# JavaScript with D3.js

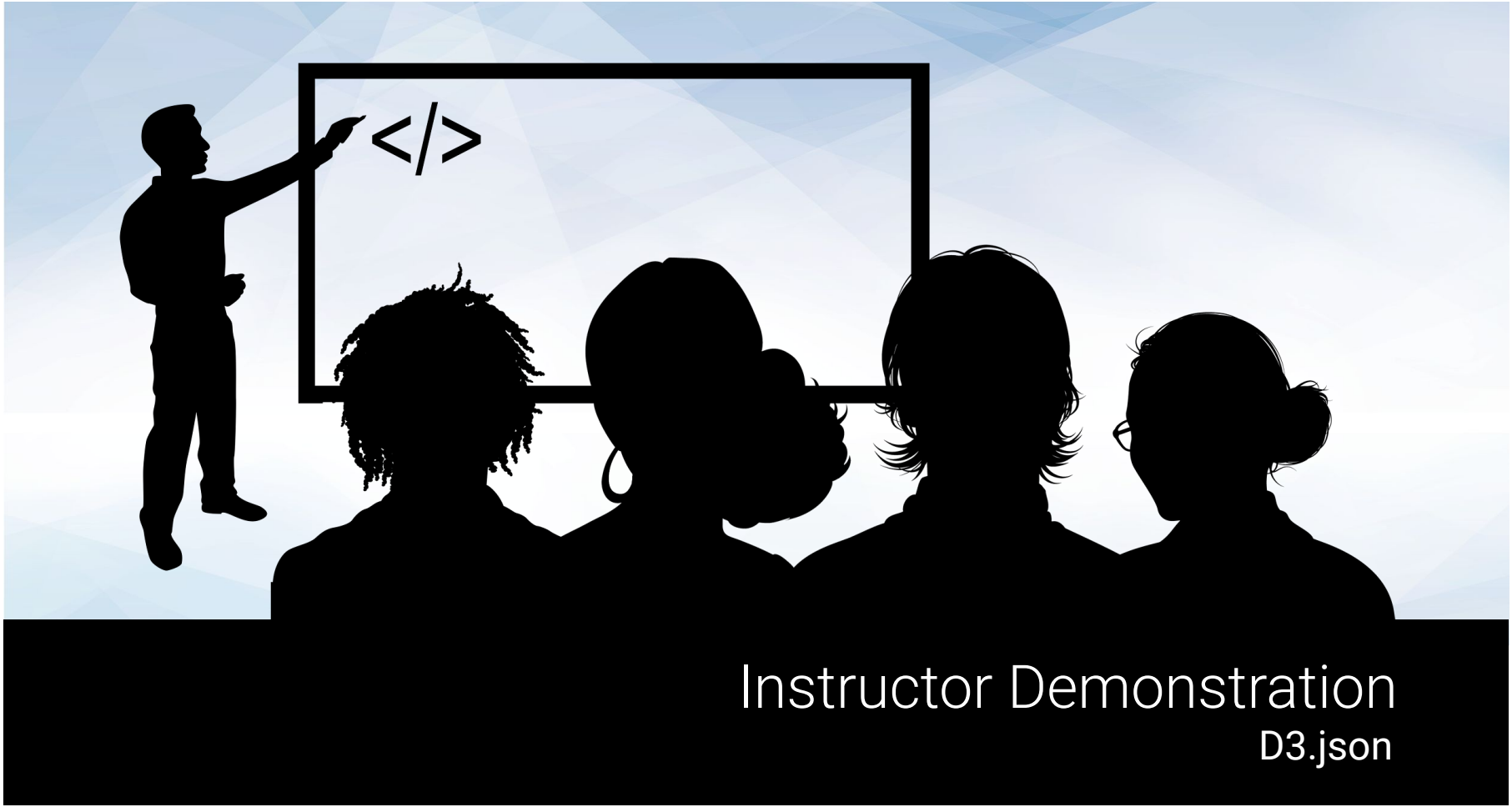
Data Boot Camp

Lesson 14.3

# Class Objectives

By the end of this lesson, you will be able to:

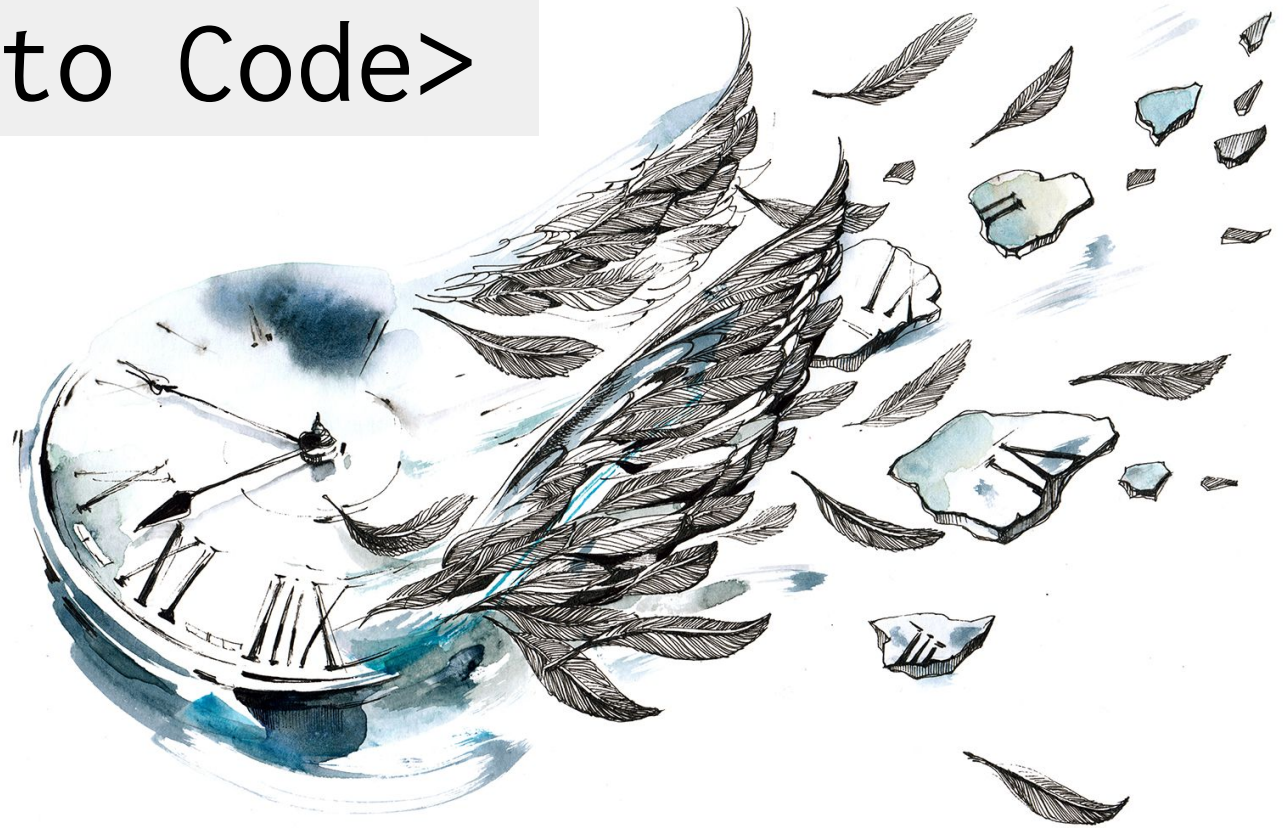
- Create charts using data from API calls.
- Use D3 for basic document object model (DOM) manipulation and event handling.
- Apply the `this` keyword to reference elements within a function.
- Dynamically manipulate the DOM through events.
- Manipulate charts through dropdown events and click events.
- Use `Plotly.restyle()` to create dynamic charts.




# Instructor Demonstration

## D3.json

# <Time to Code>



# Instructor Do: D3.json

index.html

<Time to Code>

Importing D3 using a CDN link, in a script tag.

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Document</title>
</head>
<body>
  <h1>Open the console!</h1>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/d3/5.9.7/d3.min.js"></script>
  <script src="demo.js"></script>
</body>
</html>
```

demo.js

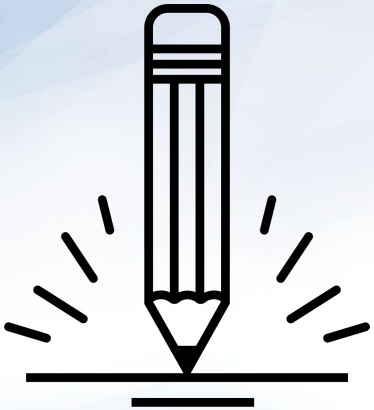
`d3.json` returns a JS promise, which places an API call to the `url`.

```js
const url = "https://api.spacexdata.com/v2/launchpads";

// Promise Pending
const dataPromise = d3.json(url);
console.log("Data Promise: ", dataPromise);

// Fetch the JSON data and console log it
d3.json(url).then(function(data) {
  console.log(data);
});
```

The argument, labeled here as `data` represents the accessible data from the API call.



# Activity: D3.json

In this activity, you will use D3.json to make API calls to SpaceX.

**Suggested Time:**
15 Minutes

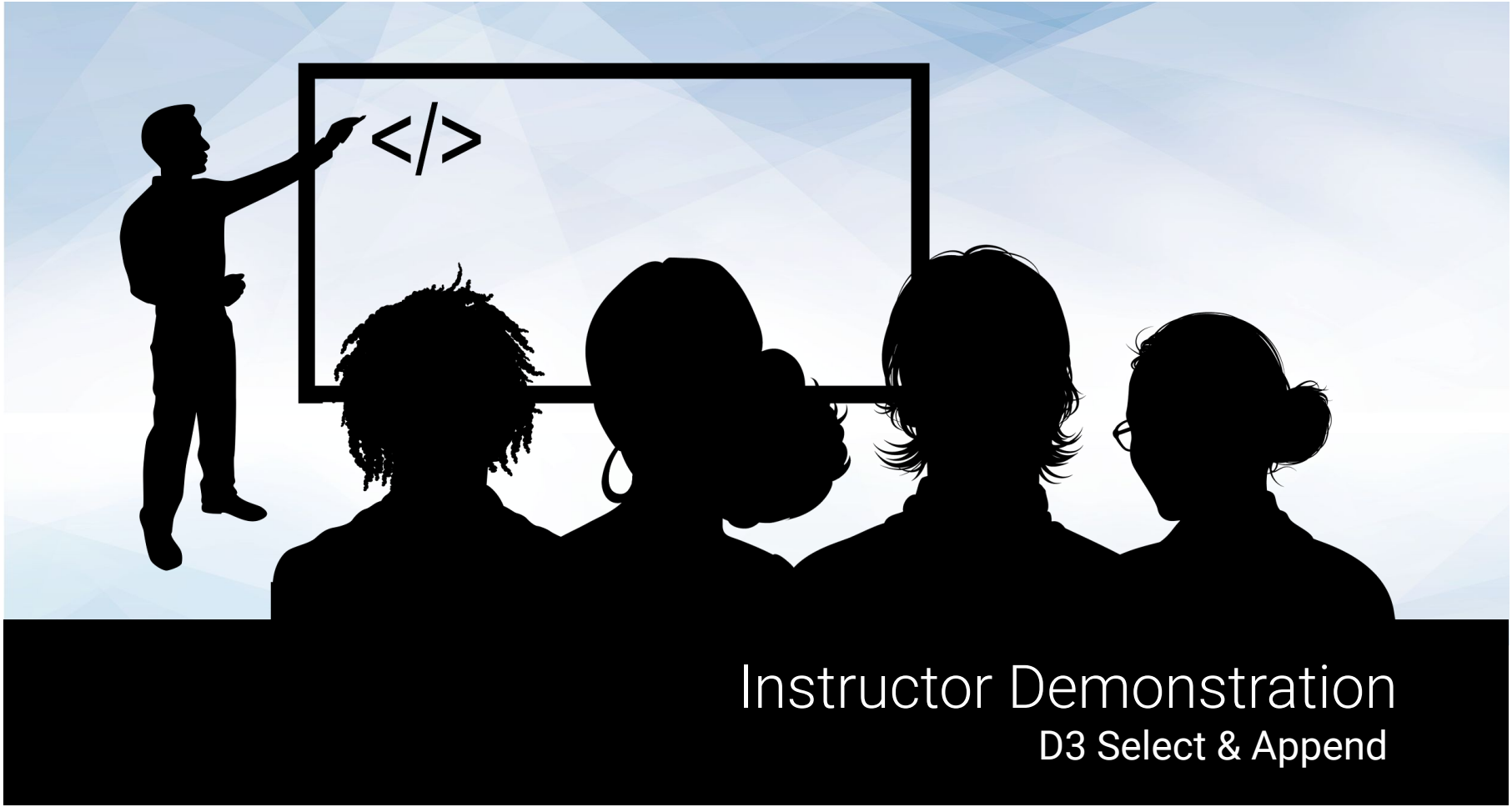
Instructions:

# Activity: D3.json

- Use d3.json to make an APi call and return the following:
  - Current information regarding the Roadster.
  - Information regarding all capsules.

- **Hint:**
  - Review the SpaceX API Docs. 

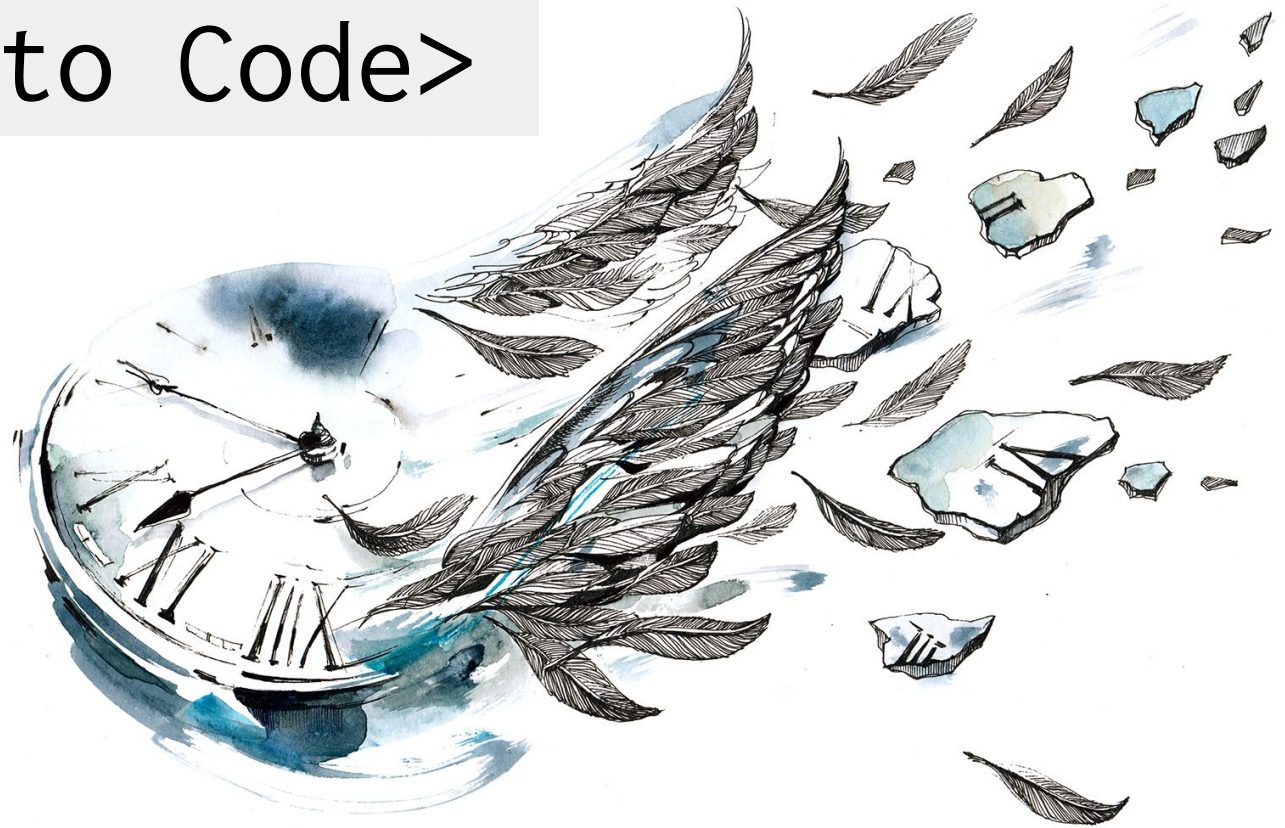

# Let's Review




# Instructor Demonstration

## D3 Select & Append

# <Time to Code>



# Instructor Do: D3 Select & Append

index.html

D3 link from d3js.org

<Time to Code>

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

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>D3 Select</title>
  <script src="https://d3js.org/d3.v5.min.js"></script>

</head>

<body>
  <h1>This is an H1</h1>
  <div class="text1">This div has a class</div>
  <div id="text2">This div has an id</div>
  <div class="my-link">
    <a href="https://github.com/d3/d3-selection">D3 Home</a>
  </div>
```

HTML

Creates a reference to DOM element with the class `text1`

Captures the text of that element

```js
// Select the text of an HTML element
var text1 = d3.select(".text1").text();
console.log("text1 says: ", text1);

var text2 = d3.select("#text2").text();
console.log("text2 says: ", text2);

// Modify the text of an HTML element
d3.select(".text1").text("Hey, I changed this!");

// Capture the HTML of a selection
var myLink = d3.select(".my-link").html();
console.log("my-link: ", myLink);

// Select an element's child element
// An object is returned
var myLinkAnchor = d3.select(".my-link>a");
console.log(myLinkAnchor);

// // Capture the child element's href attribute
var myLinkAnchorAttribute = myLinkAnchor.attr("href");
console.log("myLinkAnchorAttribute: " + myLinkAnchorAttribute);

// Change an element's attribute
myLinkAnchor.attr("href", "https://python.org");

// Use chaining to join methods
d3.select(".my-link>a").attr("href", "https://nytimes.com").text("Now this is a link to the NYT!!");

// Select all list items, then change their font color
d3.selectAll("li").style("color", "blue");

// Create a new element
var li1 = d3.select("ul").append("li");
li1.text("A new item has been added!");

// Use chaining to create a new element and set its text
var li2 = d3.select("ul").append("li").text("Another new item!");
```

index.js

console

```
Elements  Console  Sources  Network  Performance  Memory
top       Filter   Default le
text1 says:  This div has a class
text2 says:  This div has an id
my-link:
    <a href="https://github.com/d3/d3-selection">D3 Home</a>
```

# Instructor Do: D3 Select & Append

<Time to Code>

index.html

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

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>D3 Select</title>
  <script src="https://d3js.org/d3.v5.min.js"></script>

</head>

<body>
  <h1>This is an H1</h1>
  <div class="text1">This div has a class</div>
  <div id="text2">This div has an id</div>
  <div class="my-link">
    <a href="https://github.com/d3/d3-selection">D3 Home</a>
  </div>

  <div class="deeplink">
    <div class="outer">
      <div class="inner">
        <a href="https://github.com/d3/d3-selection">D3 Select</a>
      </div>
    </div>
  </div>

  <ul>
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
  </ul>
</body>
<script src="static/js/index.js"></script>

</html>
```

```js
// Select the text of an HTML element
var text1 = d3.select(".text1").text();
console.log("text1 says: ", text1);

var text2 = d3.select("#text2").text();
console.log("text2 says: ", text2);

// Modify the text of an HTML element
d3.select(".text1").text("Hey, I changed this!");

// Capture the HTML of a selection
var myLink = d3.select(".my-link").html();
console.log("my-link: ", myLink);

// Select an element's child element
// An object is returned
var myLinkAnchor = d3.select(".my-link>a");
console.log(myLinkAnchor);

// // Capture the child element's href attribute
var myLinkAnchorAttribute = myLinkAnchor.attr("href");
console.log("myLinkAnchorAttribute: " + myLinkAnchorAttribute);

// Change an element's attribute
myLinkAnchor.attr("href", "https://python.org");

// Use chaining to join methods
d3.select(".my-link>a").attr("href", "https://nytimes.com").text("Now this is a link to the NYT!!");

// Select all list items, then change their font color
d3.selectAll("li").style("color", "blue");

// Create a new element
var li1 = d3.select("ul").append("li");
li1.text("A new item has been added!");

// Use chaining to create a new element and set its text
var li2 = d3.select("ul").append("li").text("Another new item!");
```

# Instructor Do: D3 Select & Append

<Time to Code>

index.html

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

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>D3 Select</title>
  <script src="https://d3js.org/d3.v5.min.js"></script>

</head>

<body>
  <h1>This is an H1</h1>
  <div class="text1">This div has a class</div>
  <div id="text2">This div has an id</div>
  <div class="my-link">
    <a href="https://github.com/d3/d3-selection">D3 Home</a>
  </div>

  <div class="deeplink">
    <div class="outer">
      <div class="inner">
        <a href="https://github.com/d3/d3-selection">D3 Select</a>
      </div>
    </div>
  </div>

  <ul>
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
  </ul>
</body>
<script src="static/js/index.js"></script>

</html>
```

```js
// Select the text of an HTML element
var text1 = d3.select(".text1").text();
console.log("text1 says: ", text1);

var text2 = d3.select("#text2").text();
console.log("text2 says: ", text2);

// Modify the text of an HTML element
d3.select(".text1").text("Hey, I changed this!");

// Capture the HTML of a selection
var myLink = d3.select(".my-link").html();
console.log("my-link: ", myLink);

// Select an element's child element
// An object is returned
var myLinkAnchor = d3.select(".my-link>a");
console.log(myLinkAnchor);

// // Capture the child element's href attribute
var myLinkAnchorAttribute = myLinkAnchor.attr("href");
console.log("myLinkAnchorAttribute: " + myLinkAnchorAttribute);

// Change an element's attribute
myLinkAnchor.attr("href", "https://python.org");

// Use chaining to join methods
d3.select(".my-link>a").attr("href", "https://nytimes.com").text("Now this is a link to the NYT!!");

// Select all list items, then change their font color
d3.selectAll("li").style("color", "blue");

// Create a new element
var li1 = d3.select("ul").append("li");
li1.text("A new item has been added!");

// Use chaining to create a new element and set its text
var li2 = d3.select("ul").append("li").text("Another new item!");
```

***Objected Returned***


***Attributes of the Element***

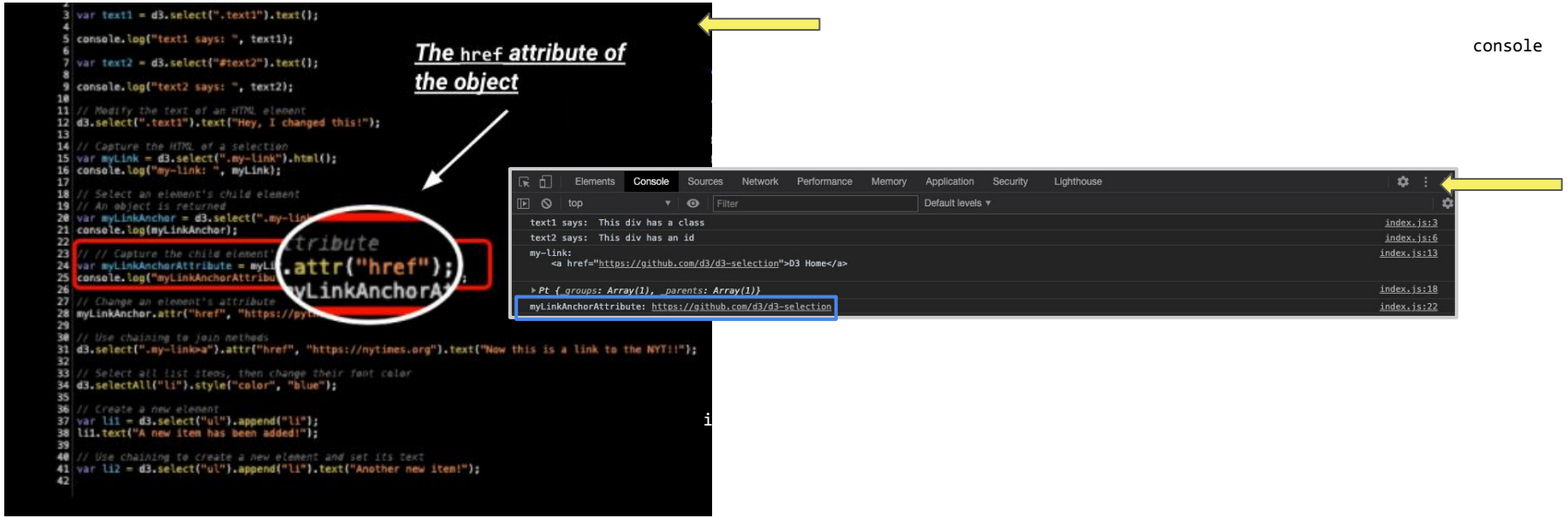
`href` ***attribute***

`href` ***attribute value***

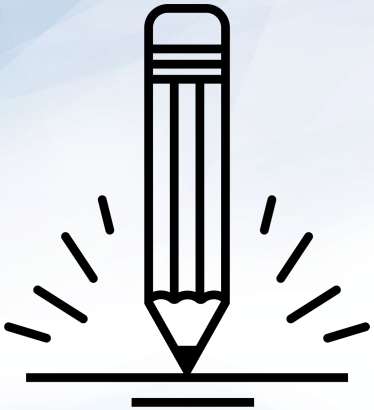
# Instructor Do: D3 Select & Append

index.html

<Time to Code>




console



# Activity: D3 Select

In this activity, you will use D3 to add a new row of data to a table.

**Suggested Time:**
15 Minutes

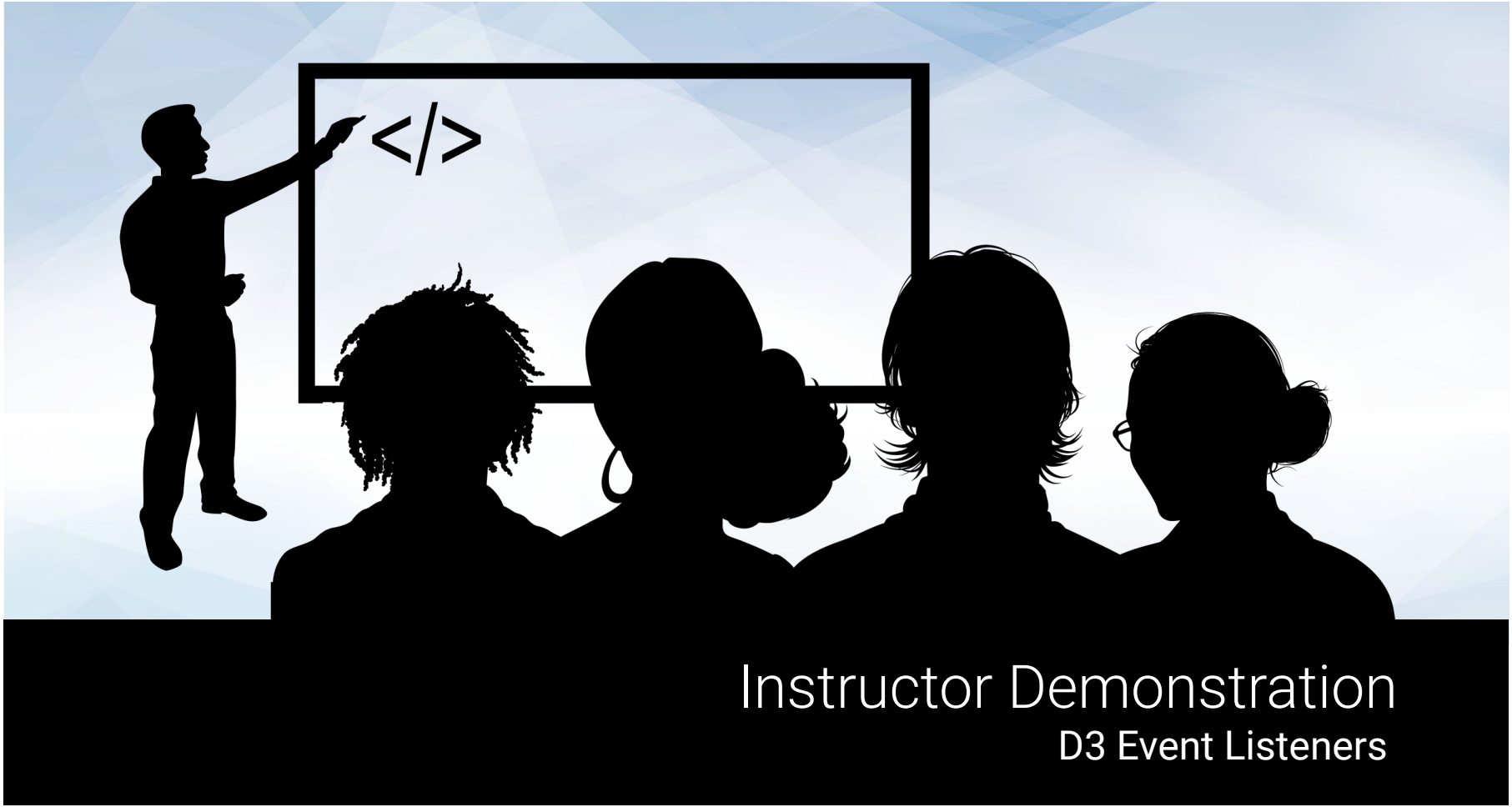
Instructions:

# Activity: D3 Select

- Use D3 to:
  - Convert the Bootstrap table into a striped table.
  - Select the table body and append a new row cells for the new student name and grade.

- **Hint:**
  - Review the Bootstrap Striped Tables. 


# Let's Review




# Instructor Demonstration

## D3 Event Listeners

# What is an Event?

- An action triggered by the user or the browser, detected by JavaScript (listen) to execute the code (interact HTML).
- There are several event types that are supported by the browser, including:
  - `click`
  - `change`
  - `keydown`
  - `scroll`
  - `pointenter`
  - `pointerleave`

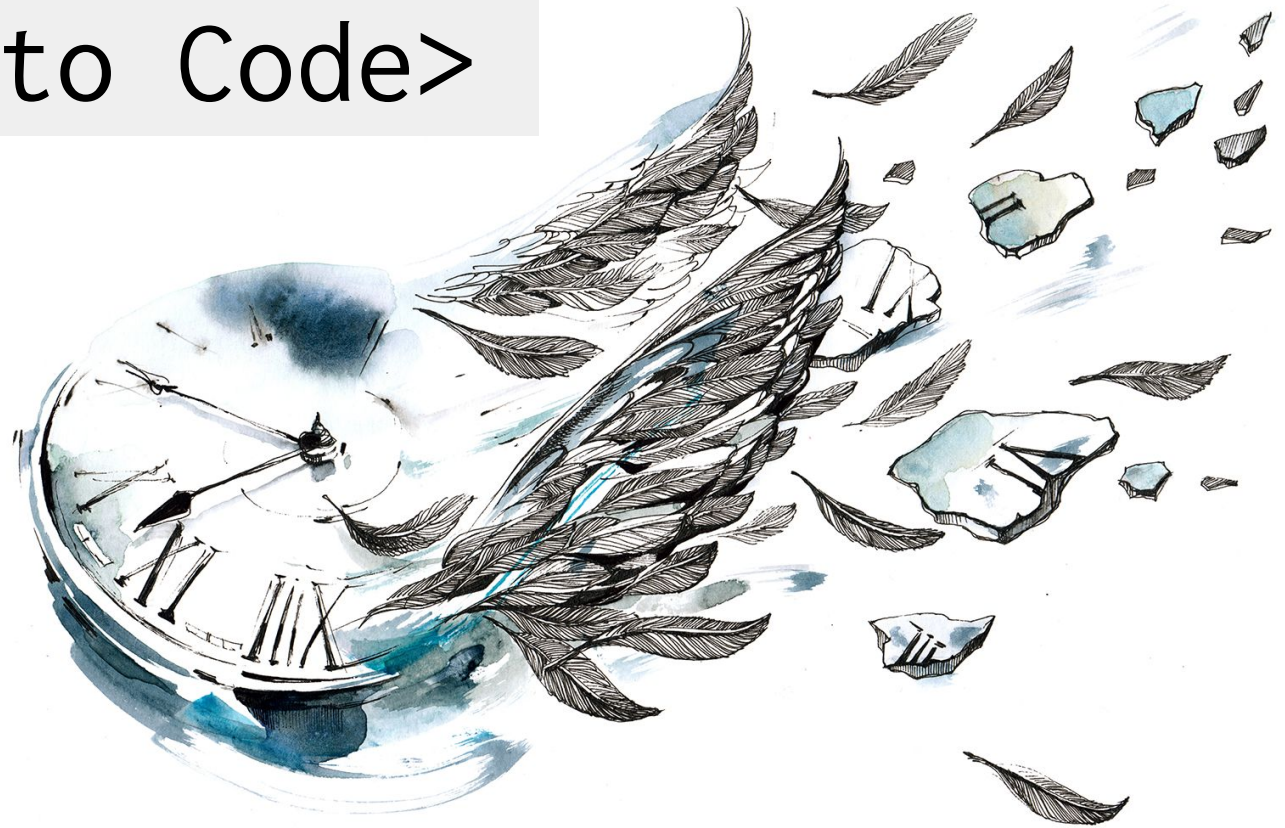
# Instructors Do: D3 Event Listeners

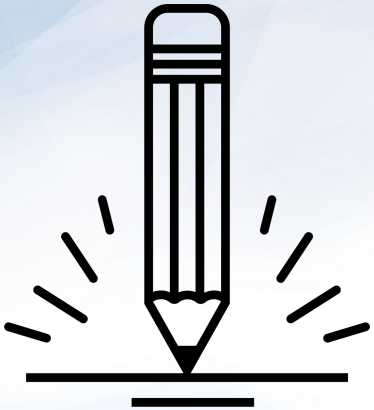
➔ Events have two main components:

```
function handleClick() {
  console.log("A button was clicked!");
  console.log(d3.event.target);
}
```

- A target: a reference to the object that dispatched the event.
- A handler: a function that executes in response to the event occurring.

# <Time to Code>





# Activity: Button Clicks

In this activity, you will use D3 to create click handlers for upvotes and downvotes.

**Suggested Time:**
15 Minutes

Instructions:

# Activity: Button Clicks

- Use D3 select `upvote` and `downvote` buttons on the page.
- Create click handlers for the upvote and downvote buttons.
- The click handlers should do the following:
  - Select the current vote count from the h3 tag.
  - Increment or decrement the count depending on which button was selected.
  - Update the vote count h3 tag using D3.


- **Bonus:**
  - Use an array to save information about each vote:
    - Store whether it was an “upvote” or “downvote”.
    - Store the current count at each click.
    - Use an array of arrays or array of objects to store data.

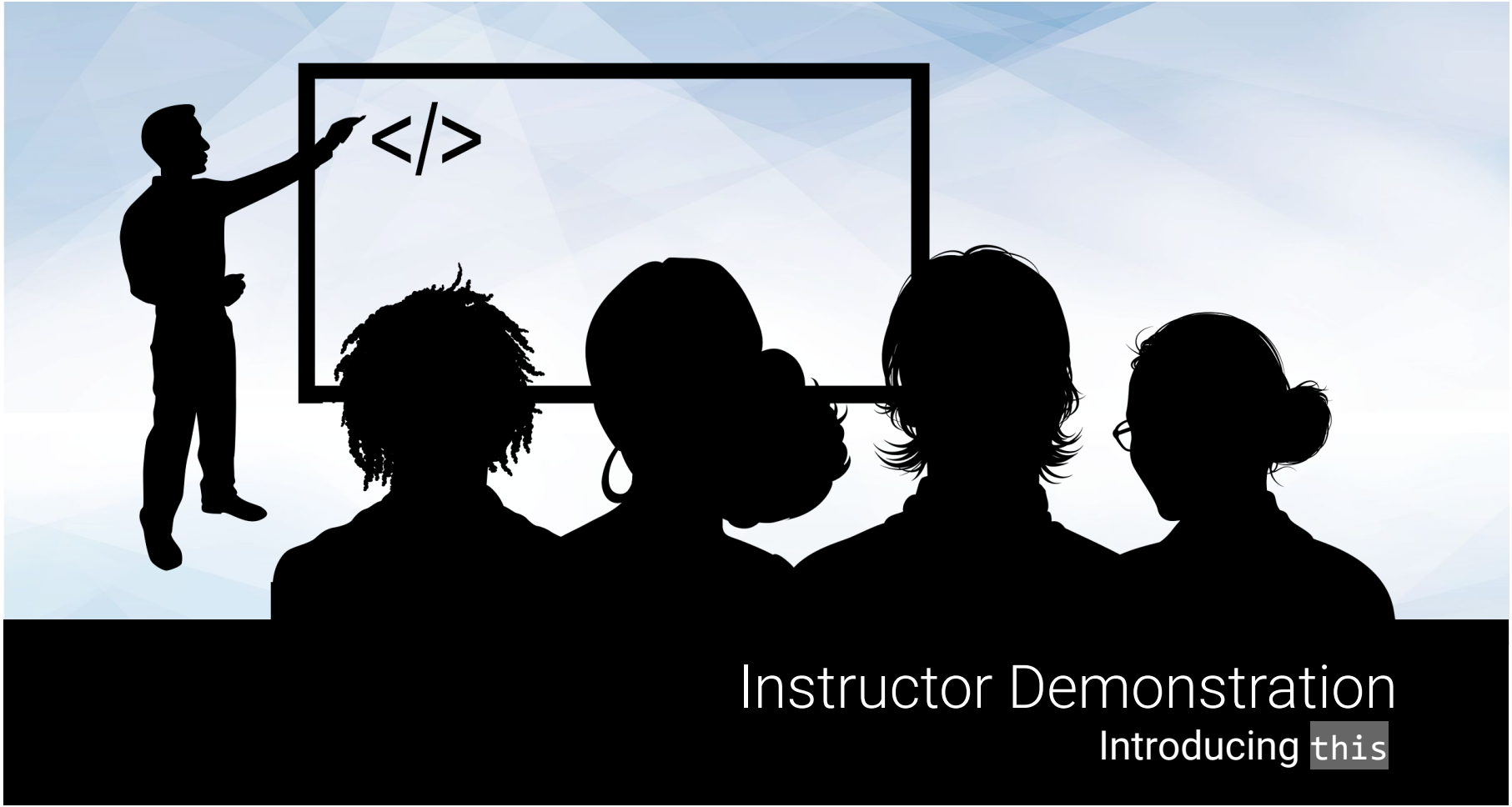
- **Hint:**
  - Don't forget to use the `.on` function to attach the click handlers to the buttons.
  - You will need one click handler for each button.
  - In Python, a number in string format can be converted into a numeric format using `int()`. In JavaScript, the same operation is performed with the `parseInt()` method. You will need to use `parseInt()` to convert the h3 vote count to a number before you can add or subtract from it.
  - Review the JavaScript `parseInt()` Documentation.


# Let's Review


Countdown timer

40:00

(with alarm)




# Instructor Demonstration

## Introducing `this`

# Instructor Do: Introducing `this`

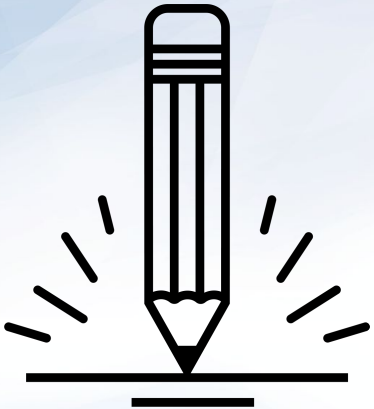
- In JavaScript the `this` keyword refers to the object it belongs to. It has different values depending on where it is used.
- It can be very resourceful to identify which element triggered an event.

```
d3.selectAll("button").on("click", function() {
  console.log(this);
});
```

Selects all the buttons in the document.
A function is triggered that will log `this` to the console.

```
d3.selectAll("li").on("click", function() {
 var listItem = d3.select(this);
 listItem.style("color", "blue");
 var listItemText = listItem.text();
 console.log(listItemText);
});
```

`li` element is assigned to the variable `listItem` via `d3.select(this)`. Selecting the element with D3 makes it possible to use D3 functions such as style or text on the element.



# Activity: `this` Button

In this activity, you will refactor the button activity with the `this` keyword.

**Suggested Time:**
15 Minutes

Instructions:

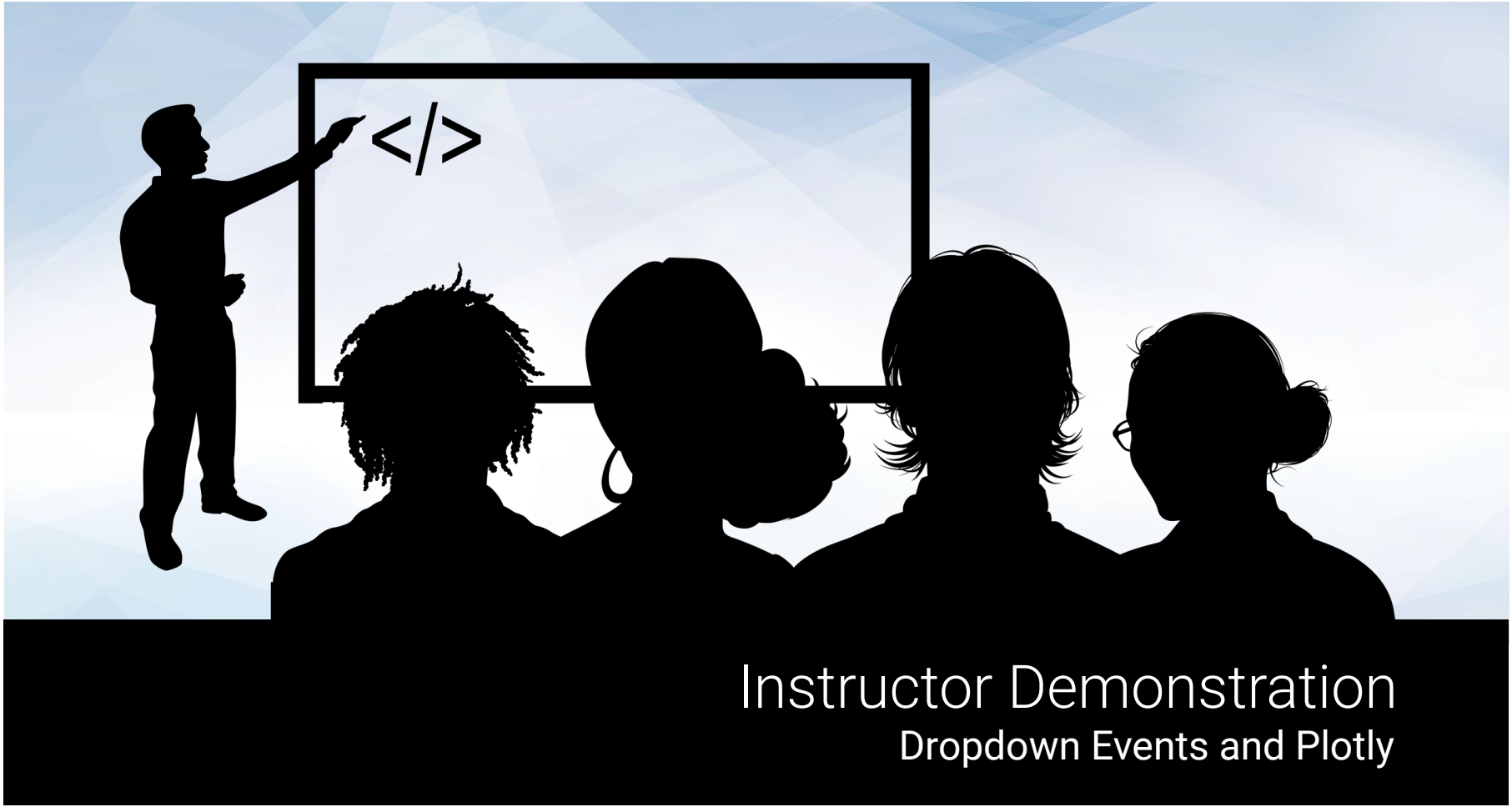
# Activity: `this` Button

- Use D3 select `upvote` and `downvote` buttons on the page.
- Create click handlers for the upvote and downvote buttons.
- The click handlers should do the following:
  - Select the current vote count from the h3 tag.
  - Increment or decrement the count depending on which button was selected.
  - Update the vote count h3 tag using D3.


- **Bonus:**
  - Use an array to save information about each vote:
    - Store whether it was an “upvote” or “downvote”.
    - Store the current count at each click.
    - Use an array of arrays or array of objects to store data.

- **Hint:**
  - Don't forget to use the `.on` function to attach the click handlers to the buttons.
  - You will need one click handler for each button.
  - You will need to use `parseInt()` to convert the h3 vote count to a number before you can add or subtract from it.

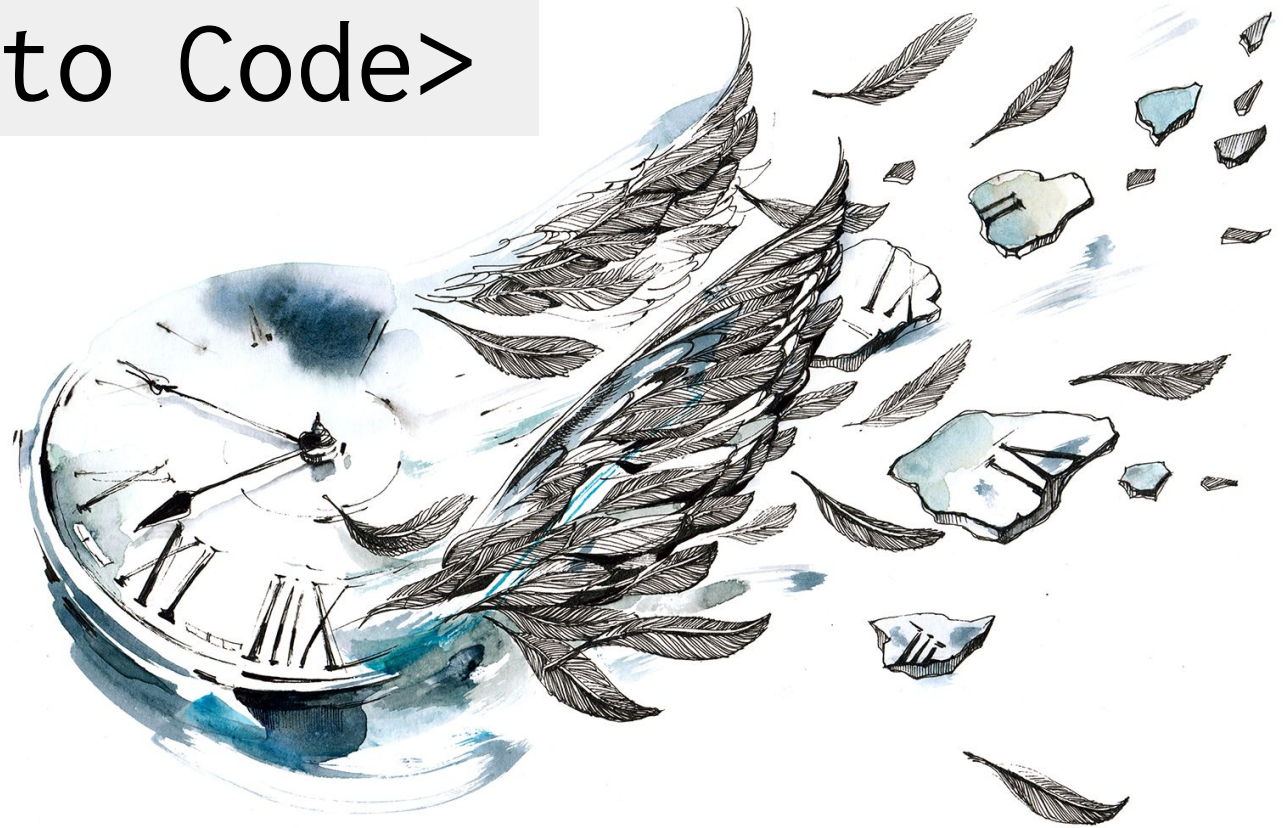

# Let's Review

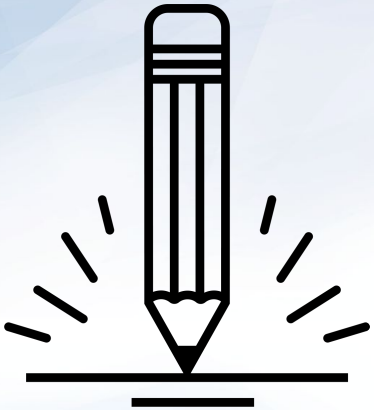



# Instructor Demonstration

## Dropdown Events and Plotly

# <Time to Code>





# Activity: A Musical Pie

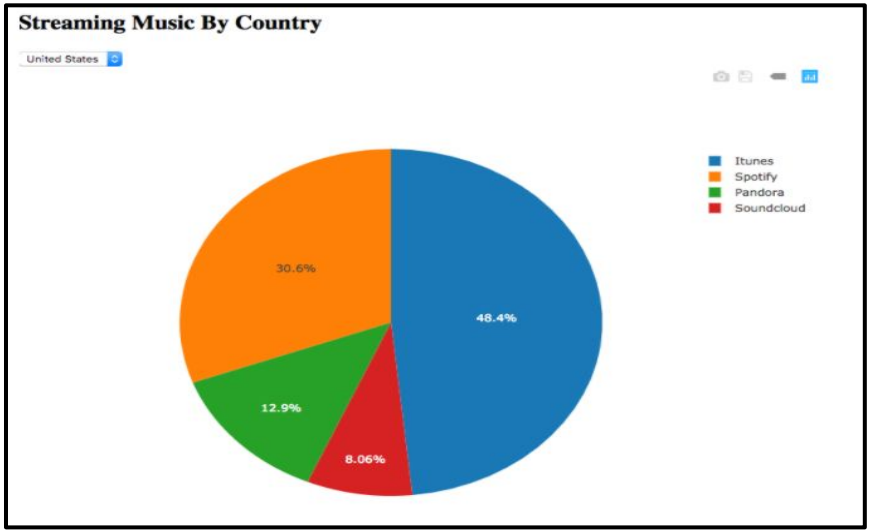
In this activity, you will create a dynamic pie chart using Plotly.

**Suggested Time:**
15 Minutes

Instructions:

# Activity: A Musical Pie

- Using the data in data.js, create a pie chart that that meets the following criteria:
  - Displays a default dataset.
  - Contains a dropdown menu listing 3 countries: United States, United Kingdom, and Canada.
  - When the dropdown menu selection changes, the chart should be restyled to reflect the new data.
- See the following image for reference.




- **Hint:**
  - Log the provided variables to the console to determine their use in creating the trace object.


# Let's Review