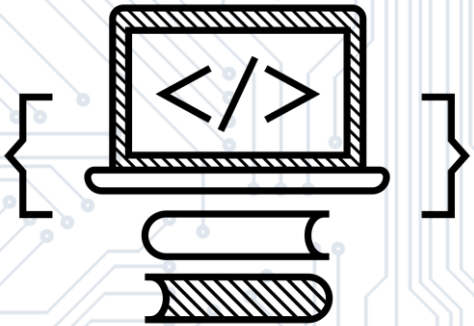


# JavaScript Juggernauts

## Web Development Boot Camp

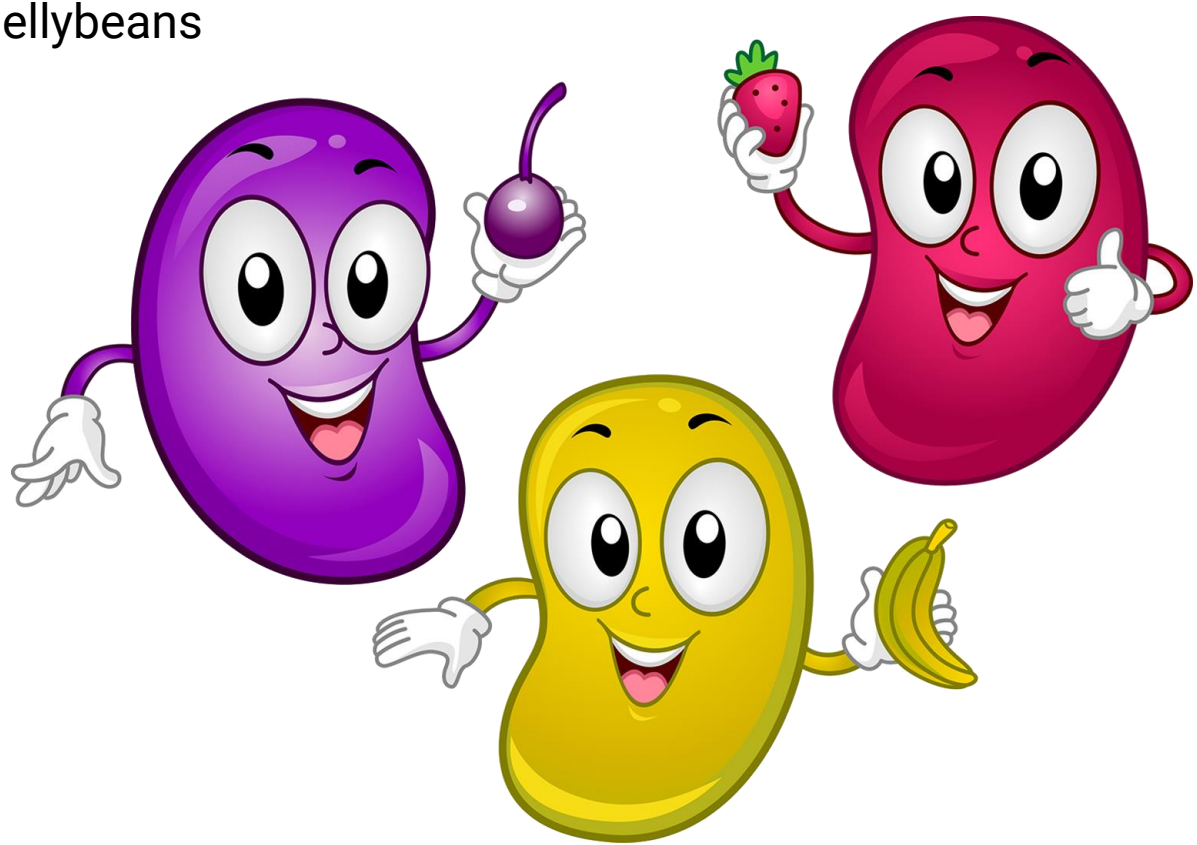
Lesson 3.3

# 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

Checkpoint (HTML/CSS/Git

# **Group Activity:**
Loop TV (Array Building)

**Suggested Time:**
10 minutes

# Group Activity: Array Building

01 Run the program sent to you via Slack.

02 Then, with your group, fill in the missing comments for each line of code.

03 **Make sure your team can fully explain what each line means.**

04 Be prepared to share with the class.

**Suggested Time:** 10 minutes

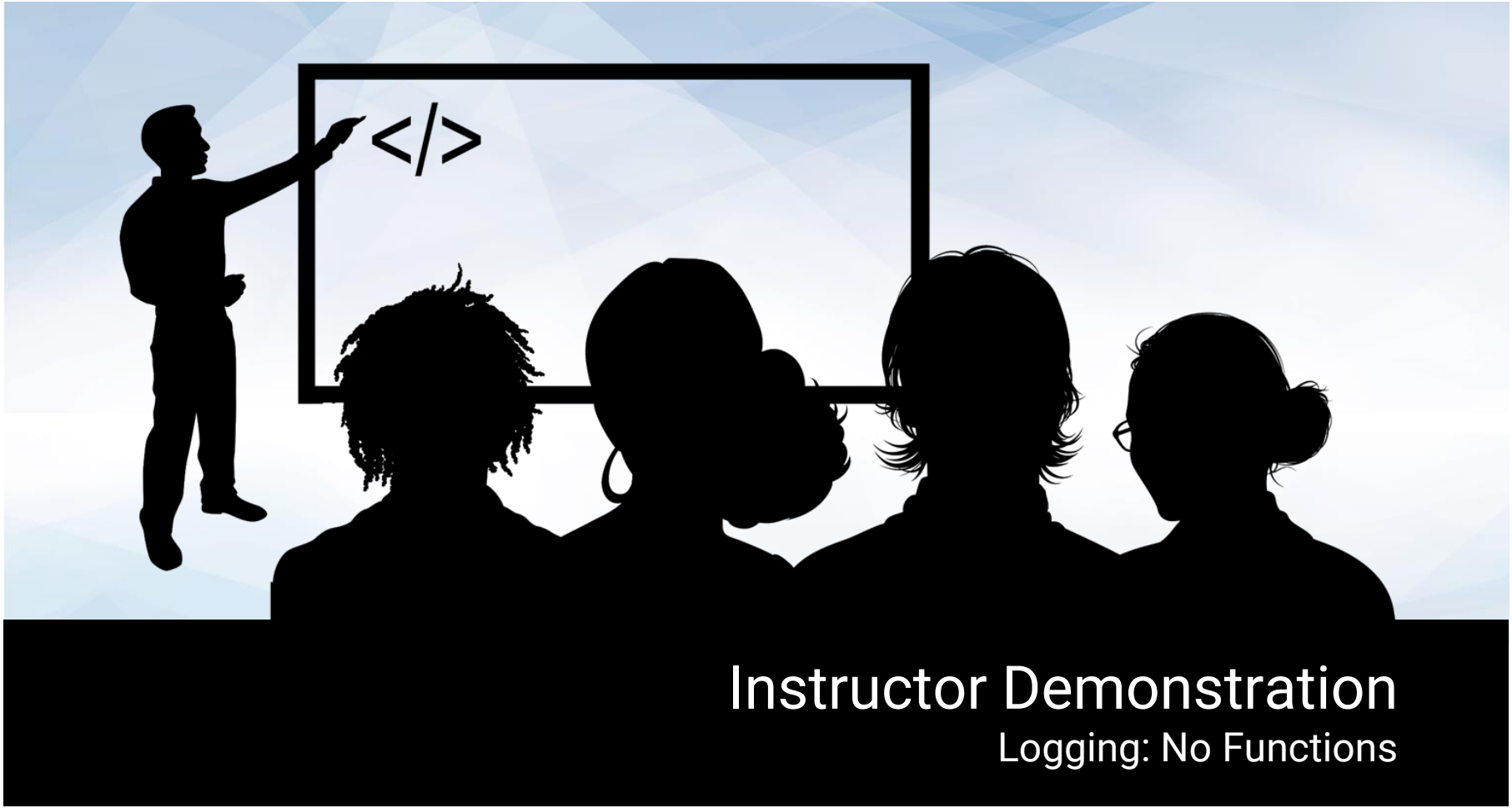
# JavaScript Functions

# Javascript Functions

- Many “programs” inside of our scripts designed to perform a particular task
- Used to segment sections of code to make it easier to manage
- Used to run repeated operations

# Javascript Functions

- Functions wrap around code blocks which contain the actual statements to be run
- Typically include some combination of variable assignment, operations, and conditions
- Functions:
  - Create a result immediately
    (e.g. change content of element on a webpage)
  - Provide an answer/output to be used by other functions or code
    (return value)



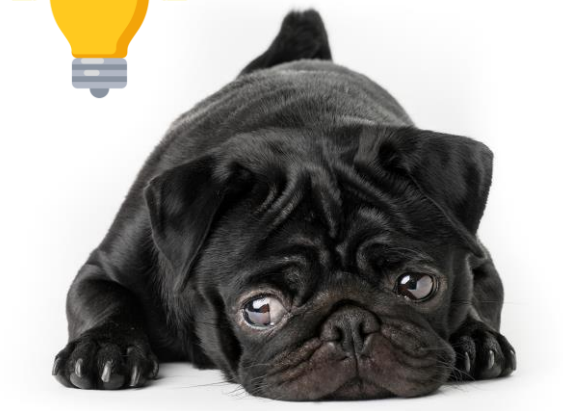

# Instructor Demonstration

## Logging: No Functions

# Mondo Repetitive

Who wants to maintain this?

**Hint:** No one.



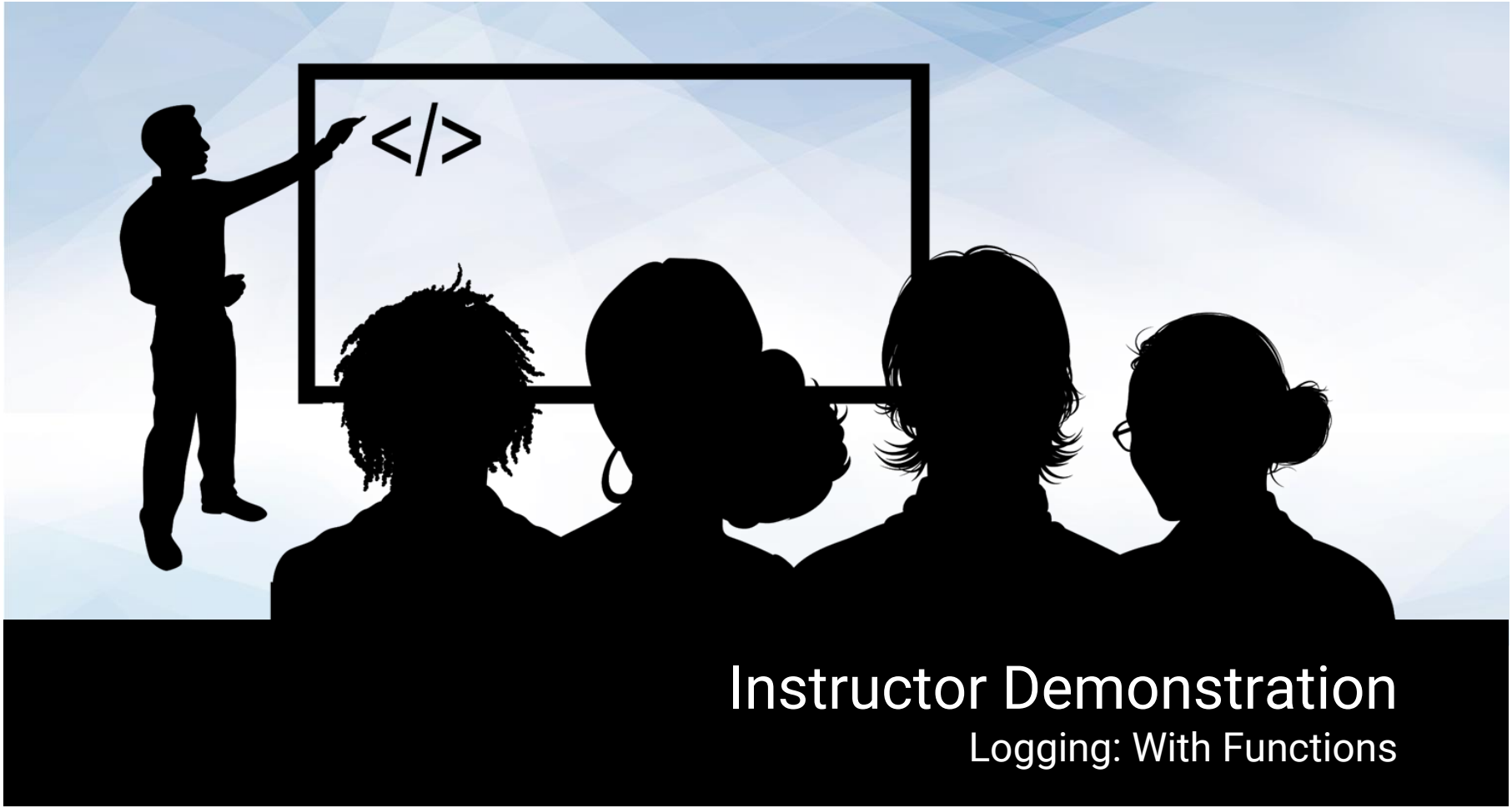
```
// For Loop for Brands
for (var i = 0; i < brands.length; i++) {
  console.log(brands[i]);
}
console.log("---------");

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

# **Group Activity:**
# My First Functions

**Suggested Time:**
20 minutes

# Group Activity: My First Functions

Working with your group 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.

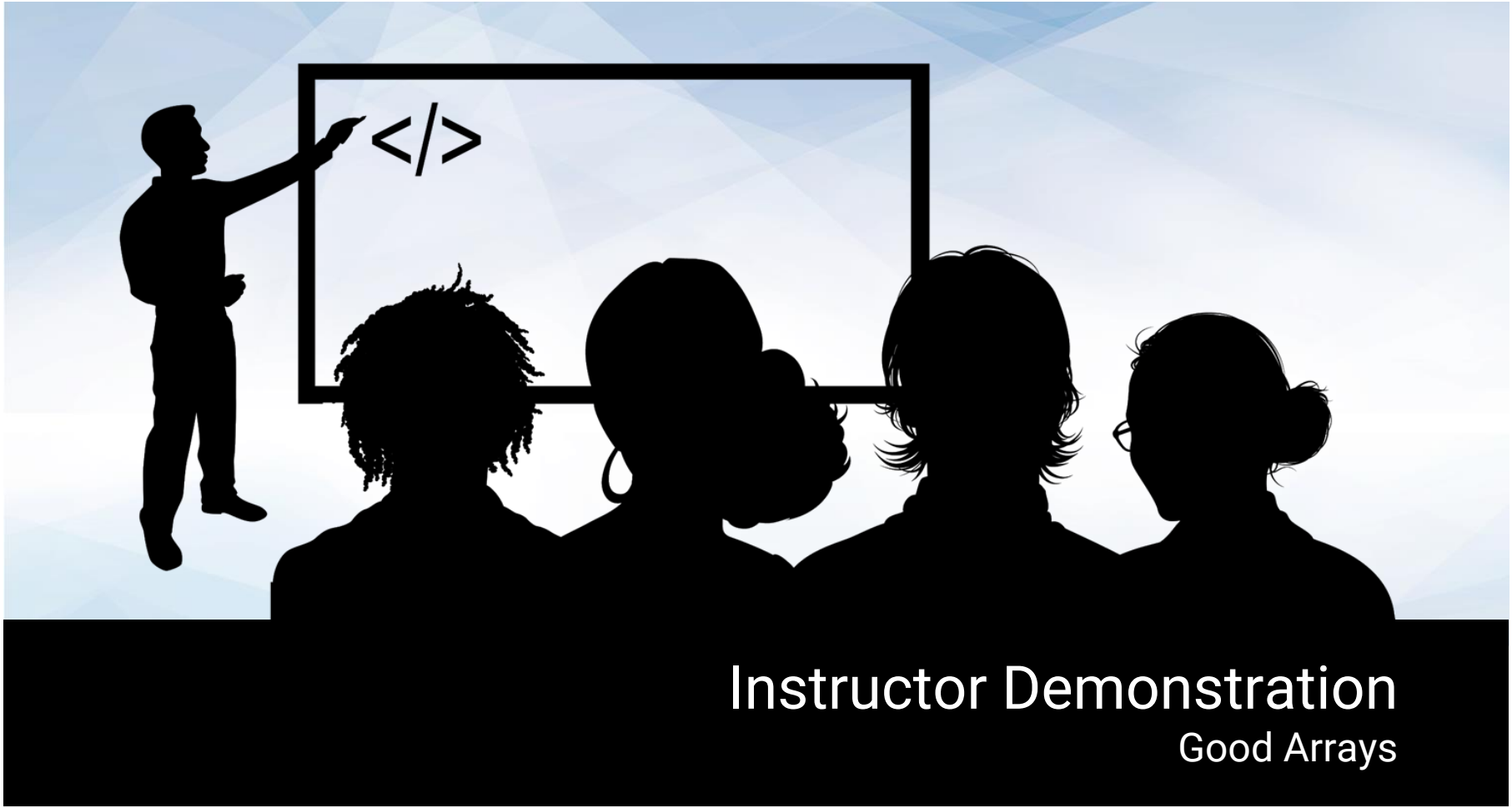
**HINT:** Look back to the previous example if you need help.

**Suggested Time:** 20 minutes

# JavaScript Objects

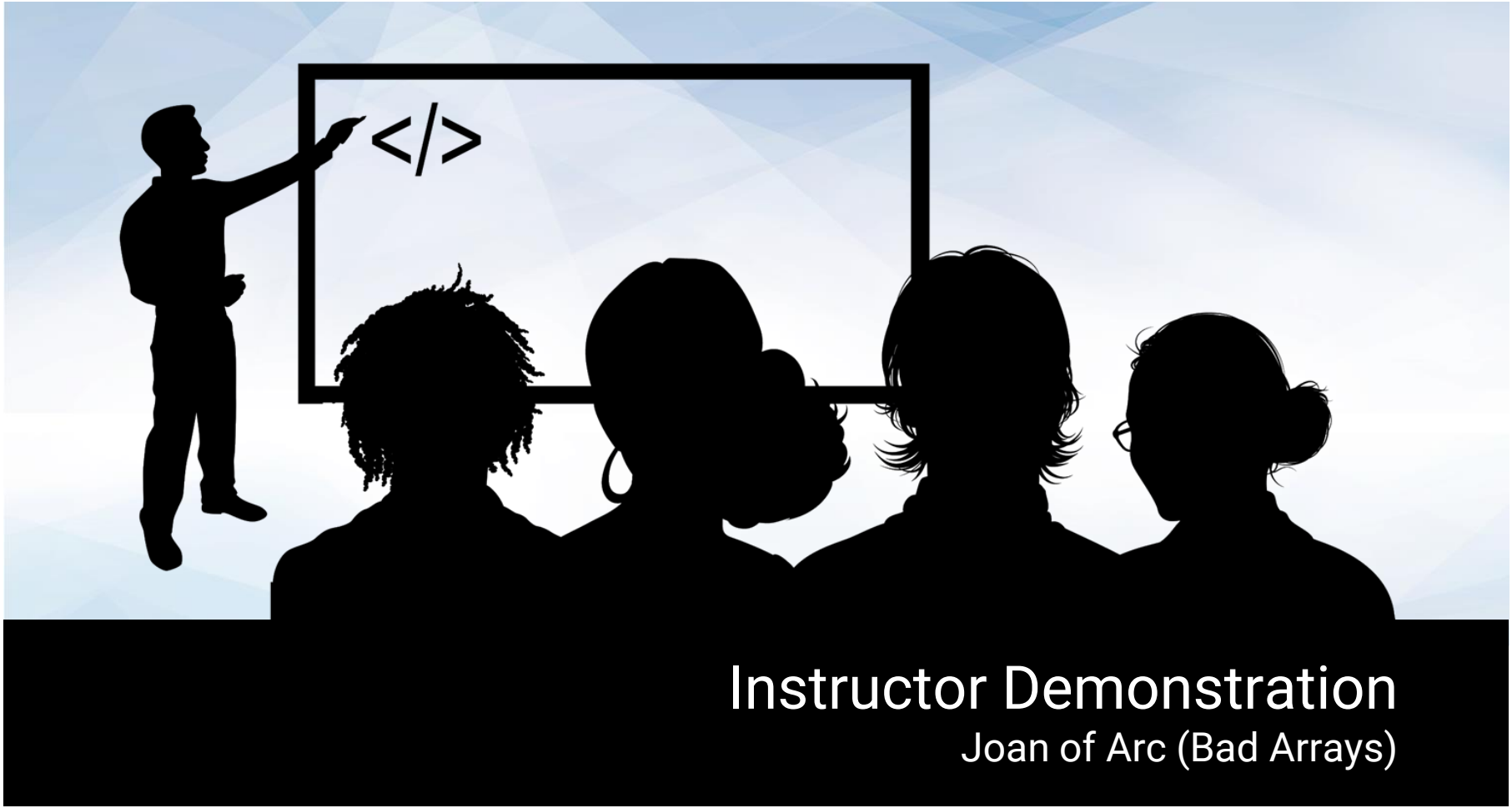
# Javascript Objects

- Javascript is an Object Oriented language
- Objects are used when we want to create a cluster of related data for something and perhaps perform some operation on that data
- Objects are data models that allow us to combine properties and methods for specific data sets in a structured way




# Instructor Demonstration

## Good Arrays

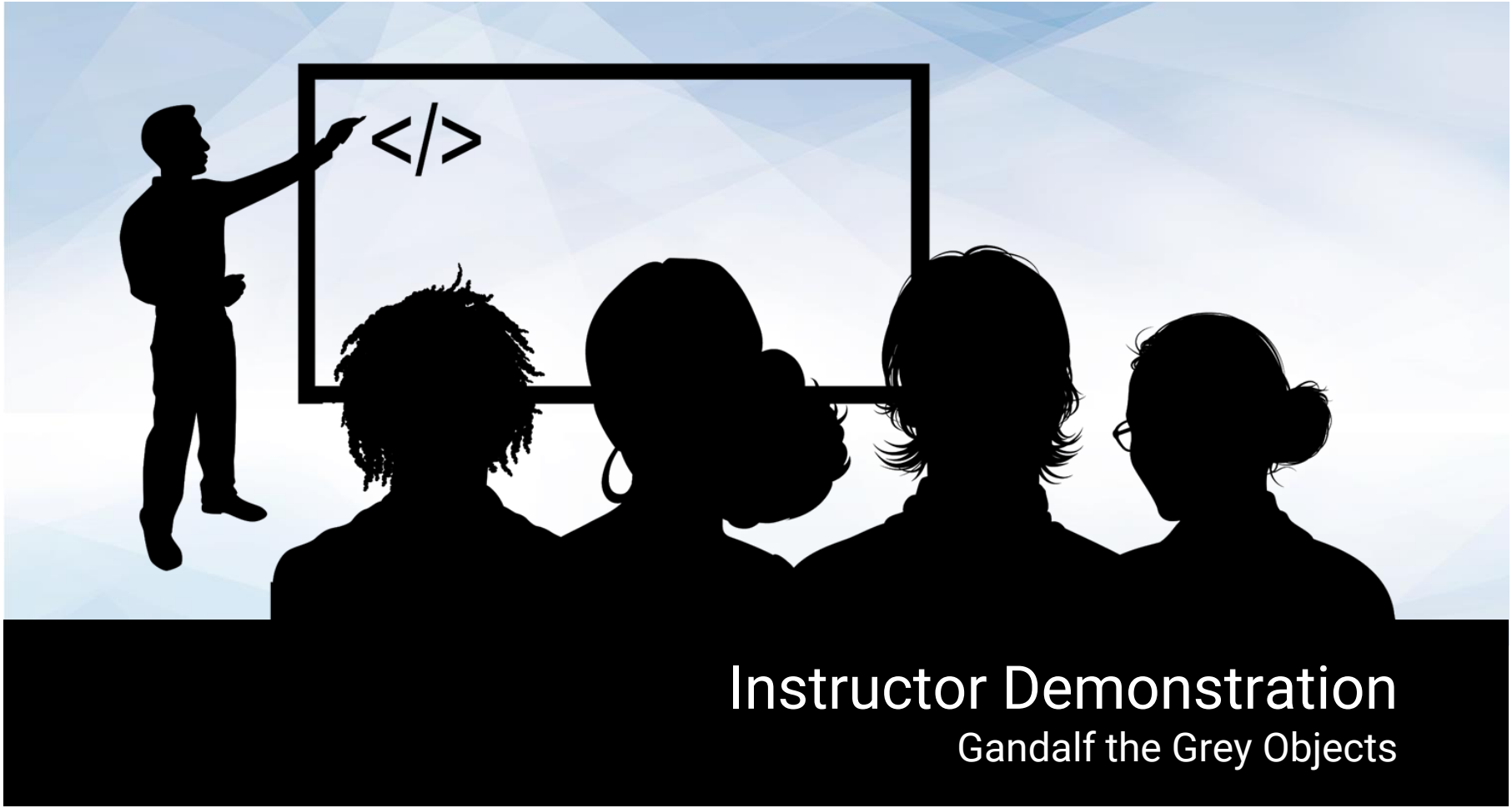

Instructor Demonstration
Joan of Arc (Bad Arrays)

# 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.

```
var gandalf = {
  "real name": "Gandalf",
  "age (est)": 11000,
  "race": "Maia",
  "haveRetirementPlan": true,
  "aliases": [
    "Greyhame",
    "Stormcrow",
    "Mithrandir",
    "Gandalf the Grey",
    "Gandalf the White"
  ]
}

// Object properties can be accessed with "bracket notation"
alert("My name is " + gandalf["real name"]);

// Or with "dot notation" if the property has no spaces
if (gandalf.haveRetirementPlan) {

  // Or with a variable that matches the name of the property
  var ageProperty = "age (est)";
  var years = gandalf[ageProperty];
  alert("My 401k has been gathering interest for " + years + " years!");
}
```

# 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

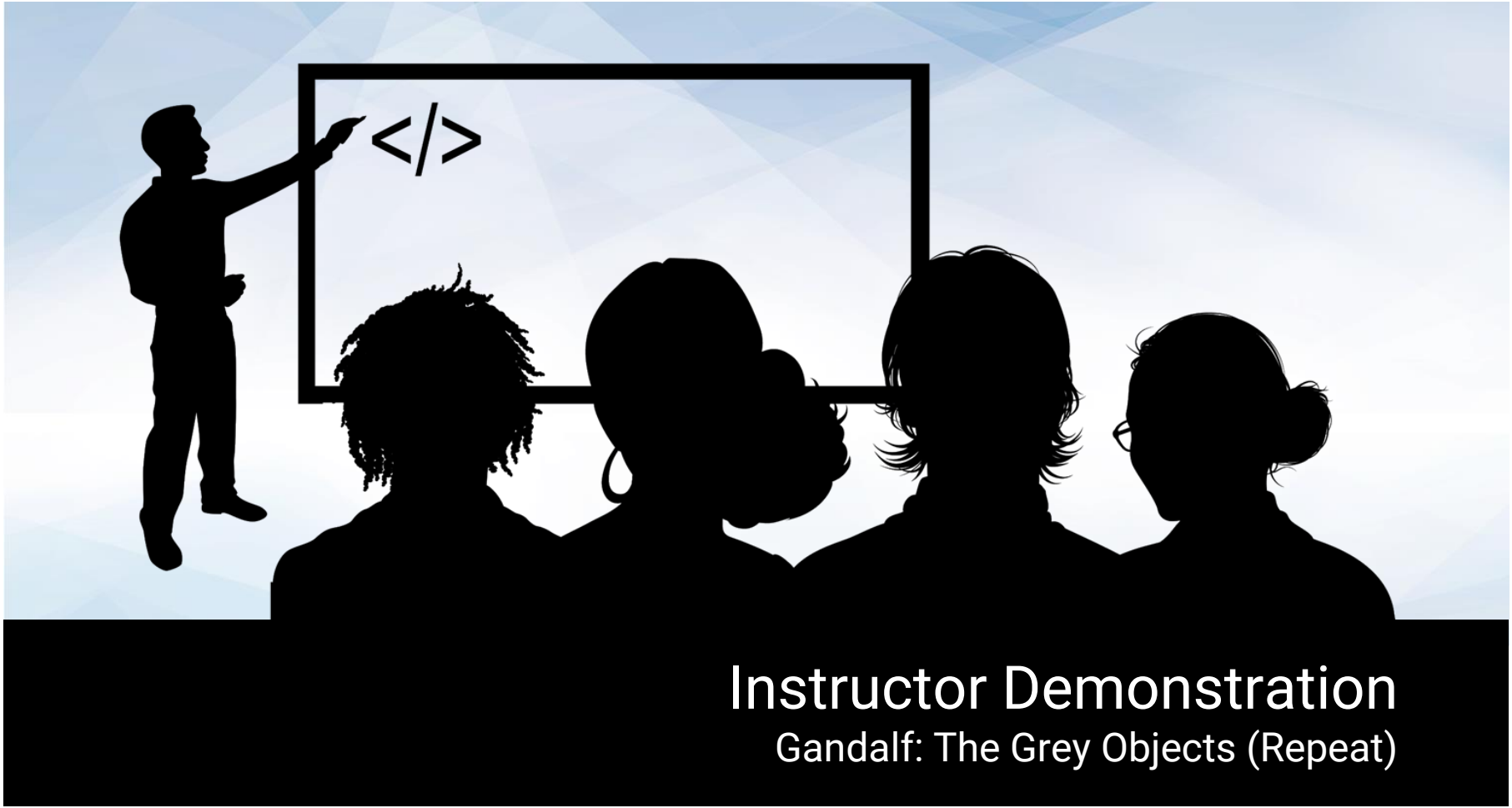
These are the **values** of Gandalf's properties.

```
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)

# **Activity:** Basic Objects

**Suggested Time:**
15 minutes

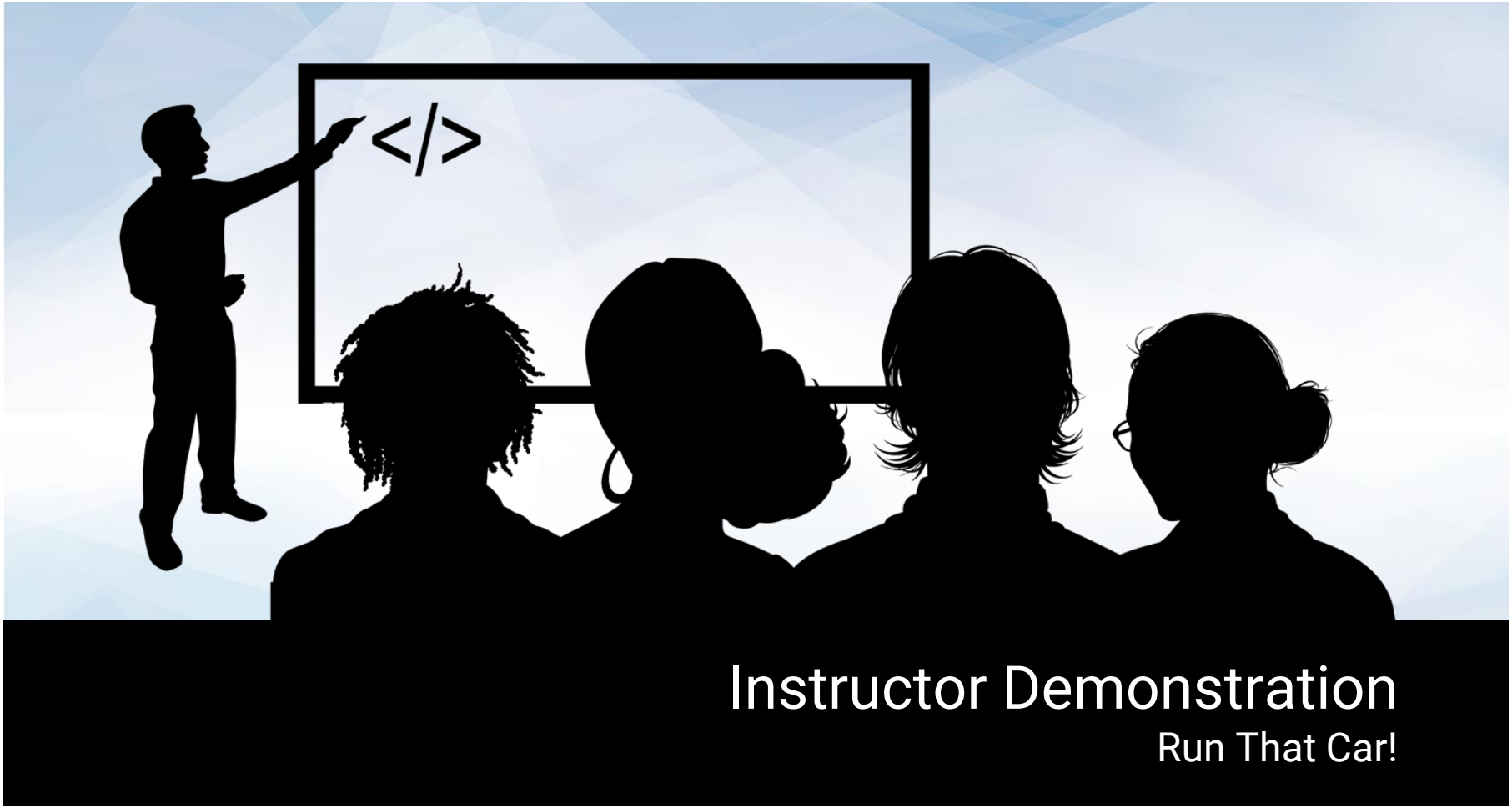
# Activity: Basic Objects

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


Instructor Demonstration
Run That Car!

# **Challenge:** Run That Car!

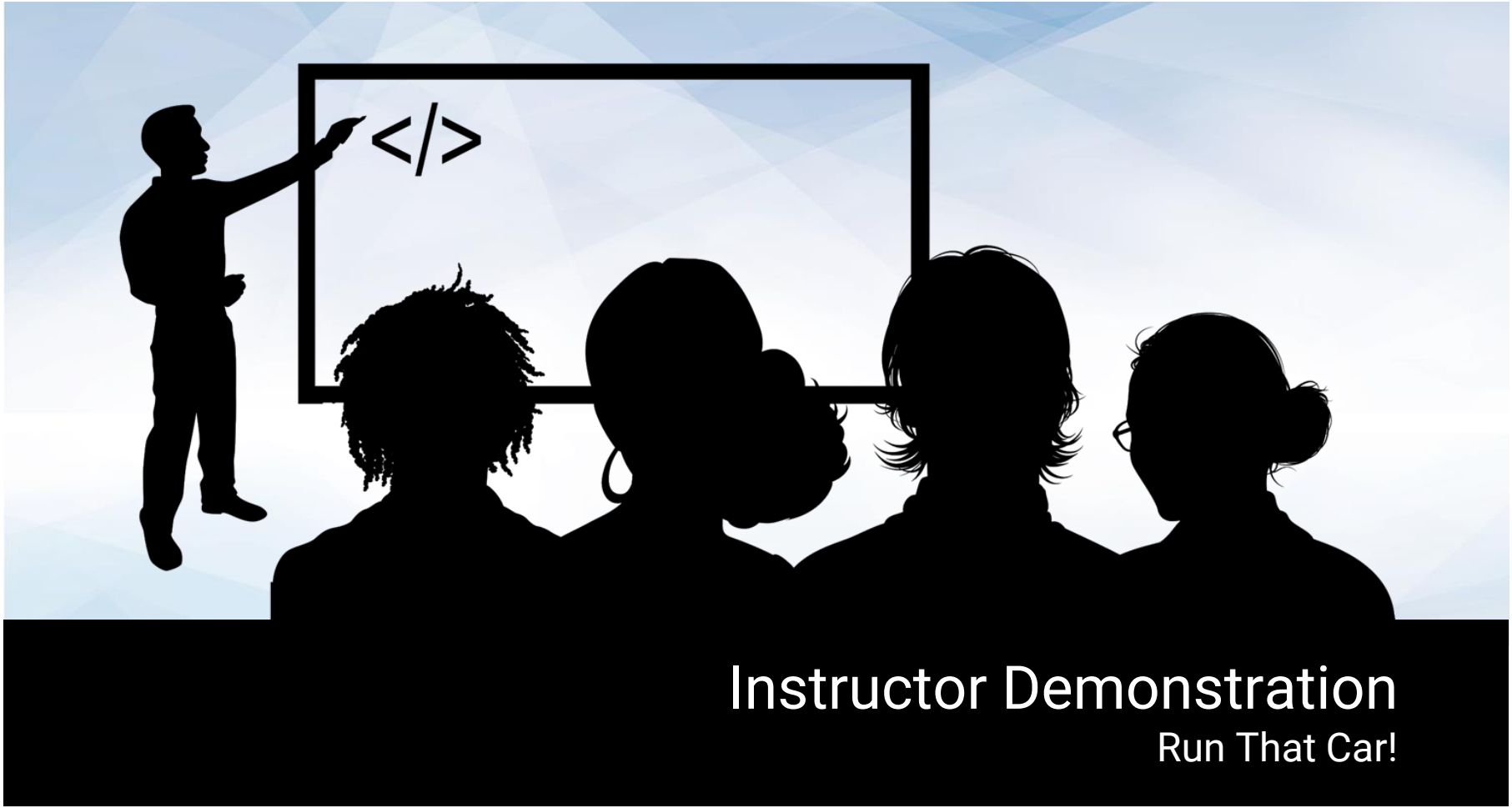
**Suggested Time:**
30 minutes

# 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


Instructor Demonstration
Run That Car!

# **Activity:** Scope & Callbacks

**Instructions sent via Slack**

**Suggested Time:**
20 minutes

# Workbook and Homework

# Group Activity: Question Game

**Suggested Time:**
45 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: 45 minutes

# Questions?