```
var students = ["Johnny", "Bodhi", "Pappas"];
```

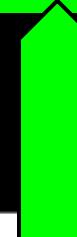
```
for (var j = 0; j < students.length; j++) {  
    console.log(students[j]);  
}
```



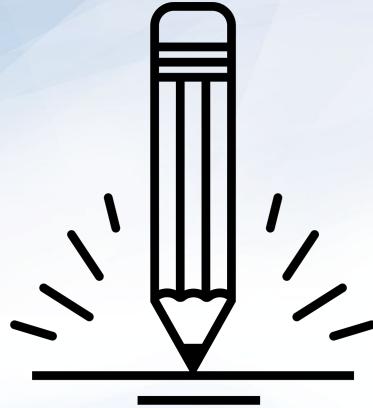
Start



End condition



Increment



## Activity: Movie Scores

In this activity, you will use conditionals and loops to iterate through an array of movie scores and sort scores into new arrays by their values.

Suggested Time:  
15 Minutes



# Instructions: Movie Scores

---

- Given a list of movie scores, determine how many good, ok, and bad movies there were.
- Create a `for` loop to go through the `movieScore` list.
- Add scores over 7 to the `goodMovies` array.
- Add scores between 5 and 7 to the `okMovies` array.
- Add the rest of the scores to the the `badMovies` array.
- Also, calculate the average rating for all of the movies.
- Finally, print out how many good, ok, and bad movies there were, and what the overall total score was.

## Hints:

- You will need to research how to push elements to an empty array.
- Check your Slack for the [documentation](#) to find the length of the array.



**Let's Review**



## Instructor Demonstration

---

# JavaScript Functions

# Functions

---



def

```
def print_hello():
    print("Hello there!")
```

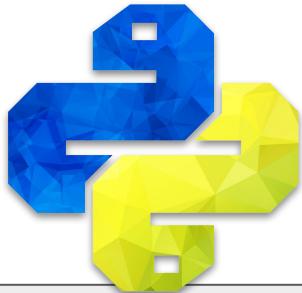


function

```
function printHello()
    console.log("Hello
there!");
}
```

# Functions

---



def

```
print_hello()  
addition(44, 50):
```



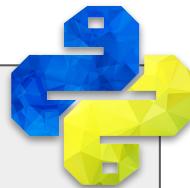
function

```
printHello();  
console.log(addition(44, 50));
```

# Functions

---

```
# Takes in a list and  
loops through
```



```
def list_loop(user_list):  
    for i in user_list:  
        print(i)
```

```
// Accepts a parameter and  
iterates through an array
```



```
function listLoop(userList) {  
    for (var i = 0; i <  
userList.length; i++) {  
        console.log(userList[i]);  
    }  
}  
var friends = ["Sarah", "Greg",  
"Cindy", "Jeff"];  
listLoop(friends);
```

# Functions

---

```
# Uses previously declared  
function
```



```
def double_addition(c, d):  
    total = addition(c, d) * 2  
  
    return total
```

```
// Functions can call  
other functions
```



```
function doubleAddition(c, d) {  
    var total = addition(c, d) * 2;  
  
    return total;  
}  
// Log results of doubleAddition  
function  
console.log(doubleAddition(3, 4));
```

# Functions

---



```
# Python built in function  
for rounding
```

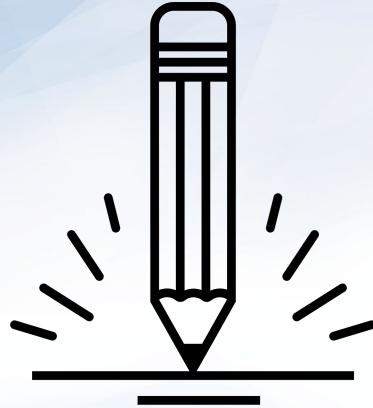
```
long_decimal = 112.34534454  
rounded_decimal =  
round(long_decimal)  
print(rounded_decimal)
```

JS

```
// JavaScript built  
in functions
```



```
var longDecimal = 112.34534454;  
var rounded Decimal =  
Math.floor(longDecimal);  
console.log(rounded Decimal);
```



## Activity: Statistics Functions

In this activity, you will create functions that return statistical values from any given array of data.

Suggested Time:  
15 Minutes



# Instructions: Movie Score Statistics

---

01 Using the movie array from earlier as a starting point, create functions that will return statistical values from any given array of data.

02 Create functions that will find the following:

- Mean
- Variance
- Standard Deviation

03 Each function should `console.log` both the name of the statistic used and its value.  
**For example:** "The Mean is: 33.3".

04 The functions should be able to take any array of numbers and return the statistical value.  
After you have the functions working with movie data set, run them on the following additional data points:  
`monthlyAvgRainFall = [3.03, 2.48, 3.23, 3.15, 4.13, 3.23]`  
`mileRunTimes = [5.06, 4.54, 5.56, 4.40, 7.10]`

## Hints:

Use the JavaScript Math library to handle calculations needing exponents or square roots.

Check your Slack to refresh on how to calculate [variance](#) and [standard deviation](#).



**Let's Review**

# Questions?

