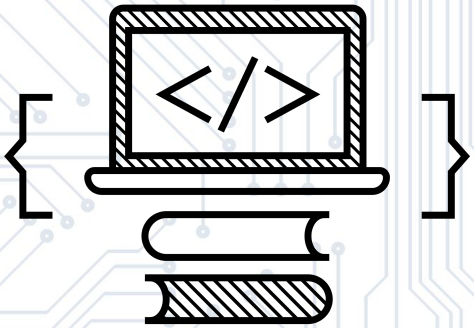
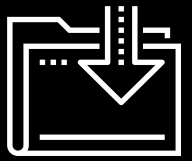


JavaScript Juggernauts

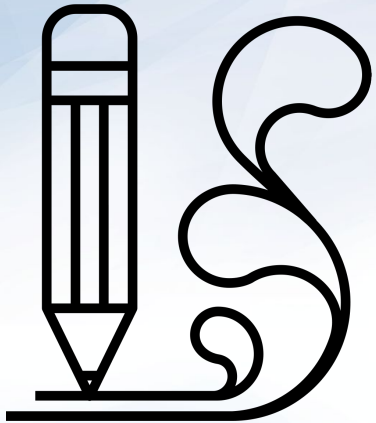
Web Development Boot Camp

Lesson 3.3



Admin Items

- Homework 2: Due today
- Pace is picking up, huh?
- Issues turning in homeworks?
- Week 4: Document Object Model Web API
- Office hours thread in **`#general`** - Dan



Homework 3:
Password Generator

Due Date:
Tuesday September 22

Today’s Class

Objectives

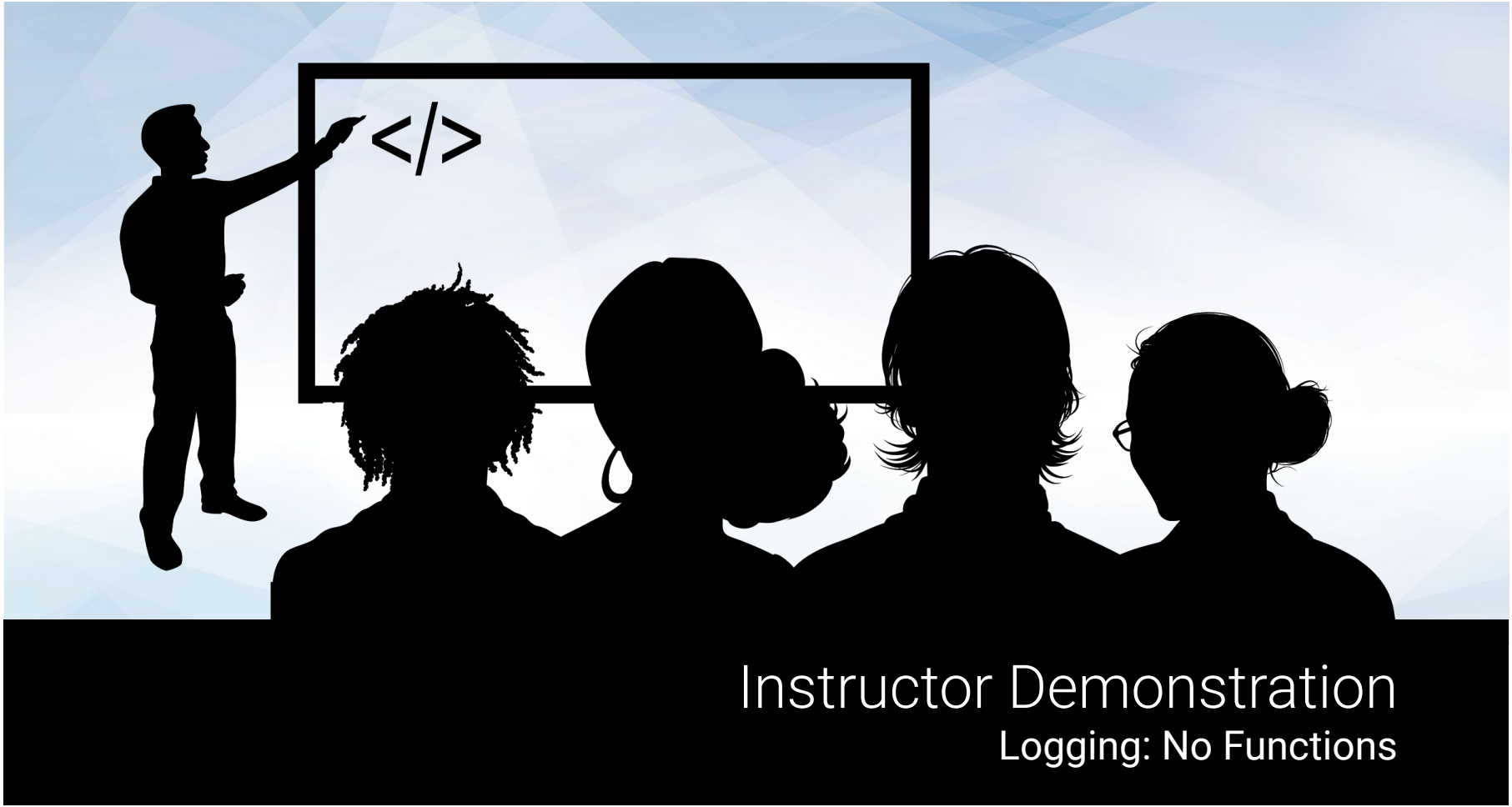
In today's class, we'll cover:

JavaScript Functions

JavaScript Objects

Building Simple JavaScript Applications

JavaScript Functions



Instructor Demonstration

Logging: No Functions

Mondo Repetitive

Who wants to maintain this?

Hint: No one.

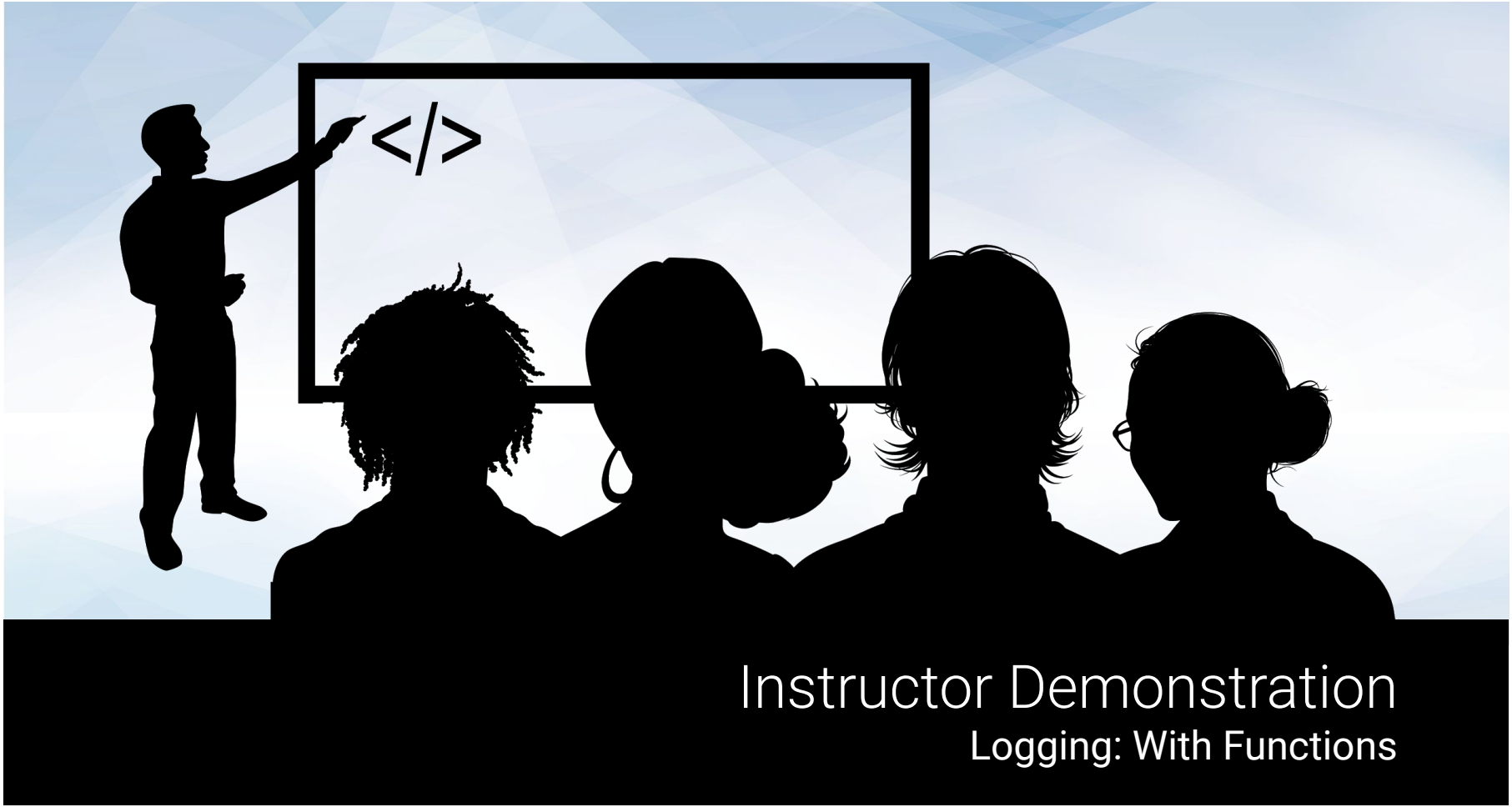
```
// For Loop for Brands
for (var i = 0; i < brands.length; i++) {
  console.log(brands[i]);
}
console.log("---------");

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

Functions are like “tools”

3 important concepts

01

Function definition

Define a behavior

- Parameters
- Body
- Return value

02

Function invocation

Trigger the function

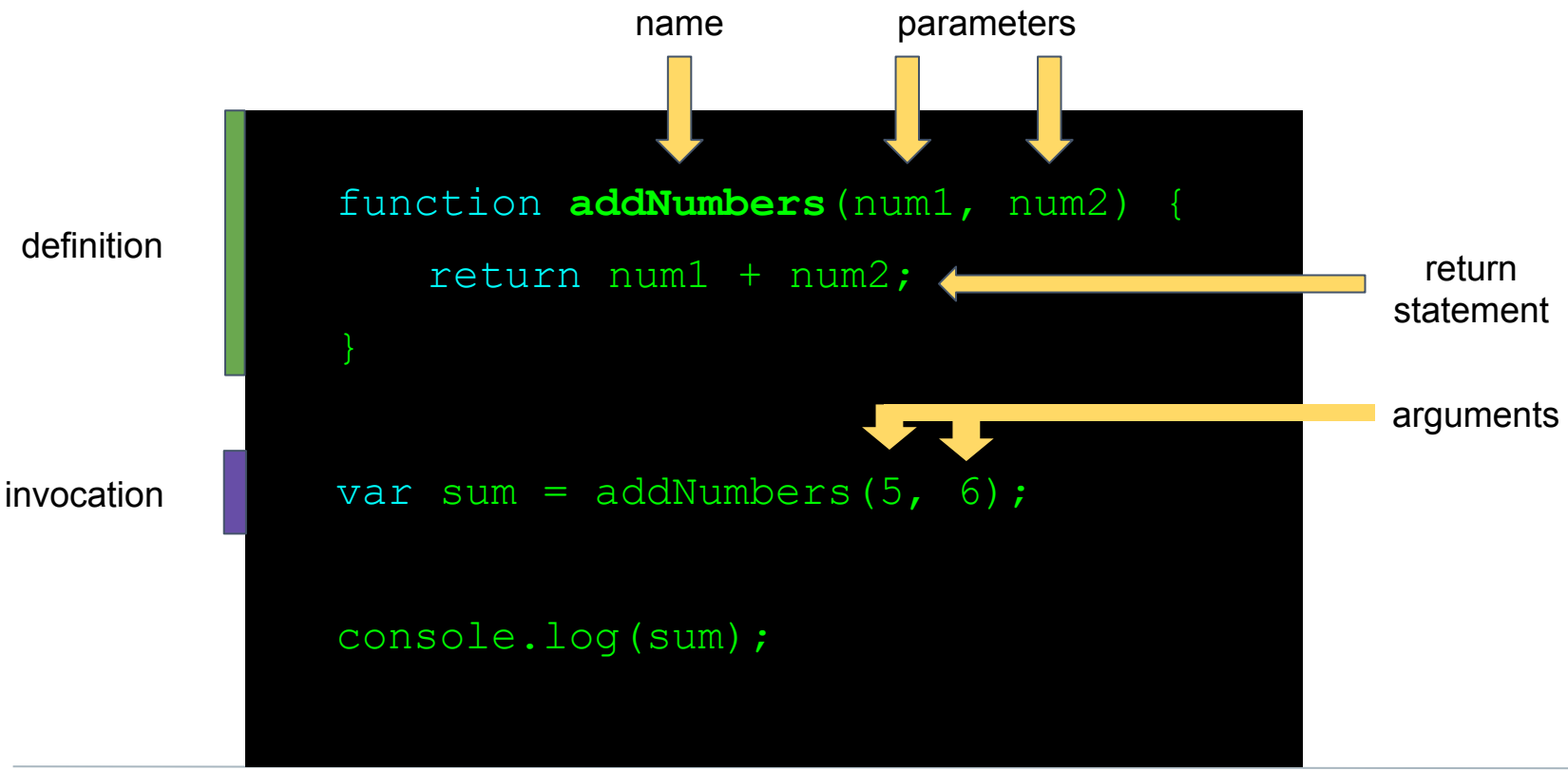
- Function name
- Invoke operator ()
- Arguments passed through parameters via the invoke operator

03

Functions are things, too

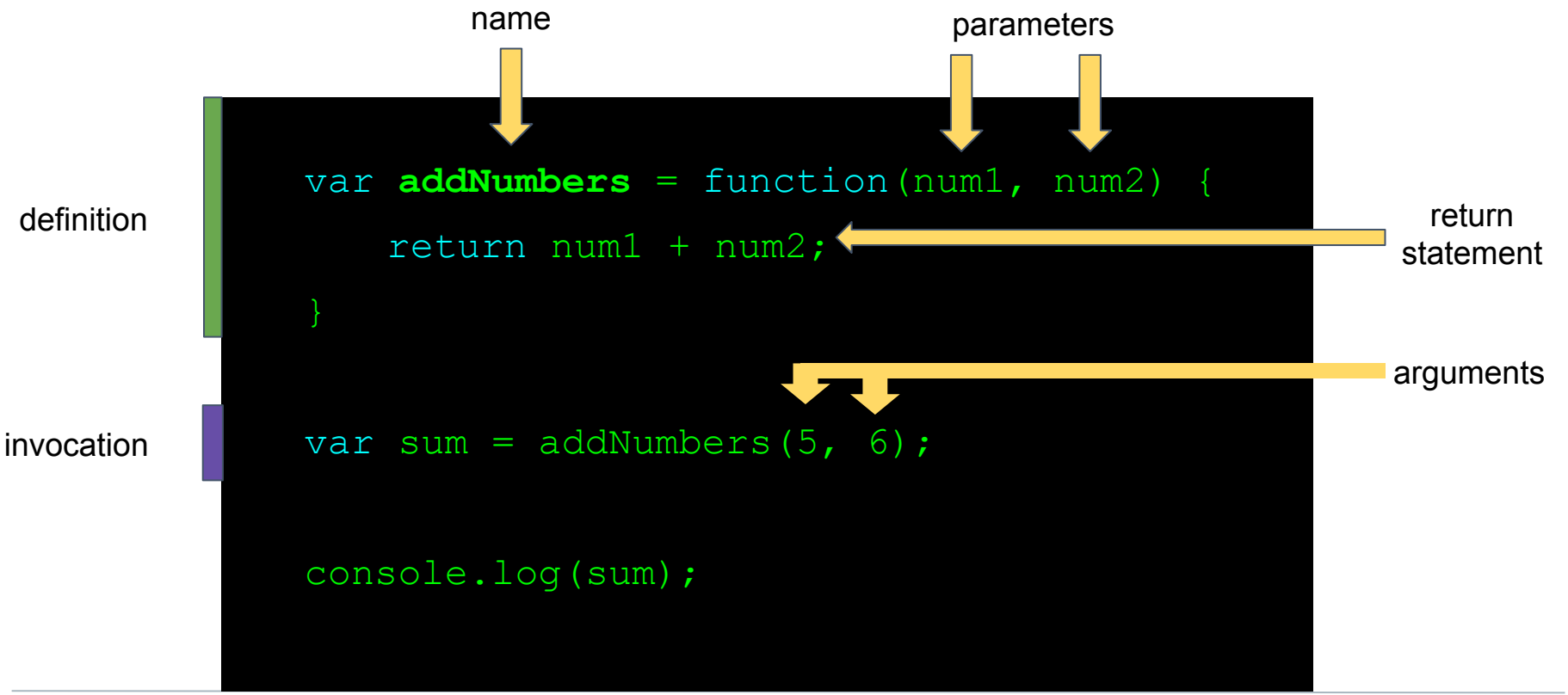
A function name can be used like a variable name

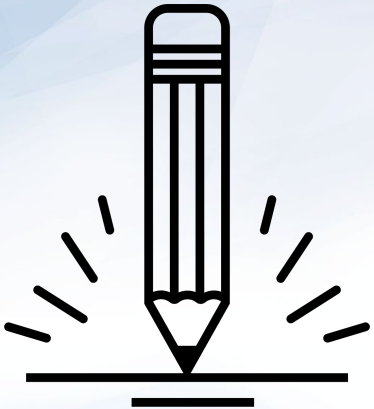
Function parts



Function parts

Function expression: a function is a thing too!





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 worry if you only get one or two. The important thing is that you completely finish at least one function.

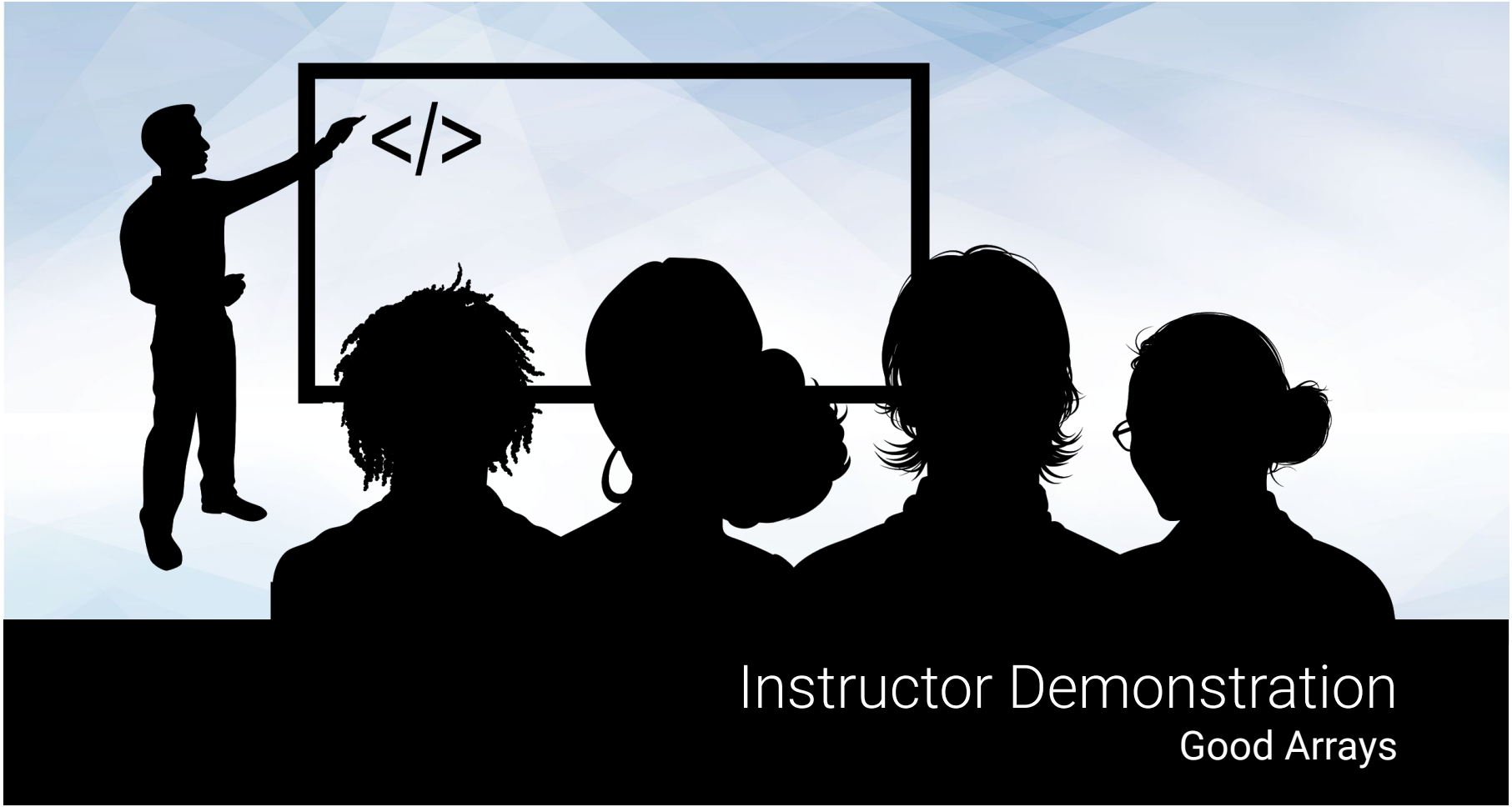
HINT: Look back to the previous example if you need help.

Suggested Time: 20 minutes

JavaScript Objects

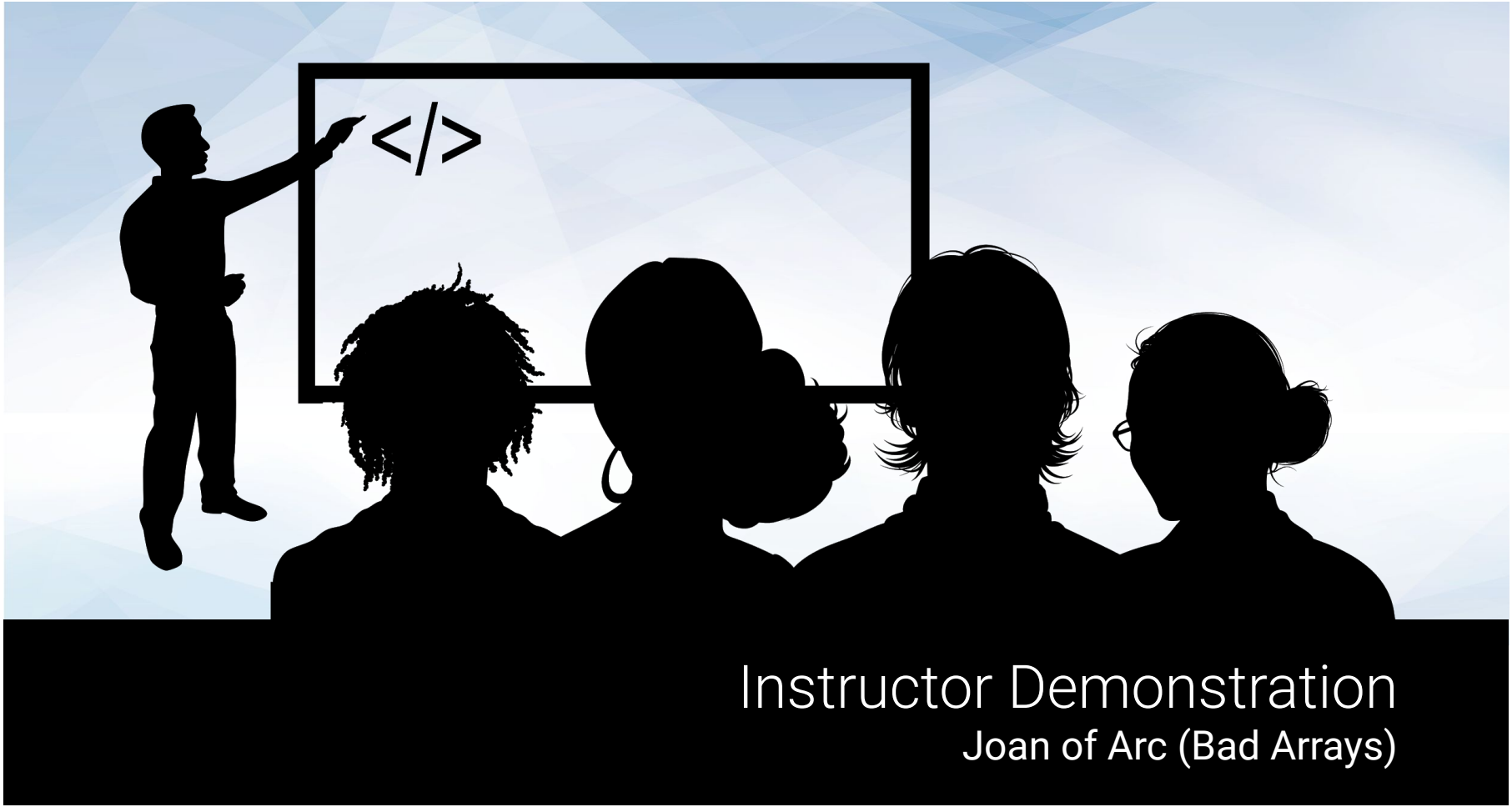
Objects are our second **data structure**.

What was the first?



Instructor Demonstration

Good Arrays



Instructor Demonstration

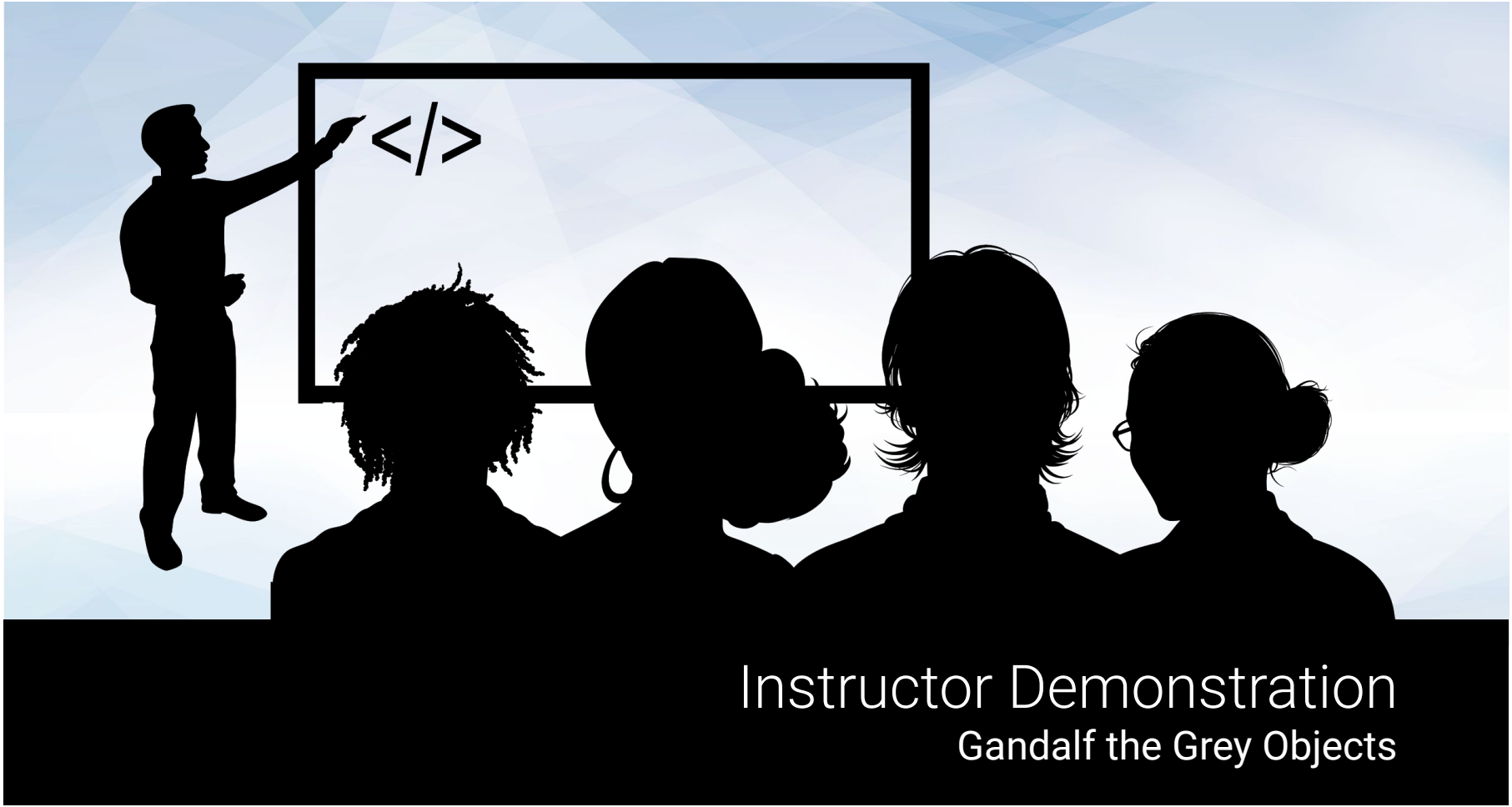
Joan of Arc (Bad Arrays)

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"`

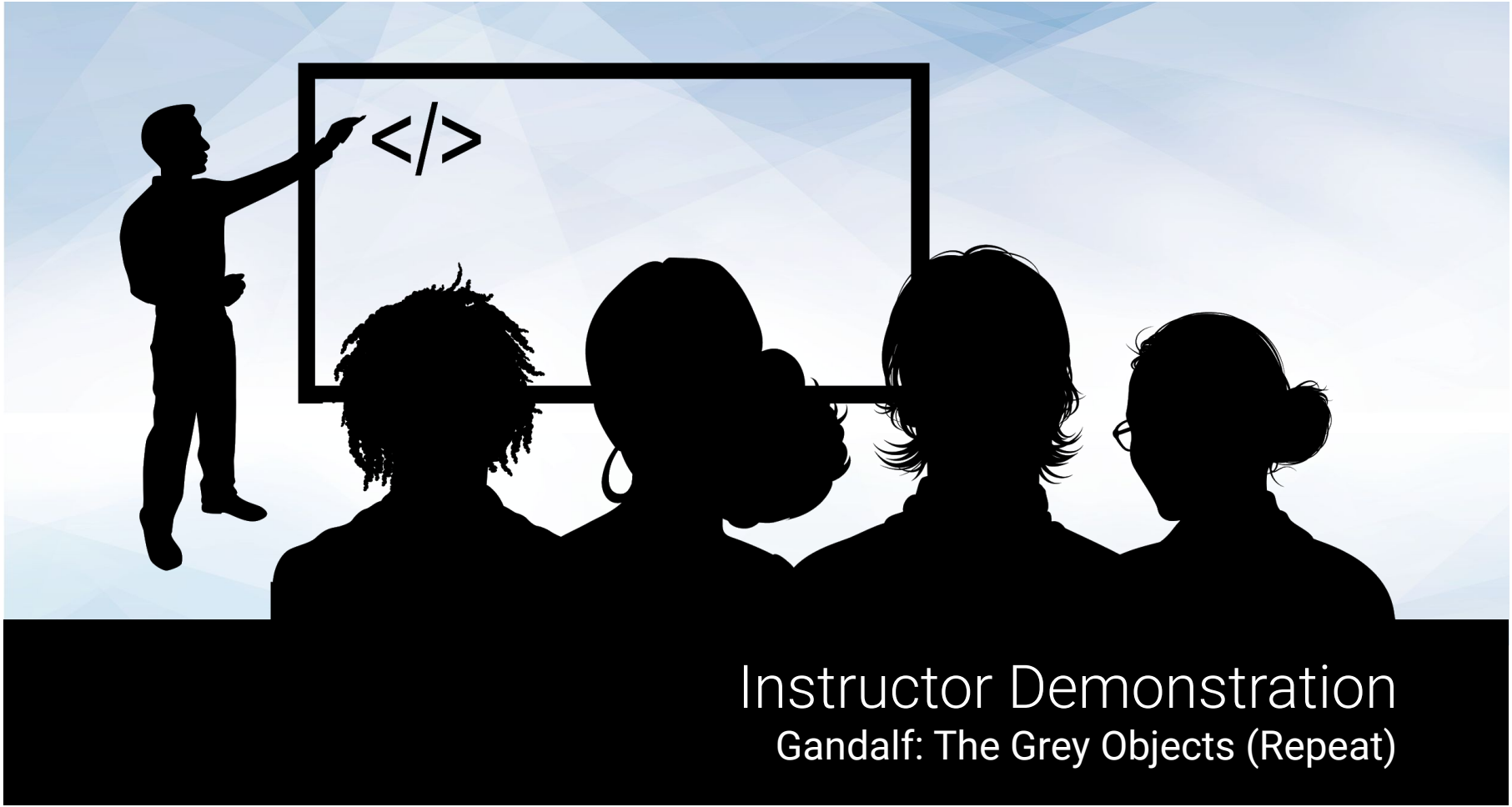
var gandalf	=	{

"real name"	:	"Gandalf"	,

"age (est)"	:	11000	,

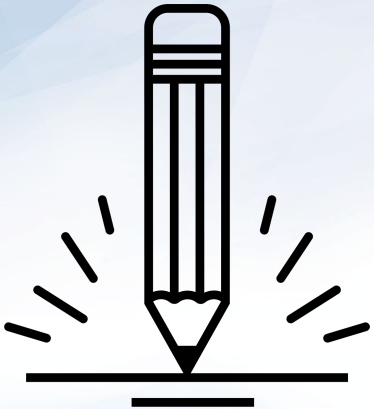
"race"	:	"Maia"

}



Instructor Demonstration

Gandalf: The Grey Objects (Repeat)



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

Suggested Time:
15 minutes

Group Activity: Basic Objects

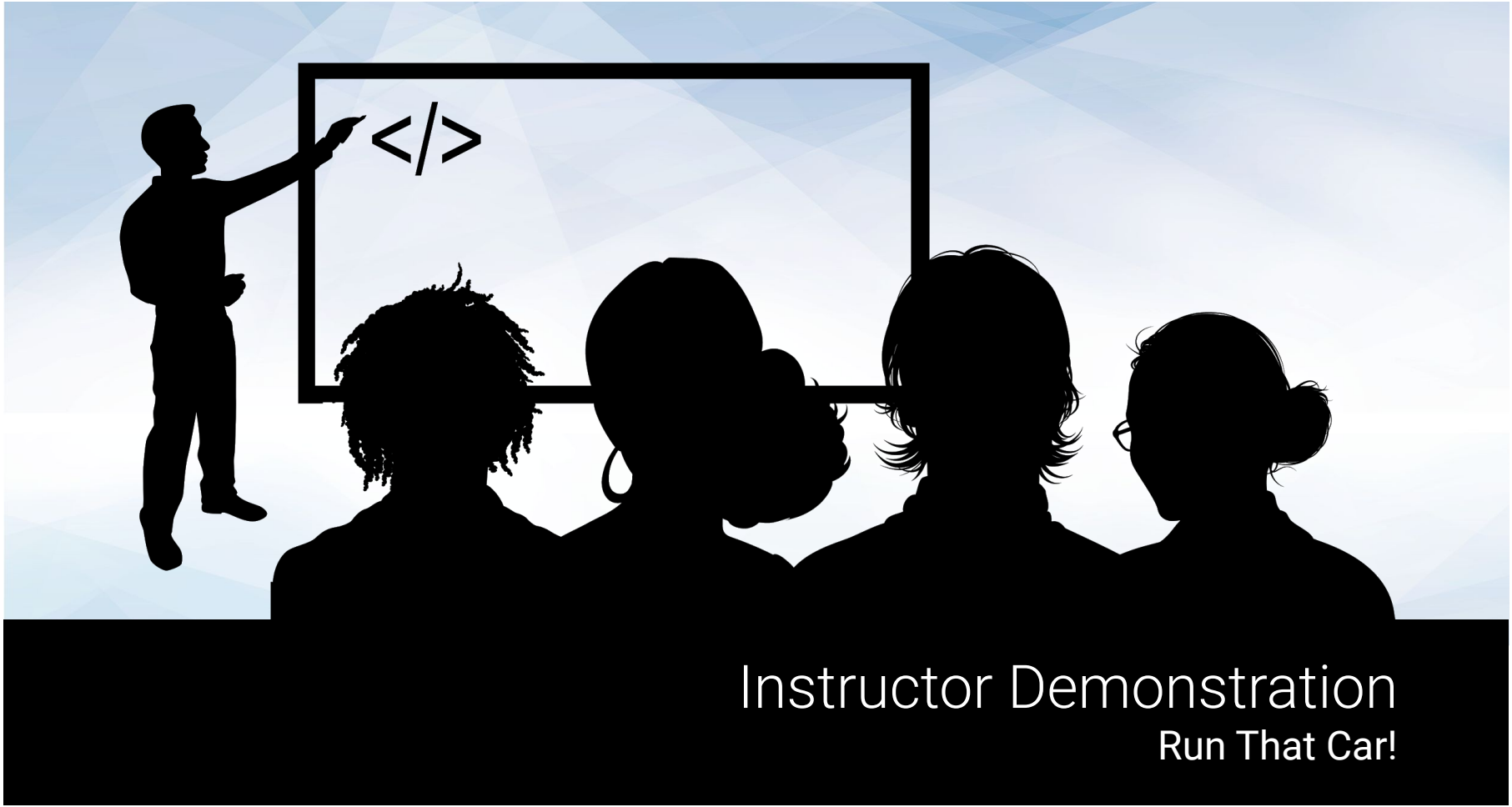
With a partner, spend a few minutes studying the code in **31-MyFirstObject**.

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

Bonus: If you finish early, create a 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


Break



Instructor Demonstration

Run That Car!

Challenge: Run That Car!

Suggested Time:
15 minutes

Challenge: Run That Car!

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

- 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.

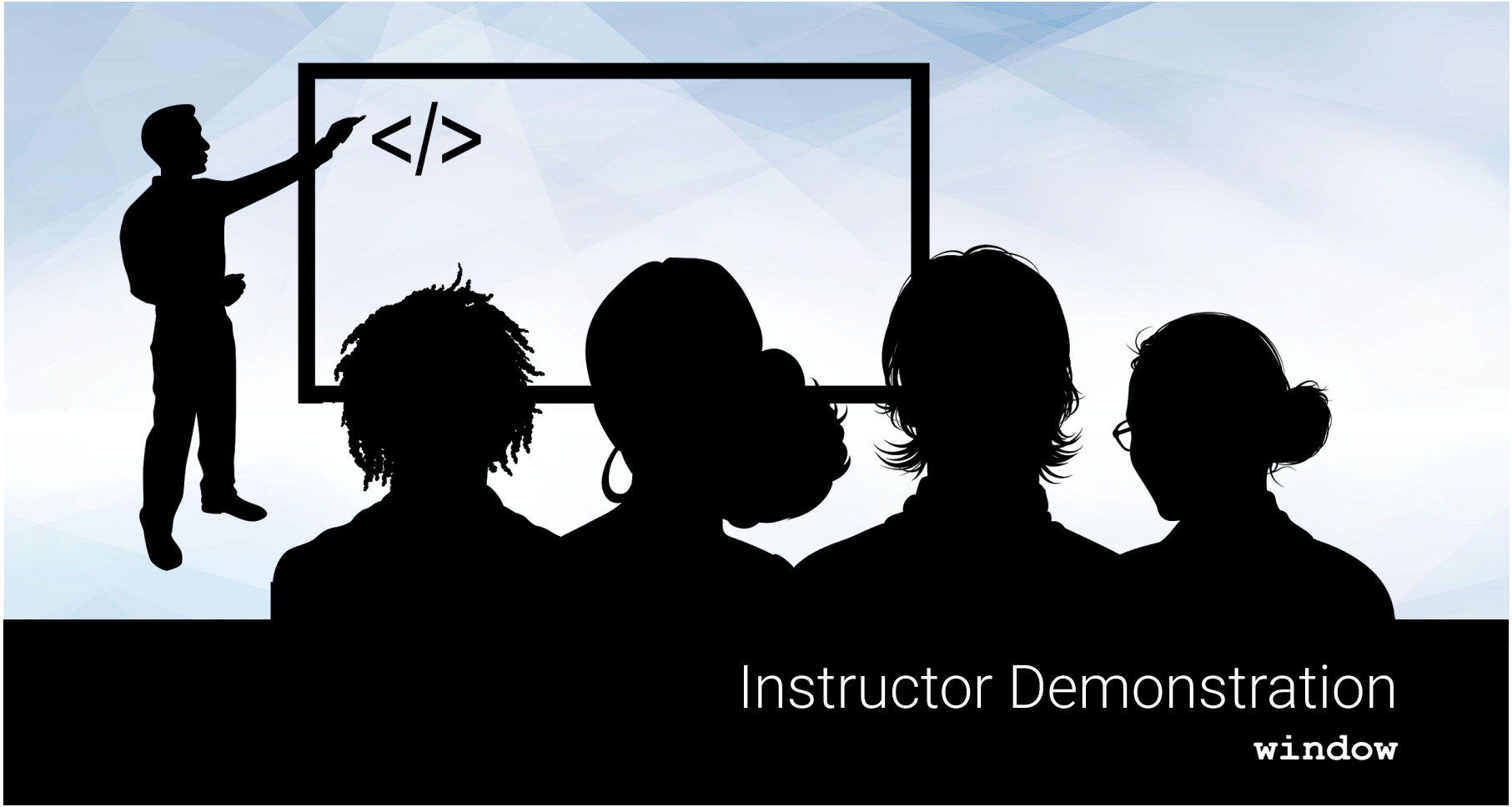
Suggested Time: 15 minutes

`window`: Our first Web API

`window`

The Browser Object Model

- A javascript object (yep. Just a big old object with properties and methods)
- Represents the browser itself
- An interface for programmers (hey that's us!) to the browser and its services
- **`alert, confirm, prompt:`** all part of window
- **`console,`** too!



Instructor Demonstration

`window`

Office Hours!