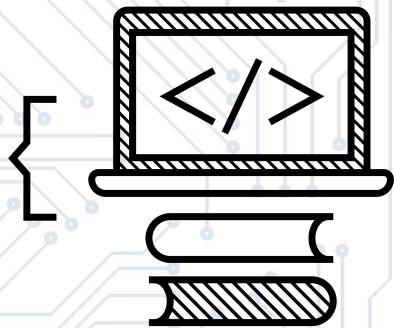


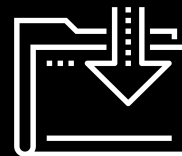
# Preclass Drill:

What is a self-closing tag? Provide 3 examples. Why are these tags self-closing? What differentiates these elements from others?



# JavaScript Juggernauts

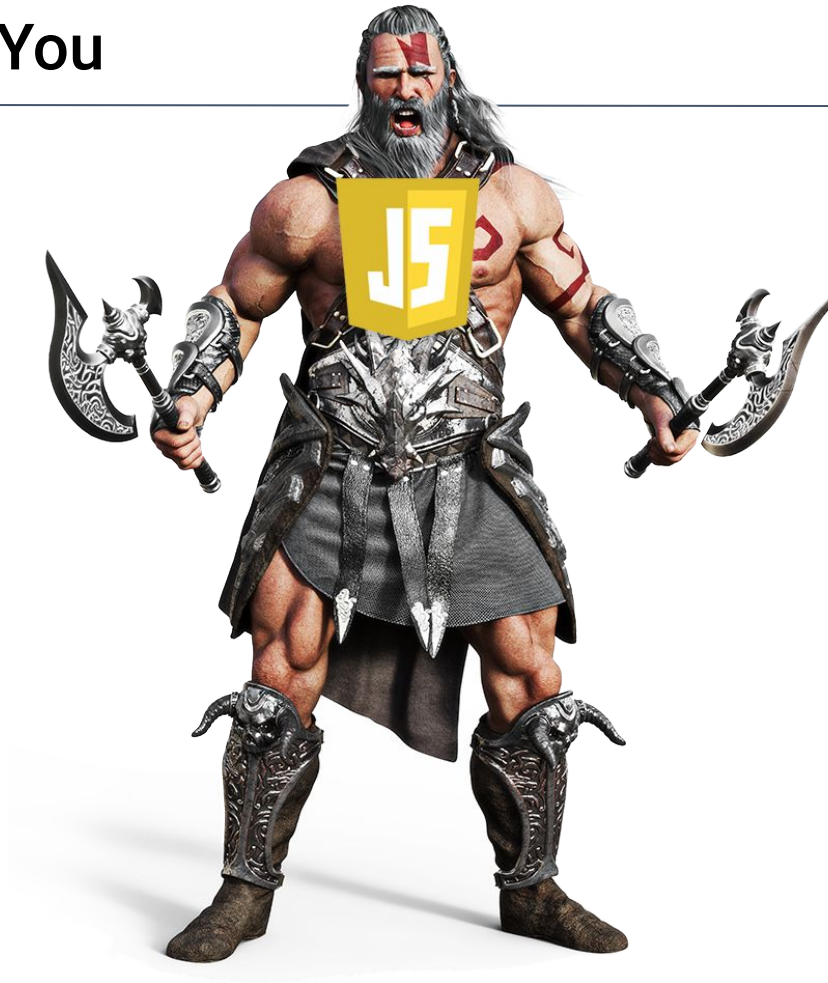
Web Development Boot Camp  
Lesson 3.3



# This Will Soon Be You

---

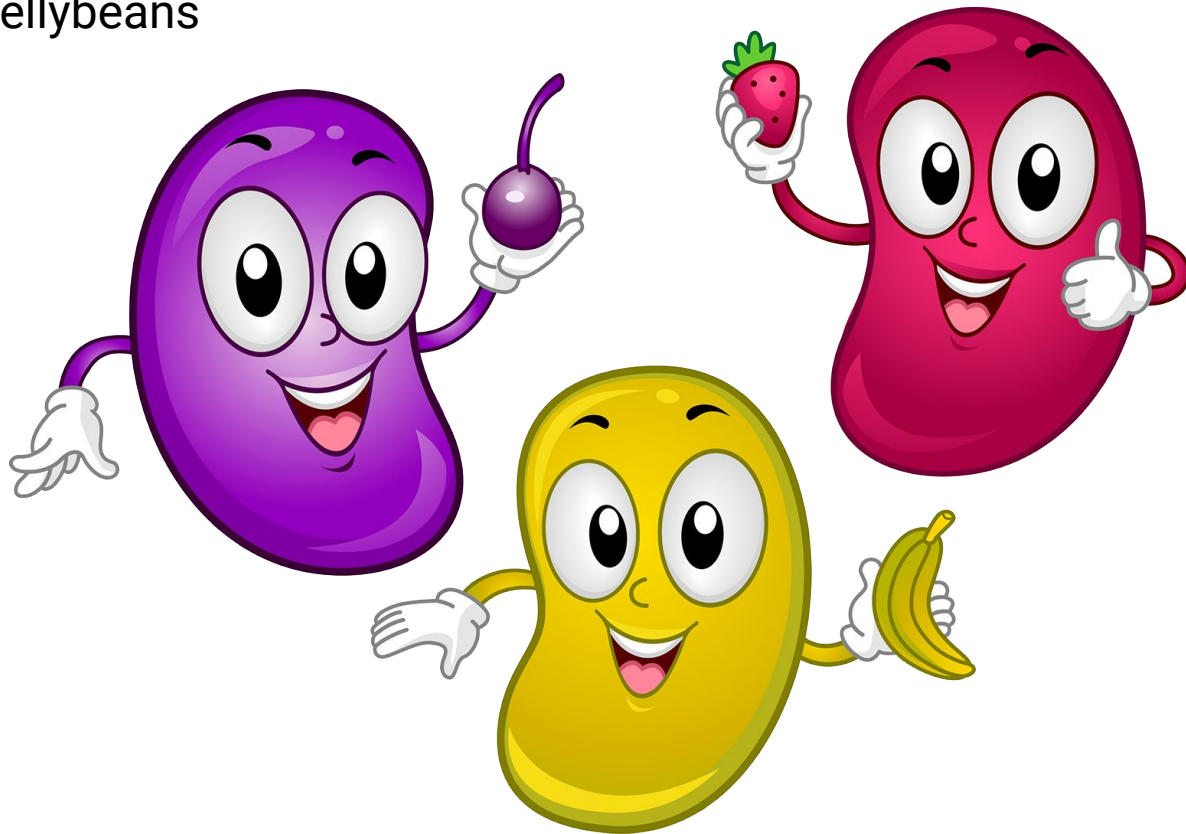
JavaScript Juggernauts



# But Right Now You Feel Like

---

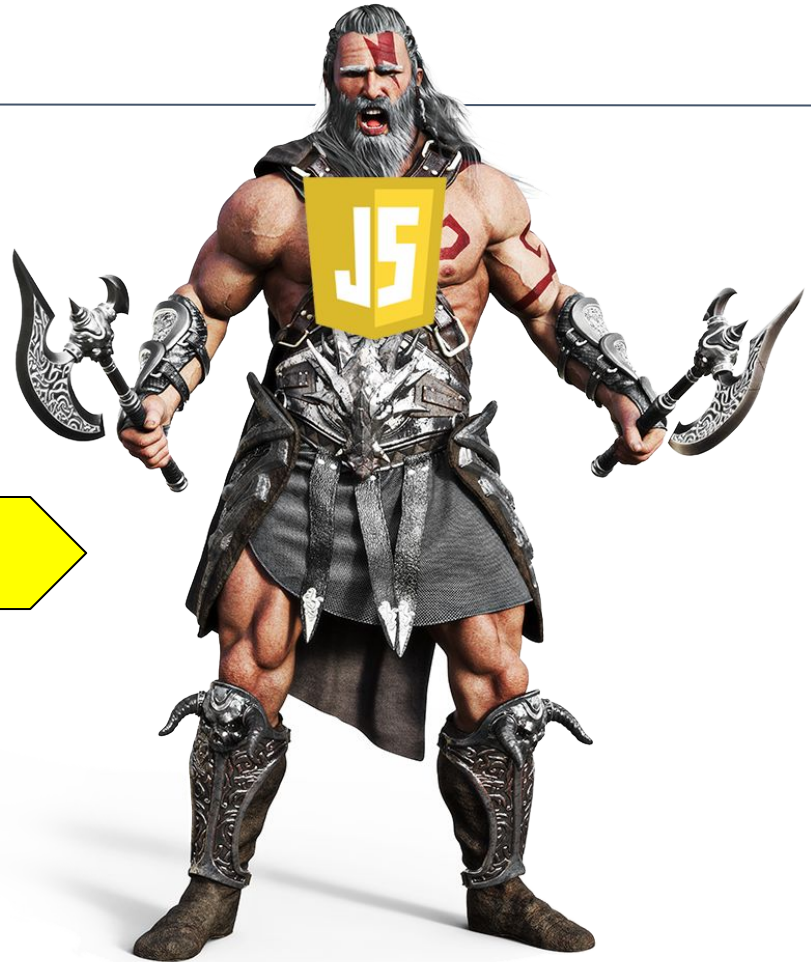
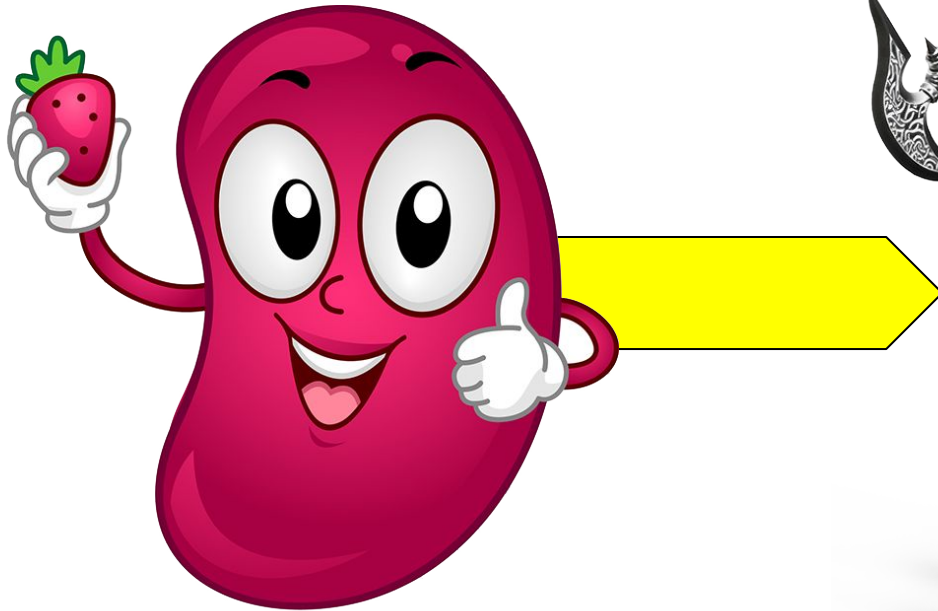
JavaScript Jellybeans



# Transformation to Come!

---

HANG IN THERE!



# Today's Class

# Objectives

---

In today's class, we'll cover:



JavaScript Functions



JavaScript Objects



Building Simple JavaScript Applications



## Partner Activity:

### Loop TV (Array Building)

**Suggested Time:**  
10 minutes





# Partner Activity: Array Building

---

01

Run the program sent to you via Slack.

02

Then, with a partner, fill in the missing comments for each line of code.

03

Make sure both of you can fully explain what each line means.

04

Be prepared to share with the class.

**Suggested Time:** 10 minutes



# JavaScript Functions

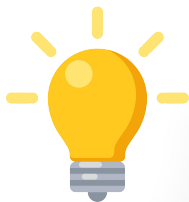


# Instructor Demonstration

## Logging: No Functions

# Mondo Repetitive

Who wants to maintain this?



**Hint:** No one.



```
// For Loop for Brands
for (var i = 0; i < brands.length; i++) {
  console.log(brands[i]);
}
console.log("-----");

// For Loop for Heroes
for (var i = 0; i < heroes.length; i++) {
  console.log(heroes[i]);
}
console.log("-----");

// For Loop for booksOnMyShelf
for (var i = 0; i < booksOnMyShelf.length; i++) {
  console.log(booksOnMyShelf[i]);
}
console.log("-----");

// For Loop for thingsInFrontOfMe
for (var i = 0; i < thingsInFrontOfMe.length; i++) {
  console.log(thingsInFrontOfMe[i]);
}
console.log("-----");

// For Loop for howIFeel
for (var i = 0; i < howIFeel.length; i++) {
  console.log(howIFeel[i]);
}
console.log("-----");
```



# Instructor Demonstration

## Logging: With Functions

# Much Better with Functions!

---

Squeaky clean code. Minimal repetition.

```
// Here we create a "Function" that allows us to "call" (run) the loop for any array we wish.  
// We pass in an array as an "argument".  
function consoleInside(arr) {  
  
    // We then loop through the selected array.  
    for (var i = 0; i < arr.length; i++) {  
  
        // Each time we print the value inside the array.  
        console.log(arr[i]);  
    }  
    console.log("-----");  
}
```



# Partner Activity:

## My First Functions

**Suggested Time:**  
20 minutes



# Partner Activity: My First Functions

---



Working in pairs and using the starter file sent to you via Slack, fill in the missing functions and function calls.



**Note:** Try to finish all four functions if you can, but don't be distressed if you only get 1 or 2. The important thing is that you get at least one function fully done.



**HINT:** Look back to the previous example if you need help.

**Suggested Time:** 20 minutes





# JavaScript Objects



# Instructor Demonstration

## Good Arrays



# Instructor Demonstration

## Joan of Arc (Bad Arrays)

# Associated Data ==/= Arrays

---

Relating two separate arrays is not fun.

```
var joanOfArcInfoParts = ["Real Name", "Grew Up Where", "Known For", "Scars", "Symbolism"];

var joanOfArcInfoValues = ["Jehanne la Pucelle.", "Domremy, a village in northeastern France.",
    "Peasant girl, daughter of a farmer, who rose to become Commander of the French army.",
    "Took an arrow to the shoulder and a crossbow bolt to the thigh while trying to liberate Paris.",
    "Stands for French unity and nationalism."];
```



# Instructor Demonstration

## Gandalf the Grey Objects

# Gandalf: The Object

Gandalf's **properties** and **values** are associated in object form, making it easy to recall specific data.

```
11  var gandalf = {  
12      "real name": "Gandalf",  
13      "age (est)": 11000,  
14      "race": "Maia",  
15      "haveRetirementPlan": true,  
16      "aliases": [  
17          "Greyhame",  
18          "Stormcrow",  
19          "Mithrandir",  
20          "Gandalf the Grey",  
21          "Gandalf the White"  
22      ]  
23  }  
24  
25  // Object properties can be accessed with "bracket notation"  
26  alert("My name is " + gandalf["real name"]);  
27  
28  // Or with "dot notation" if the property has no spaces  
29  if (gandalf.haveRetirementPlan) {  
30  
31      // Or with a variable that matches the name of the property  
32      var ageProperty = "age (est)";  
33      var years = gandalf[ageProperty];  
34      alert("My 401k has been gathering interest for " + years + " years!");  
35  }
```

# Objects Visualized

This is Gandalf. According to code, Gandalf is an **object**.

var gandalf	=	{
-------------	---	---



"real name"	:	"Gandalf"	,
-------------	---	-----------	---

"age (est)"	:	11000	,
-------------	---	-------	---

"race"	:	"Maia"
--------	---	--------

}
---

# Objects Visualized

These are Gandalf's **properties** (like descriptors).

var gandalf	=	{
-------------	---	---



"real name"	:	"Gandalf"	,
-------------	---	-----------	---

"age (est)"	:	11000	,
-------------	---	-------	---

"race"	:	"Maia"
--------	---	--------

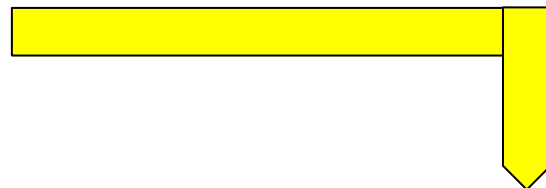
}
---



# Objects Visualized

These are the **values** of Gandalf's properties.

var gandalf	=	{
-------------	---	---



"real name"	:	"Gandalf"	,
-------------	---	-----------	---

"age (est)"	:	11000	,
-------------	---	-------	---

"race"	:	"Maia"
--------	---	--------

}
---

# Objects Visualized

Thus: `gandalf["race"] = "Maia"`

`var gandalf`

`=`

`{`



`"real name"`

`:`

`"Gandalf"`

`,`

`"age (est)"`

`:`

`11000`

`,`

`"race"`

`:`

`"Maia"`

`}`



# Instructor Demonstration

## Gandalf: The Grey Objects (Repeat)



## **Group Activity** (2 people): Basic Objects

**Suggested Time:**  
15 minutes



# Group Activity: Basic Objects

---



With a partner, spend the next few moments studying the code just slacked to you.



Then, write code below each comment to log the relevant information about the provided `car` object.



**Bonus:** If you finish early, create a brand new object of your own. Slack out a snippet of the code to the class when you are done. Be Creative!

Suggested Time: 15 minutes



# Take a Break!

---





# Instructor Demonstration

Run That Car!



## Challenge: Run That Car!

Suggested Time:  
30 minutes





# Challenge: Run That Car!

Using the code from the previous activity as a starting point, create a complete application such that:



Each of the car's methods are assigned to a key.



When the user presses a key, it calls the appropriate function.



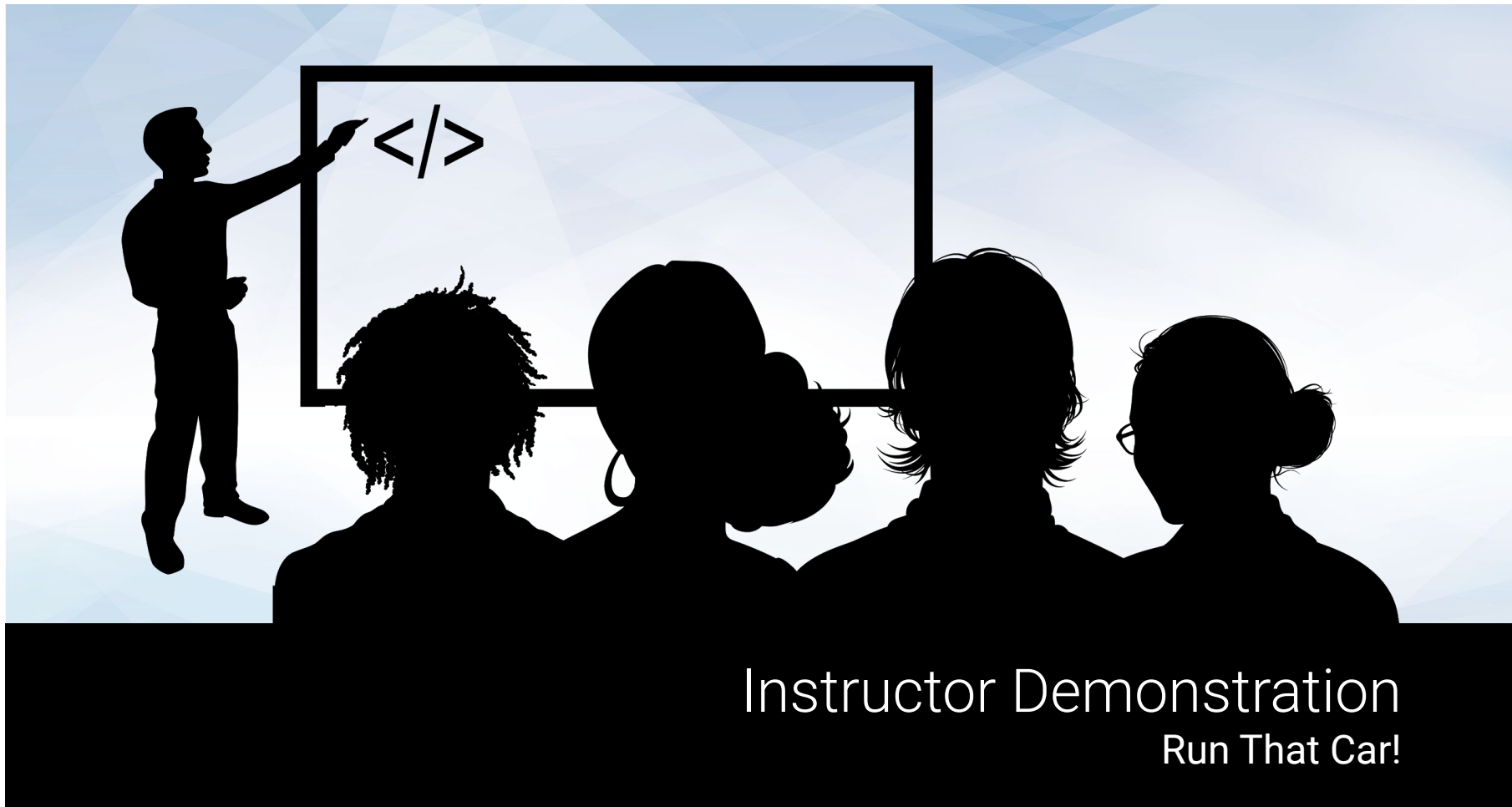
These letters also trigger a global function called `reWriteStats()` that logs the car's make, model, color, mileage, and `isWorking` status to the console.



**HINT:** You will need to use the `document.onkeyup()` function to collect input from the user's keyboard.

Suggested Time: 30 minutes







# **Activity:** Scope & Callbacks

Instructions sent via Slack

**Suggested Time:**  
20 minutes



# **Workbook and Homework**



## **Group Activity:** Question Game

**Suggested Time:**  
20 minutes



# Group Activity: Question Game

Starting from a blank HTML file:



Create an object with 10 questions. The object should be structured like this:  
**q1:** ["QUESTION", "ANSWER"] **q2:** ["QUESTION", "ANSWER"]



Then create code that will ask the user questions, one by one. The user must answer by hitting **t** (for true) or **f** (for false).



Check the user's answer against the correct answer, and provide them with an alert telling them if they are right or wrong.



**Bonus:** Keep track of the user's score.



**Hint:** Don't worry about having DRY code to start with. Just focus on getting working code first.

Suggested Time: 20 minutes





# Questions?