JavaScript with D3.js

Class Objectives

By the end of today's class you will be able to:

Use D3 for basic DOM manipulation.

Populate a table using static data structure.

Understand events.

Use `this` to reference elements.

Use D3 to attach events to DOM elements.
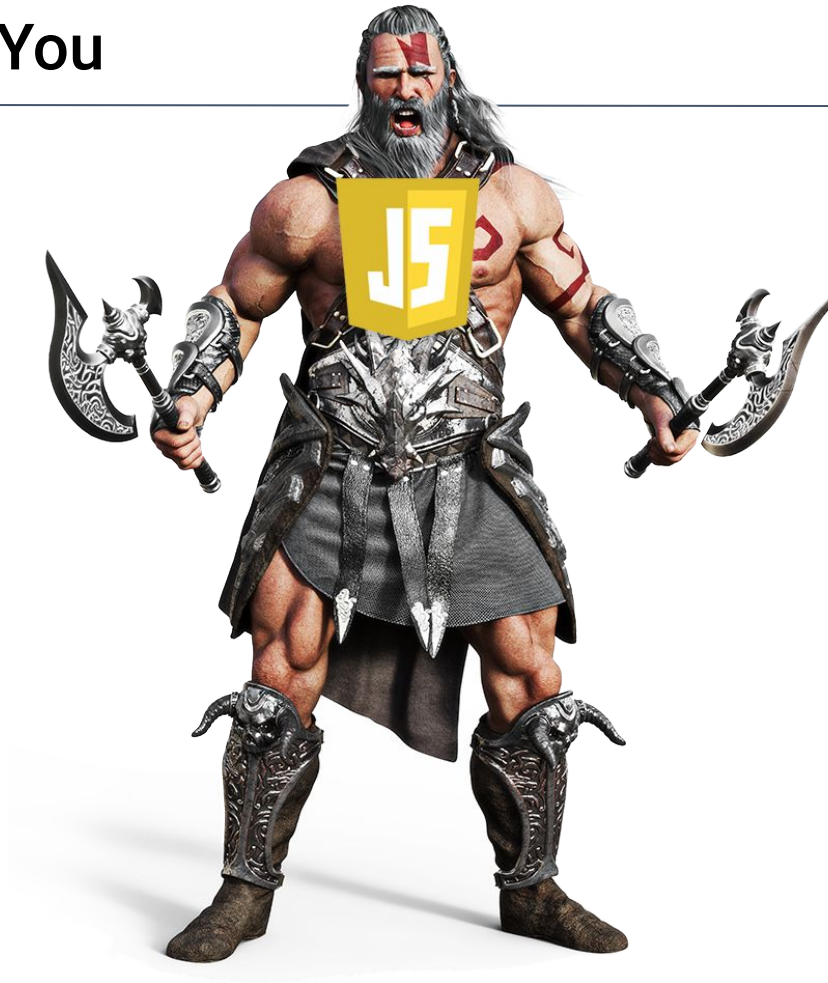
Dynamically manipulate the DOM through events.
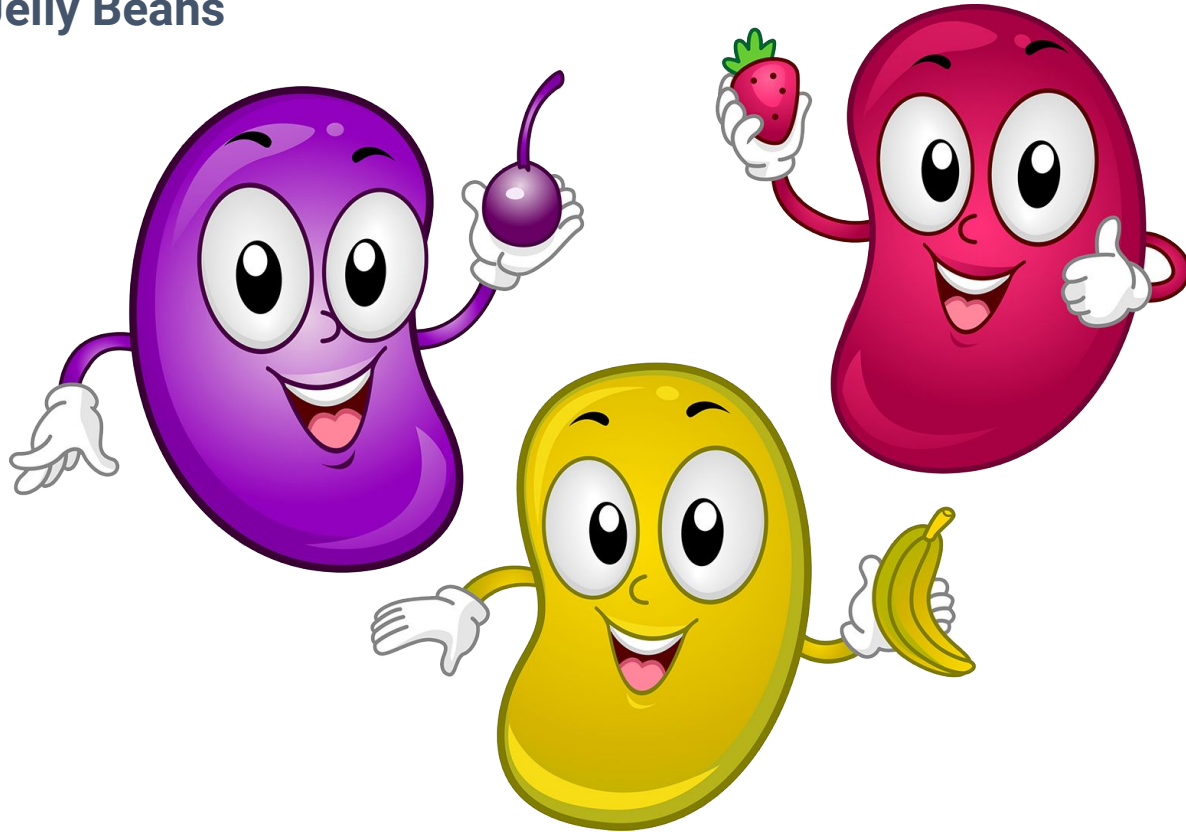
Dynamically filter tables.

This Will Soon Be You

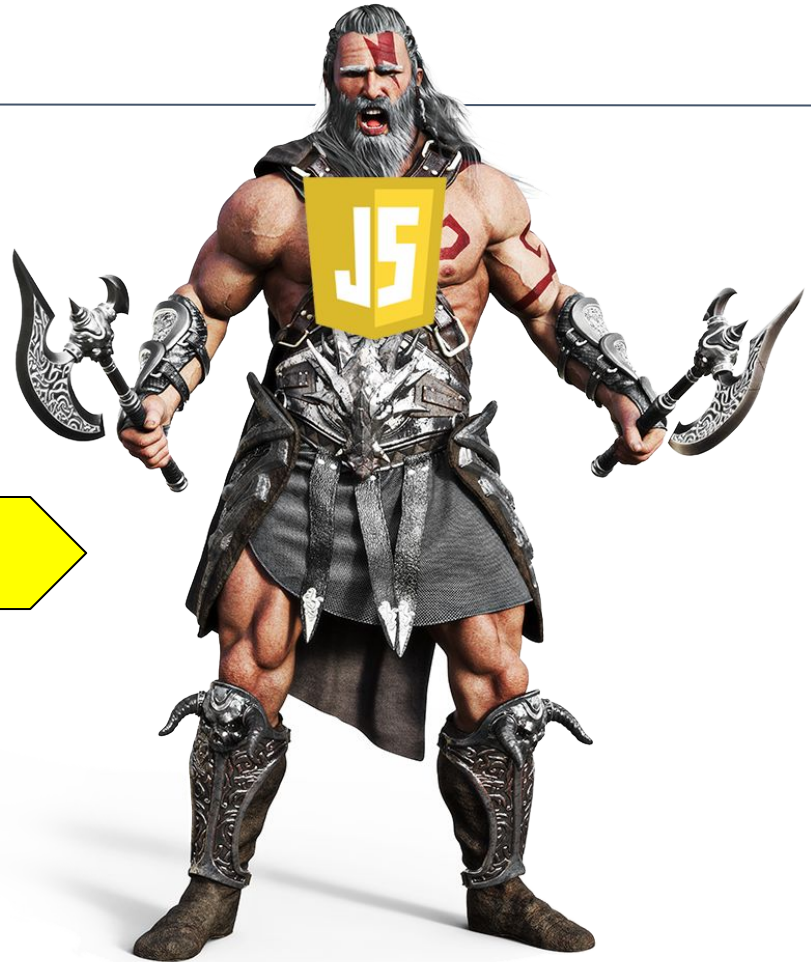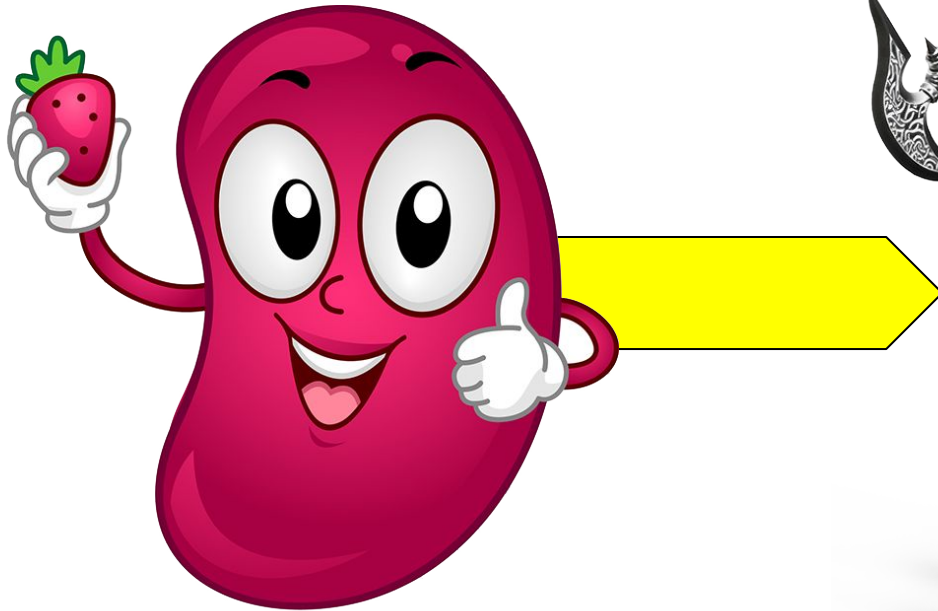**JavaScript Juggernauts**

But Right Now You Feel Like:

JavaScript Jelly Beans

Transformation to Come!

Hang in there!

Everyone Do: Intro to D3 and Select

In this activity, everyone will be introduced and kick off with **D3** basics.

Suggested Time:
25 Minutes

Everyone Do: Intro to D3 and Select

Everyone Do: Intro to D3 and Select

```
index.js    index.html ×                                          index.js* ×

1  <!DOCTYPE html>                                     1  // Select the text of an HTML element
2  <html lang="en">                                    2
3                                                       3  var text1 = d3.select(".text1").text();
4  <head>                                               4
5    <meta charset="UTF-8">                             5  console.log("text1 says: ", text1);
6    <meta name="viewport" content="width=device-width, 6
       initial-scale=1.0">                              7  var text2 = d3.select("#text2").text();
7    <meta http-equiv="X-UA-Compatible" content="ie=edge"> 8
8    <title>D3 Select</title>                           9  console.log("text2 says: ", text2);
9    <script src="https://d3js.org/d3.v5.min.js"></script> 10
10                                                      11  // Modify the text of an HTML element
11 </head>                                              12  d3.select(".text1").text("Hey, I changed this!");
12                                                      13
13 <body>                                               14  // Capture the HTML of a selection
14   <h1>This is an H1</h1>                             15  var myLink = d3.select(".my-link").html();
15   <div class="text1">This div has a class</div>     16  console.log("my-link: ", myLink);
16   <div id="text2">This div has an id</div>          17
17   <div class="my-link">                             18  // Select an element's child element
18     <a href="https://github.com/d3/d3-selection">D3 Home</a> 19  // An object is returned
19   </div>                                             20  var myLinkAnchor = d3.select(".my-link>a");
20                                                      21  console.log(myLinkAnchor);
21   <div class="deeplink">                            22
22     <div class="outer">                             23  // // Capture the child element's href attribute
23       <div class="inner">                           24  var myLinkAnchorAttribute = myLinkAnchor.attr("href");
24         <a href="https://github.com/d3/d3-selection">D3 Select</a> 25  console.log("myLinkAnchorAttribute: " + myLinkAnchorAttribute);
25       </div>                                         26
26     </div>                                           27  // Change an element's attribute
27   </div>                                             28  myLinkAnchor.attr("href", "https://python.org");
28                                                      29
29   <ul>                                               30  // Use chaining to join methods
30     <li>Item 1</li>                                  31  d3.select(".my-link>a").attr("href", "https://nytimes.org").text("Now this is a link to the NYT!!");
31     <li>Item 2</li>                                  32
32     <li>Item 3</li>                                  33  // Select all list items, then change their font color
33   </ul>                                              34  d3.selectAll("li").style("color", "blue");
34 </body>                                              35
35 <script src="static/js/index.js"></script>          36  // Create a new element
36                                                      37  var li1 = d3.select("ul").append("li");
37 </html>                                              38  li1.text("A new item has been added!");
38                                                      39
                                                        40  // Use chaining to create a new element and set its text
                                                        41  var li2 = d3.select("ul").append("li").text("Another new item!");
                                                        42
```

Everyone Do: Intro to D3 and Select


```
index.js        index.html ×
1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5    <meta charset="UTF-8">
6    <meta name="viewport" content="width=device-width, initial-scale=1.0">
7    <meta http-equiv="X-UA-Compatible" content="ie=edge">
8    <title>D3 Select</title>
9    <script src="https://d3js.org/d3.v5.min.js"></script>
10
11 </head>
12
13 <body>
14   <h1>This is an H1</h1>
15   <div class="text1">This div has a class</div>
16   <div id="text2">This div has an id</div>
17   <div class="my-link">
18     <a href="https://github.com/d3/d3-selection">D3 Home</a>
19   </div>
20
21   <div class="deeplink">
22     <div class="outer">
23       <div class="inner">
24         <a href="https://github.com/d3/d3-selection">D3 Select</a>
25       </div>
26     </div>
27   </div>
28
29   <ul>
30     <li>Item 1</li>
31     <li>Item 2</li>
32     <li>Item 3</li>
33   </ul>
34 </body>
35 <script src="static/js/index.js"></script>
36
37 </html>
38
```

```
index.js* ×
1  // Select the text of an HTML element
2
3  var text1 = d3.select(".text1").text();
4
5  console.log("text1 says: ", text1);
6
7  var text2 = d3.select("#text2").text();
8
9  console.log("text2 says: ", text2);
10
11 // Modify the text of an HTML element
12 d3.select(".text1").text("Hey, I changed this!");
13
14 // Capture the HTML of a selection
15 var myLink = d3.select(".my-link").html();
16 console.log("my-link: ", myLink);
17
18 // Select an element's child element
19 // An object is returned
20 var myLinkAnchor = d3.select(".my-link>a");
21 console.log(myLinkAnchor);
22
23 // // Capture the child element's href attribute
24 var myLinkAnchorAttribute = myLinkAnchor.attr("href");
25 console.log("myLinkAnchorAttribute: " + myLinkAnchorAttribute);
26
27 // Change an element's attribute
28 myLinkAnchor.attr("href", "https://python.org");
29
30 // Use chaining to join methods
31 d3.select(".my-link>a").attr("href", "https://nytimes.org").text("Now this is a link to the NYT!!");
32
33 // Select all list items, then change their font color
34 d3.selectAll("li").style("color", "blue");
35
36 // Create a new element
37 var li1 = d3.select("ul").append("li");
38 li1.text("A new item has been added!");
39
40 // Use chaining to create a new element and set its text
41 var li2 = d3.select("ul").append("li").text("Another new item!");
42
```

Objected Returned →

console

Attributes of the Element

___href attribute___

___href attribute value___ →
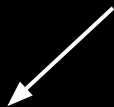
Everyone Do: Intro to D3 and Select


```
index.js* ×

1  // Select the text of an HTML element
2
3  var text1 = d3.select(".text1").text();
4
5  console.log("text1 says: ", text1);
6
7  var text2 = d3.select("#text2").text();
8
9  console.log("text2 says: ", text2);
10
11  // Modify the text of an HTML element
12  d3.select(".text1").text("Hey, I changed this!");
13
14  // Capture the HTML of a selection
15  var myLink = d3.select(".my-link").html();
16  console.log("my-link: ", myLink);
17
18  // Select an element's child element
19  // An object is returned
20  var myLinkAnchor = d3.select(".my-link
21  console.log(myLinkAnchor);
22
23  // // Capture the child element
24  var myLinkAnchorAttribute = myLi
25  console.log("myLinkAnchorAttribu
26
27  // Change an element's attribute
28  myLinkAnchor.attr("href", "https://pyth
29
30  // Use chaining to join methods
31  d3.select(".my-link>a").attr("href", "https://nytimes.org").text("Now this is a link to the NYT!!");
32
33  // Select all list items, then change their font color
34  d3.selectAll("li").style("color", "blue");
35
36  // Create a new element
37  var li1 = d3.select("ul").append("li");
38  li1.text("A new item has been added!");
39
40  // Use chaining to create a new element and set its text
41  var li2 = d3.select("ul").append("li").text("Another new item!");
42
```

The `href` attribute of the object

```
.attr("href");
```

Elements   **Console**   Sources   Network   Performance   Memory   Application   Security   Lighthouse

top ▼   Filter   Default levels ▼

text1 says: This div has a class   index.js:3
text2 says: This div has an id   index.js:6
my-link: index.js:13
    D3 Home

▼Pt {_groups: Array(1), _parents: Array(1)} ⬛   index.js:18
    ▶ _groups: [Array(1)]
    ▶ _parents: [html]
    ▶ __proto__ : Object

myLinkAnchorAttribute: https://github.com/d3/d3-selection   index.js:22
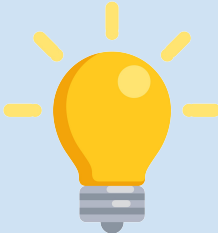▶
```

# Activity: D3 Select

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

**Suggested Time:**
20 Minutes

# Activity: D3 Select

| Instructions | Hints |
|---|---|
| Use D3 to convert the Bootstrap table into a striped table. | `.table-striped`<br><br>`<tbody>` |
| Use D3 to select the table body and append a new row and cells for the new student name and grade. | |

# Time's Up! Let's Review.

# Everyone Do: D3 Table

In this activity, everyone will use D3 to select data and build a raw table.

<Time to Code>

</>  **D3 Event Listeners**

Instructor Demonstration
D3 Event Listeners

# What is an Event?

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

# 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.

D3 Event Listeners

➜   In **D3,** Events are attached using the `.on()` function.

```
button.on("click", handleClick);
```

➜   Alternatively, the click handler can be defined inline.

```
button.on("click", function() {
  console.log("Hi, a button was clicked!");
  console.log(d3.event.target);
});
```

D3 Event Listeners

➔   Handlers are simply functions; they can execute code or call other functions.

```
button.on("click", function() {
 d3.select(".giphy-me").html("<img src='https://gph.to/2Krfn0w'alt='giphy'>");

});
```

➔   A *change* event can be triggered by input elements.

```
inputField.on("change", function() {

  var newText = d3.event.target.value;

  console.log(newText);

});
```

<Time to Code>

Activity: On Change

In this activity, you will use **D3** to reverse the input text and display it on the page.

Suggested Time:
20 Minutes

Activity: On Change

Instructions	Bonus	Hints
Use d3 to select the input (`#text`) and output (`.output`) elements from the page.	Instead of reversing the string, try to calculate the number of characters in the string.	You may need to iterate through the object using `Object.entries` and `forEach`.
Use d3 to attach an event listener to the input field. This event should call the `handleChange` function any time that the input text changes.		
Finally, complete the `handleChange` function to select the text from the input field and reverse the string. This function will use d3 to set the output element to the value of the reversed string.	Edit the `index.html` file to change the `h1` tag to an unordered list ul. Append each word: `count` as a `li` element.	

Time's Up! Let's Review.

Countdown timer

40:00

(with alarm)

Activity: Button Clicks

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

Activity: Button Clicks

- Use d3 to select the `upvote` and `downvote` buttons on the page.
- Create click handlers for both buttons which:
  - 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 an array of objects to store the data.

HINTS
- 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.

Time's Up! Let's Review.

Introducing `this` **& Forms**

Instructor Demonstration
Introducing `this` & Forms

Introducing `this` & Form

➜ 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.

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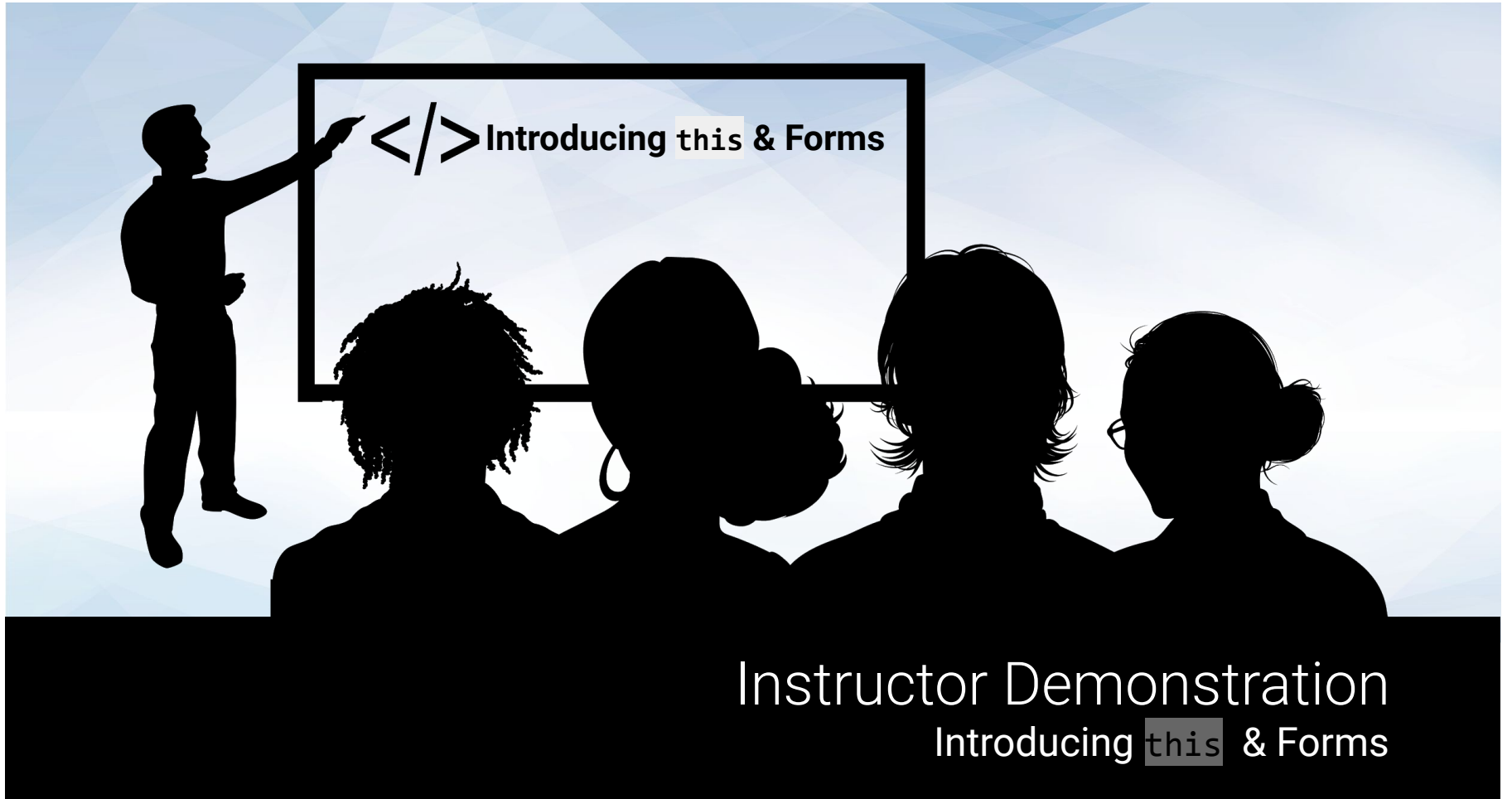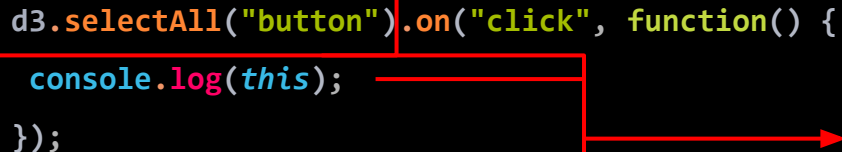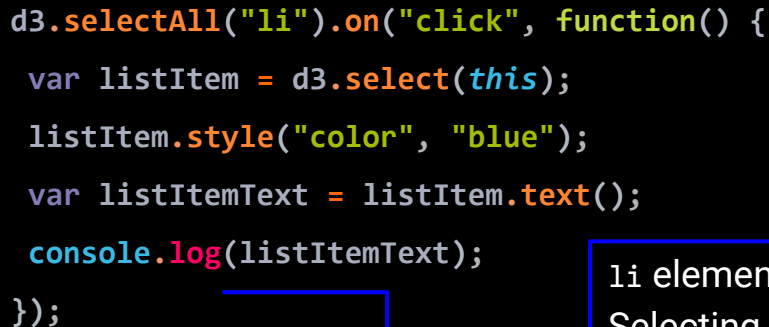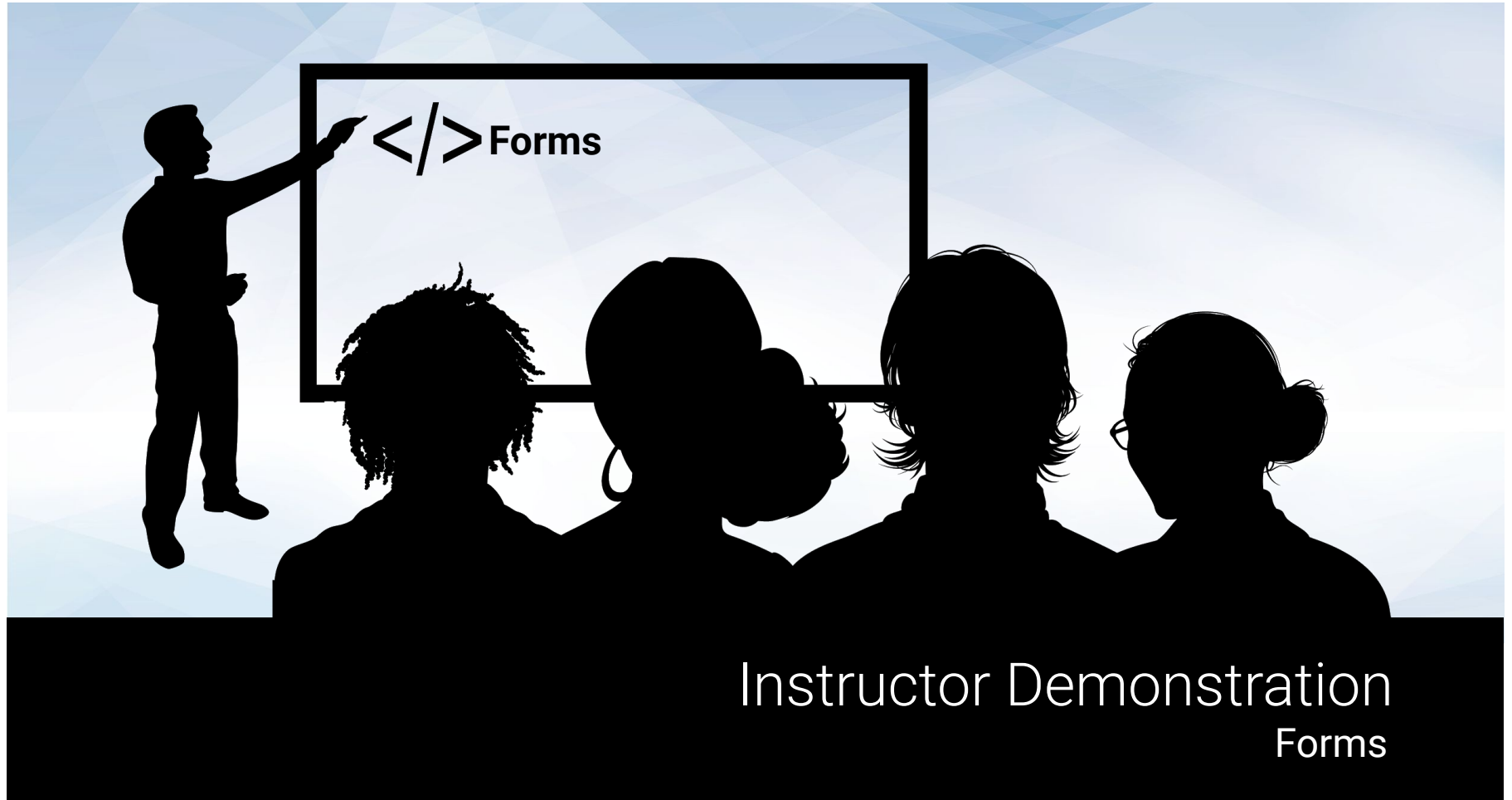d3.selectAll("button").on("click", function() {
    console.log(this);
  });
```

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

```javascript
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.

</>Forms

Instructor Demonstration
Forms

Forms

```javascript
var button = d3.select("#button");

var form = d3.select("#form");

button.on("click", runEnter);

form.on("submit",runEnter);


function runEnter() {
 d3.event.preventDefault();
 var inputElement = d3.select("#example-form-input");


 var inputValue = inputElement.property("value");
 console.log(inputValue);
 d3.select("h1>span").text(inputValue);
```

Activity: Form Filter

In this activity, you and your partner will use a form to filter data by blood type.

Suggested Time:
25 Minutes

Activity: Form Filter

Instructions
- Use d3 to add an event listener to the form button that does the following:
- Selects the value of the input element.
- Uses the form input to filter the data by bloodType.

Bonus
- Use the math.js library to calculate the mean, median, mode, variance, and standard deviation for the age field in the filtered data.
- Populate the unordered list `ul` with `li` elements for each summary statistic.

Hints
- Use `filter()` to filter the data by bloodType.
- Use `map()` to create an array of age values from the filtered data.
- Use the math.js library to calculate the summary statistics.

Time's Up! Let's Review.