

# **JavaScript Juggernauts**

---

**The Coding Bootcamp**

This will soon be you...

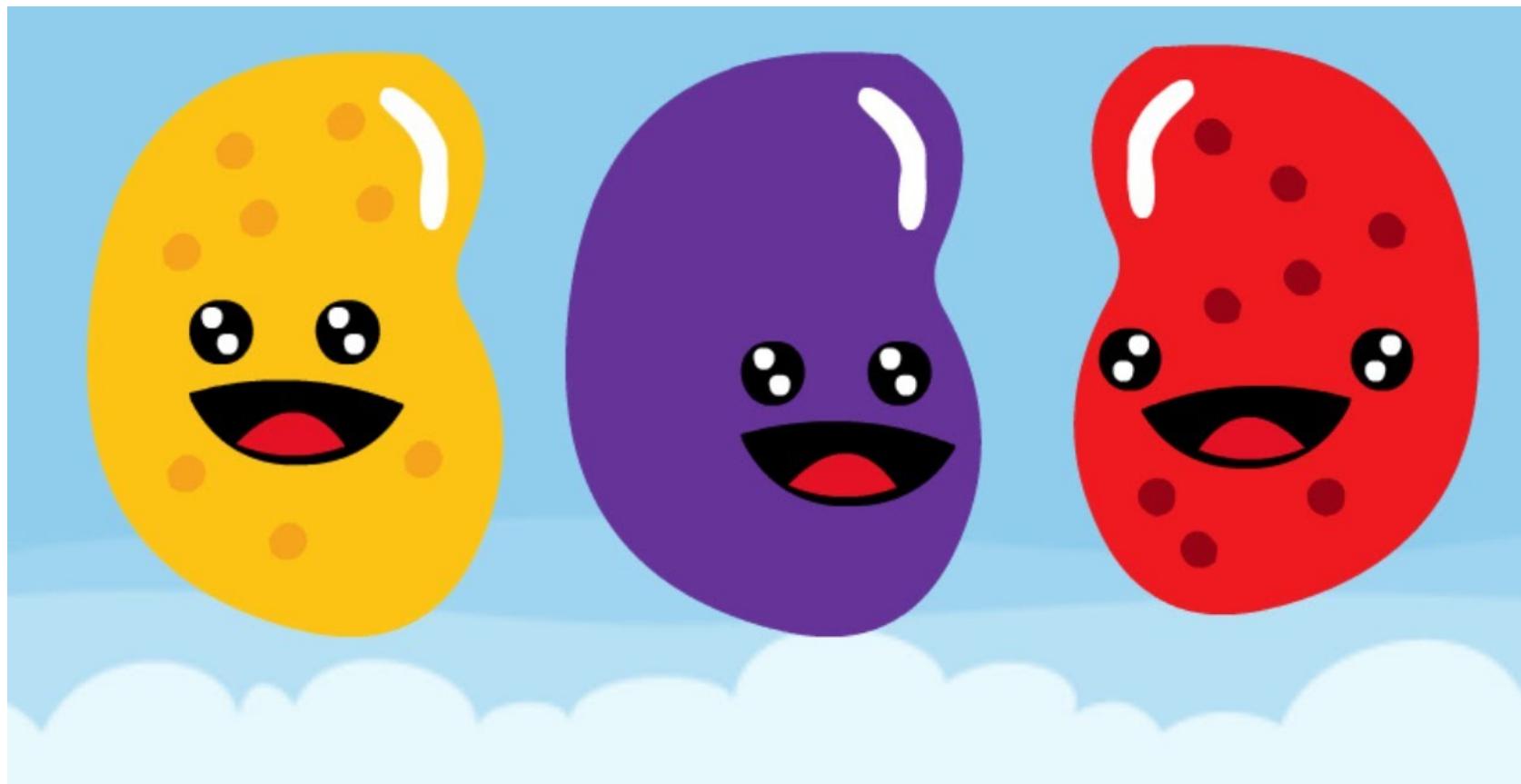
---



*JavaScript Juggernauts.*

But right now...

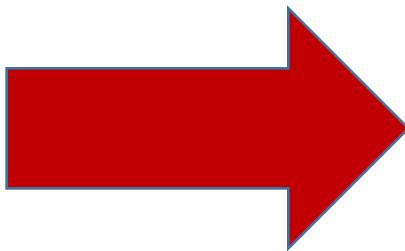
---



*Maybe feeling like*  
***JavaScript Jellybeans.***

# Transformation to Come

---



**HANG IN THERE!**

# Today's Class

# Objectives

---

**In today's class we'll be covering:**

- JavaScript Functions
- JavaScript Objects
- Building Simple JavaScript Applications

# *Functions*

---

## Code Dissection: Array Building

- Run the program sent to you via slack.
- Then, with a partner, fill in the missing comments for each line of code.
- Make sure both of you can fully explain what each line means.
- Be prepared to share with the class.

## Demo Time

---

### ***Instructor: Demo***

*(SuperHeroLogging\_NoFunctions.html | 2-SuperHeroLogging)*

# Mondo Repetitive...

```
// 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("-----");
```

**Who wants  
to maintain  
this??**

**Hint: No one.**

## Demo Time

---

### ***Instructor: Demo***

*(SuperHeroLogging\_WithFunctions.html | 2-SuperHeroLogging)*

# Much Better with Functions!

```
// 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("-----");  
}
```

# Squeaky Clean Code.

Minimal repetition

## Code Creation: Function Building

- 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.

# *Objects*

---

## Demo Time

---

***Instructor: Demo***

*(GoodArray.html | 4-GoodArray)*

## Demo Time

---

### ***Instructor: Demo***

*(JoanOfArcArrays.html | 5-JoanOfArcArrays)*

# Associated Data ==/== Arrays

---

```
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."];
```

**Relating two separate arrays is not fun.**

## Demo Time

---

### ***Instructor: Demo***

*(gandalf-the-grey-objects.html | 30-GandalfTheGreyObjects)*

# Gandalf – The Object

```
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 |     }
```

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

# Objects Visualized

var gandalf

=

{



“real name”

:

“Gandalf”

,

“age (est)”

:

11000

,

“race”

:

“Maia”

}

This is Gandalf. According to code... Gandalf is an **Object**.

# Objects Visualized

```
var gandalf
```

```
=
```

```
{
```



```
“real name”
```

```
:
```

```
“Gandalf”
```

```
,
```

```
“age (est)”
```

```
:
```

```
11000
```

```
,
```

```
“race”
```

```
:
```

```
“Maia”
```

```
}
```

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

# Objects Visualized

```
var gandalf
```

```
=
```

```
{
```



```
“real name”
```

```
:
```

```
“Gandalf”
```

```
,
```

```
“age (est)”
```

```
:
```

```
11000
```

```
,
```

```
“race”
```

```
:
```

```
“Maia”
```

```
}
```

These are the **“values”** of Gandalf’s **properties**.

# Objects Visualized

var gandalf

=

{



“real name”

:

“Gandalf”

,

“age (est)”

:

11000

,

“race”

:

“Maia”

}

Thus: gandalf[“race”] = “Maia”

## Demo Time

---

### **Instructor: Repeat Demo**

(*gandalf-the-grey-objects.html* | 30-GandalfTheGreyObjects)

## Code Dissection / Creation: 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!

# Demo Time

---

***Instructor: Demo in Browser***

(*carGame\_Solved.html | 8-CarGame*)

## Code Creation: Run that Car!

- Using the code from the previous activity as a starting point, create a complete application such that:
  - Users can enter keyboard input (letters).
  - 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.

**Everyone Do: Scope & Callbacks**

# Demo Time

---

**TA: Demo**  
(Homework Videos!)

## Extra Activity: Trivia Game

- With whatever class time remains, complete the following activity in pairs.
- 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.

# *Questions*

---