GENERAL ASSEMBLY

JAVASCRIPT DEVELOPMENT

Sasha Vodnik, Instructor

HELLO!

1. Pull changes from the `svodnik/JS-SF-10-resources` repo to your computer and open the `07-dom-jquery > starter-code` folder in your code editor
2. Push your homework to the Homework repo and submit a pull request
3. **To submit your Slack bot project**, DM the URL of your Hubot repo on GitHub to Sasha

JAVASCRIPT DEVELOPMENT

Intro the the DOM & jQuery

LEARNING OBJECTIVES

At the end of this class, you will be able to

- Explain and use JavaScript methods for DOM manipulation.
- Create DOM event handlers to respond to user actions
- Manipulate the DOM by using jQuery selectors and functions.
- Create DOM event handlers using jQuery

AGENDA

- Intro the the DOM
- Getting and setting DOM elements
- Responding to events
- jQuery

WEEKLY OVERVIEW

WEEK 5	Intro the the DOM & jQuery / Advanced jQuery
WEEK 6	Ajax & APIs / Asynchronous JavaScript & Callbacks
WEEK 7	Advanced APIs / Project 2 lab

HOMEWORK — PUSH CODE TO GITHUB

EXERCISE

KEY OBJECTIVE

- Push your Hubot code to GitHub and submit it to the instructional team

TIMING

2 min

1. Create a new repo on github.com.
2. Use the 2 commands provided by GitHub to upload your existing Hubot repo to your new GitHub repo.
3. DM the URL of your new repo to Sasha and Aurielle.

HOMEWORK — GROUP DISCUSSION

EXERCISE

TYPE OF EXERCISE

- Groups of 3

TIMING

6 min

1. Show off your bot! What can it do?
2. Share a challenge you encountered, and how you overcame it.
3. If you tried something that didn't work, or wanted to add functionality but weren't quite sure how, brainstorm with your group how you might approach it.

HOMEWORK — GROUP DISCUSSION

EXERCISE

TYPE OF EXERCISE

- Groups of 3

TIMING

4 min

1. Share your solutions for the objects homework and for the JSON homework.
2. Share a challenge you encountered, and how you overcame it.
3. Share 1 thing you found challenging. If you worked it out, share how; if not, brainstorm with your group how you might approach it.

EXIT TICKET QUESTIONS

1. Why do we use Json? Could you give an example?
2. Do we need certain permissions to utilize API's or data from the internet in our applications?
3. When writing object in JS can we write it in JSON format?
4. Don't understand the DOM at all yet. But assume I will with more time.
5. Solo vs paired exercises

WARM-UP EXERCISE – DOM MANIPULATION

EXERCISE

KEY OBJECTIVE

- Identify web page features that respond to user actions or other events

TYPE OF EXERCISE

- Groups of 2-3

TIMING

2 min

1. On a website you use regularly, identify at least one thing that changes after the page loads (for instance, showing new data after you click, or updating itself on a set interval).
2. Demonstrate the change to your partner/group.

DOM MANIPULATION

REFERENCING A SCRIPT IN HTML

`script` element at the bottom of the `body` element

just before the closing `</body>` tag

```
<html>
  <head>
  </head>
  <body>
    <h1>JavaScript resources</h1>
    <script src="script.js"></script>
  </body>
</html>
```

Selecting an element in the DOM

- `getElementById()`
- `getElementsByClassName()`
- `getElementsByTagName()`
- `querySelector()`
- `querySelectorAll()`

Let us select DOM elements using CSS selector syntax

querySelector()

- Takes a single argument, a string containing CSS selector

HTML

```
<body>
  …
  <p id="main">Lorem ipsum</p>
  …
</body>
```

JavaScript

```
document.querySelector('#main');
```

querySelector()

- Selects the **first** DOM element that matches the specified CSS selector

```
<body>
  …
  <ul>
    <li>Lorem ipsum</li>
    <li>Lorem ipsum</li>
    <li>Lorem ipsum</li>
  </ul>
…
</body>
```

JavaScript

```
document.querySelector('li');
```

querySelectorAll()

- Takes a single argument, a string containing CSS selector
- Selects all DOM elements that match this CSS selector
- Returns a NodeList, which is similar to an array

```
<body>
  …
  <ul>
    <li>Lorem ipsum</li>
    <li>Lorem ipsum</li>
    <li>Lorem ipsum</li>
  </ul>
…
</body>
```

JavaScript

```
document.querySelectorAll('li');
```

What can we do with a selected element?

- Get and set its text content with the `innerHTML` property
- Get and set its attribute values by referencing them directly (`id`, `src`, etc.)

innerHTML

- Gets the existing content of an element, including any nested HTML tags
- Sets new content in an element

```
var item = document.querySelector('li');

console.log(item.innerHTML) // Gets value: "Lorem ipsum"

item.innerHTML = 'Apples' // Sets value: 'Apples'
```

`className` property

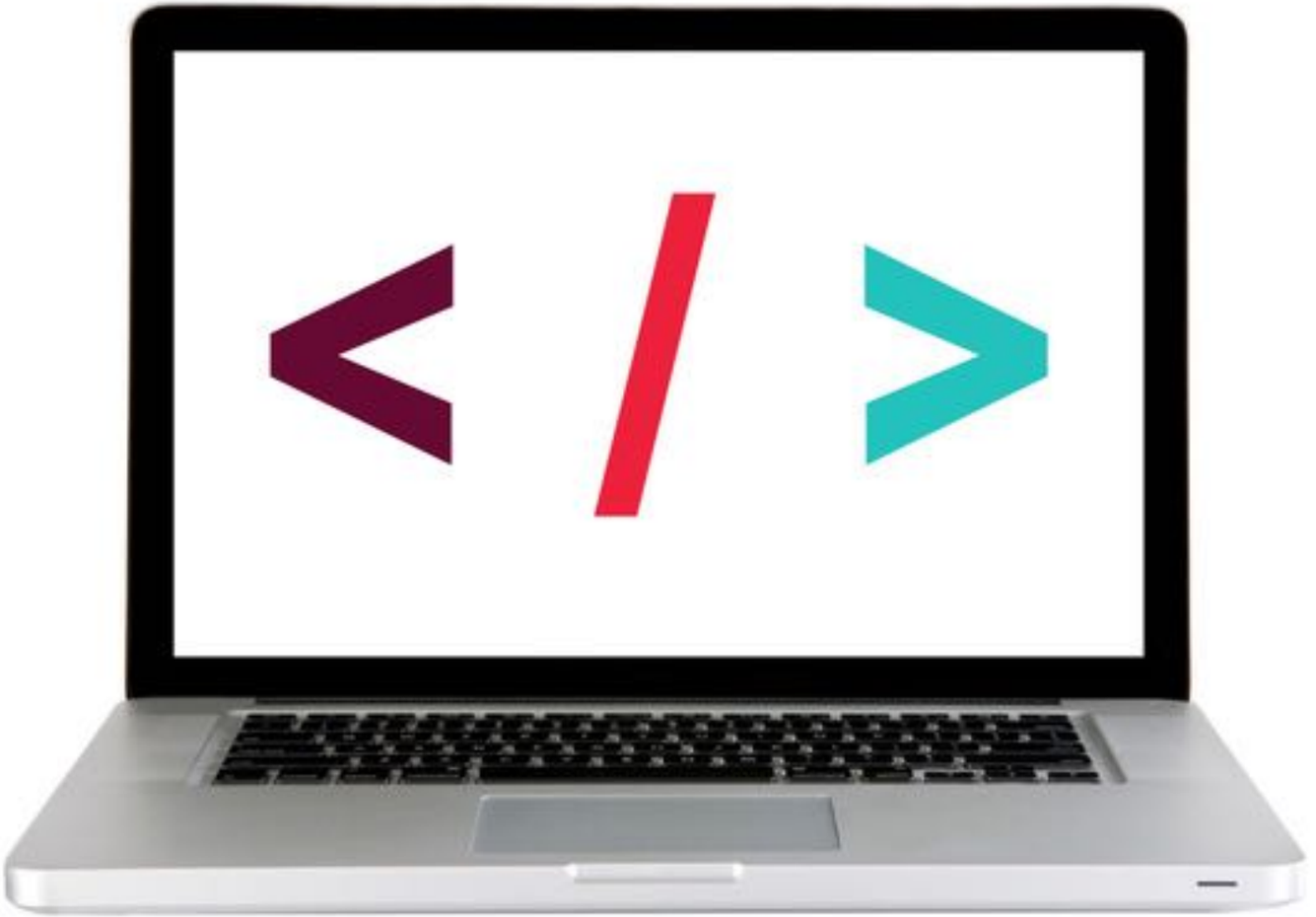
- Gets/sets an element's `class` attribute value
- CSS style sheet contains a style rule for each class
  - » Appearance of element changes based on which class is applied
  - » This is the best practice.

```
var item = document.querySelector('li');

console.log(item.className) // Gets value: 'default'

item.className = 'selected'
// Sets value: 'selected'
```

LET'S TAKE A LOOK



EXERCISE

LOCATION

▸ `starter-code > 1-dom-exercise`

TIMING

5 min

1. Open index.html in your editor, then scroll to the bottom.
2. Add a reference to the app.js file where indicated, then save your changes.
3. Open app.js in your editor, then follow the instructions.

EXERCISE

LOCATION

▸ `starter-code > 2-dom-attributes-exercise`

TIMING

5 min

1. Open app.js in your editor, then follow the instructions.

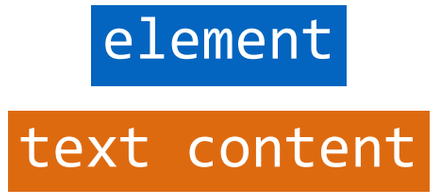
Adding content to the DOM

1. create a new element with `document.createElement()`

`element`

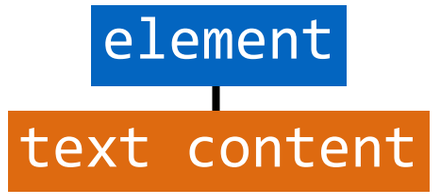
Adding content to the DOM

1. create a new element with `document.createElement()`
2. create new content for that element with `document.createTextNode()`



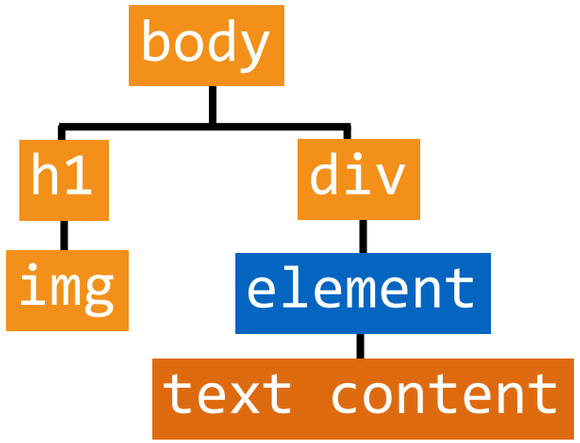
Adding content to the DOM

1. create a new element with `document.createElement()`
2. create new content for that element with `document.createTextNode()`
3. attach the new text content to the new element with `appendChild()`



Adding content to the DOM

1. create a new element with `document.createElement()`
2. create new content for that element with `document.createTextNode()`
3. attach the new text content to the new element with `appendChild()`
4. attach the new element to the DOM with `appendChild()`



createElement()

- Creates a new element

```
document.createElement('li'); // creates an li element
```

- Created element isn't attached to DOM
  - » assign variable when creating so you can reference later

```
let item1 = document.createElement('li');
let item2 = document.createElement('li');
```

createTextNode()

- Creates text content that can be added as the child of another element
- Created text node isn't attached to DOM
  - » assign variable when creating so you can reference later

```
let text1 = document.createTextNode('banana');
let text2 = document.createTextNode('apple');
```

appendChild()

- Attaches element or node as child of specified element
  - » Attaching to an element that's not part of the DOM creates/expands a **document fragment**
- Syntax:
  parent.appendChild(*child*);

```
item1.appendChild(text1);   // adds text1 text to item1 li
item2.appendChild(text2);   // adds text2 text to item2 li
```

appendChild() (continued)

- Attaches element or node as child of specified element
  - » Attaching to a DOM element makes it part of the DOM
- Syntax:
  parent.appendChild(*child*);

```
let list = document.querySelector('ul'); // selects ul element
list.appendChild(item1);     // adds item1 li to list ul
list.appendChild(item2);     // adds item2 li to list ul
```

EXERCISE

KEY OBJECTIVE

- Explain and use JavaScript methods for DOM manipulation.

TYPE OF EXERCISE

- Groups of 3-4

TIMING

2 min

1. Work together to create and complete a list of the four steps in DOM manipulation.
2. For each step in your list, add the method used.

EXERCISE – ADD CONTENT TO A WEB PAGE USING JAVASCRIPT

LOCATION

▸ `starter-code > 4-create-append—exercise`

TIMING

15 min

1. Open preview.png. Your task is to use DOM manipulation to build the sidebar shown in the image and add it to the blog.html web page.
2. Open app.js in your editor, then follow the instructions to create and the “About us” heading and the 2 paragraphs of text to the sidebar.
3. BONUS 1: Open preview-bonus.png, then write JavaScript code to add the image shown to the sidebar. (Filename and location in app.js.)
4. BONUS 2: Create and append the “Recent issues” heading and list.

EVENTS

EVENT LISTENERS

selecting element

```
let button = document.querySelector(‘.submitBtn’);
```

element reference

```
button.addEventListener('click', function() {
  // your code here
}, false);
```

EVENT LISTENERS

```
let button = document.querySelector(‘.submitBtn’);
```

method to add event listener

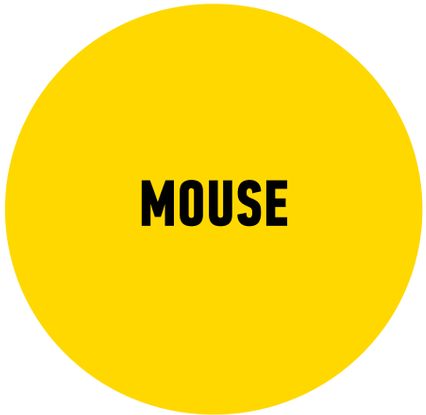
```
button.addEventListener('click', function() {
  // your code here
}, false);
```

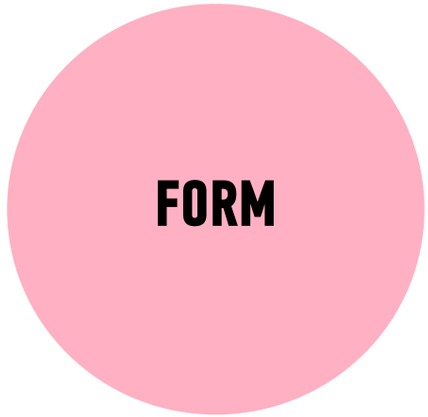
EVENT LISTENERS

```
let button = document.querySelector(‘.submitBtn’);
```

type of event

```
button.addEventListener('click', function() {
  // your code here
}, false);
```





click

dblclick

mouseenter

mouseleave

keypress

keydown

keyup

submit

change

focus

blur

resize

scroll

```
button.addEventListener('eventgoeshere', function() {
  // your code here
}, false);
```

EVENT LISTENERS

```
let button = document.querySelector(‘.submitBtn’);
```

```
button.addEventListener('click', function() {
  // your code here
}, false);
```

function to run when event is triggered

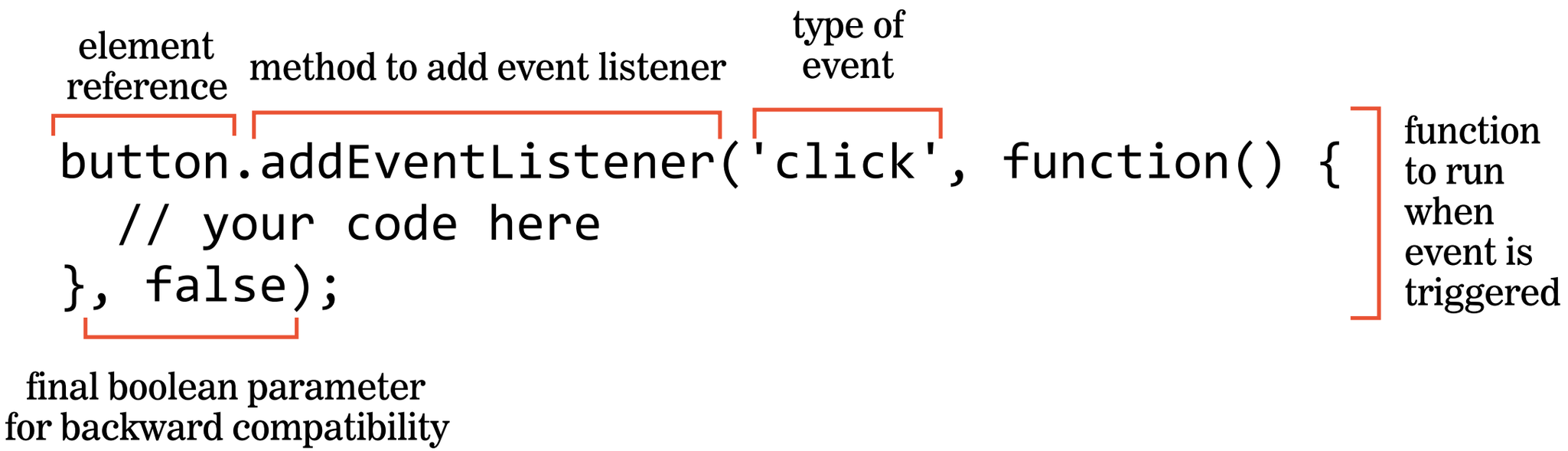
EVENT LISTENERS

```
let button = document.querySelector(‘.submitBtn’);
```

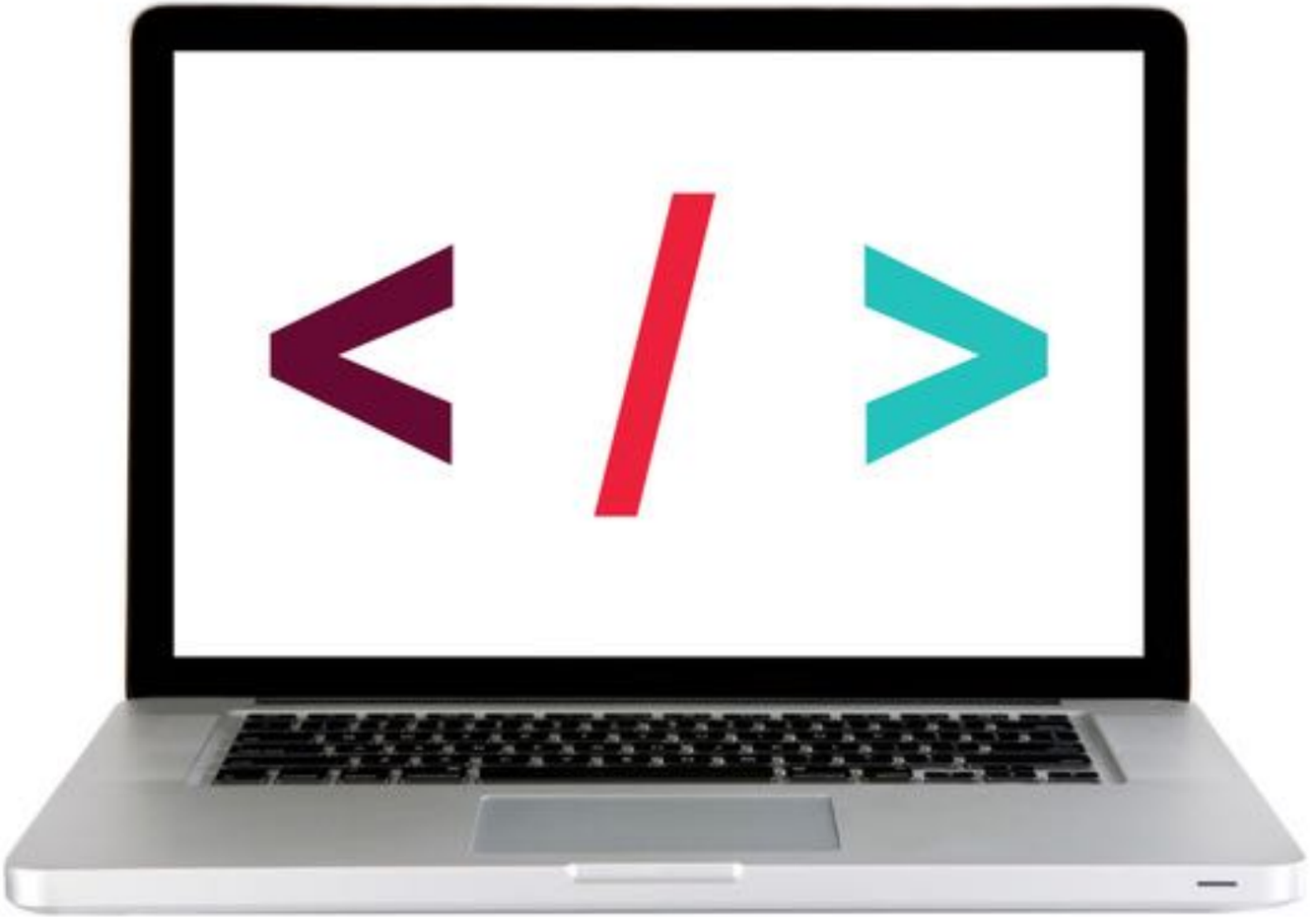
```
button.addEventListener('click', function() {
  // your code here
}, false);
```

final boolean parameter
for backward compatibility

EVENT LISTENERS



LET'S TAKE A LOOK



ACTIVITY

KEY OBJECTIVE

- Create DOM event handlers to respond to user actions

TYPE OF EXERCISE

- Individual/Partner

AS A CLASS

10 min

Exercise is in `6-events-exercise` folder

1. Add event listeners to the 3 buttons at the top of the page. Clicking each button should hide the block below it with the corresponding color.
2. Use cheat sheet/slides as a guide for syntax
3. BONUS: Add an event listener for the "Show all blocks" button that removes the hidden class from all the colored block elements.

`preventDefault()`

- Prevents element from executing default behavior in response to an event

Referencing an event

- An object containing information about the triggering event is passed to a function called in response to an event
- Specify a parameter to be able to reference this event in your code
  - » By convention, we use `event`, `evt`, or `e`

```
submitButton.onclick = function(event) {
  event.preventDefault();
  ...
}
```

EXERCISE

KEY OBJECTIVE

- Create DOM event handlers to respond to user actions

LOCATION

- `starter-code > 7-js-dom-exercise`

TIMING

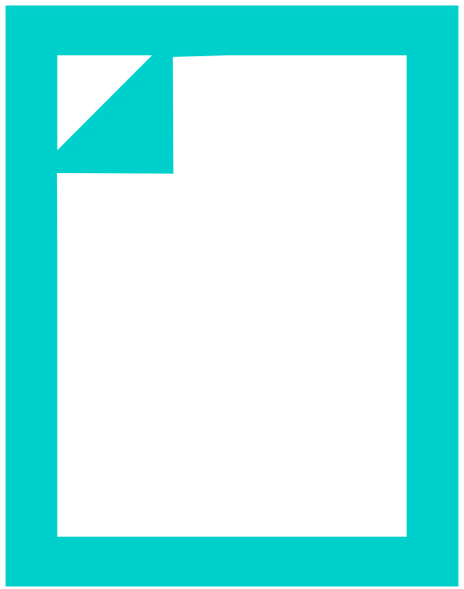
10 min

1. Open index.html in your browser.
2. Open main.js in your editor, then follow the instructions to make the submit button functional and use DOM manipulation to add items to the list.
3. BONUS: Add functionality that adds a message to the page to alert the user when they click Submit without typing anything. (Use DOM manipulation, not the `alert` method.)

JQUERY

INTRO TO JQUERY — YOUR NEW BEST FRIEND!

jQuery is a JavaScript library you include in your pages.



JQUERY VS. JAVASCRIPT

jQuery allows us to keep using the CSS-style selectors that we know and love — but more concisely! Yay!

JQUERY VS. JAVASCRIPT

jQuery statements for DOM manipulation are also more concise!

JS:

```
document.querySelector('#heading').innerHTML = "Your Name";
```

JQUERY:

```
$('#heading').text('Your Name');
```

You could do everything jQuery does with plain-old vanilla Javascript

JQUERY VS. JAVASCRIPT — A COMPARISON OF BENEFITS

JQUERY

- Write way less code to achieve the same tasks

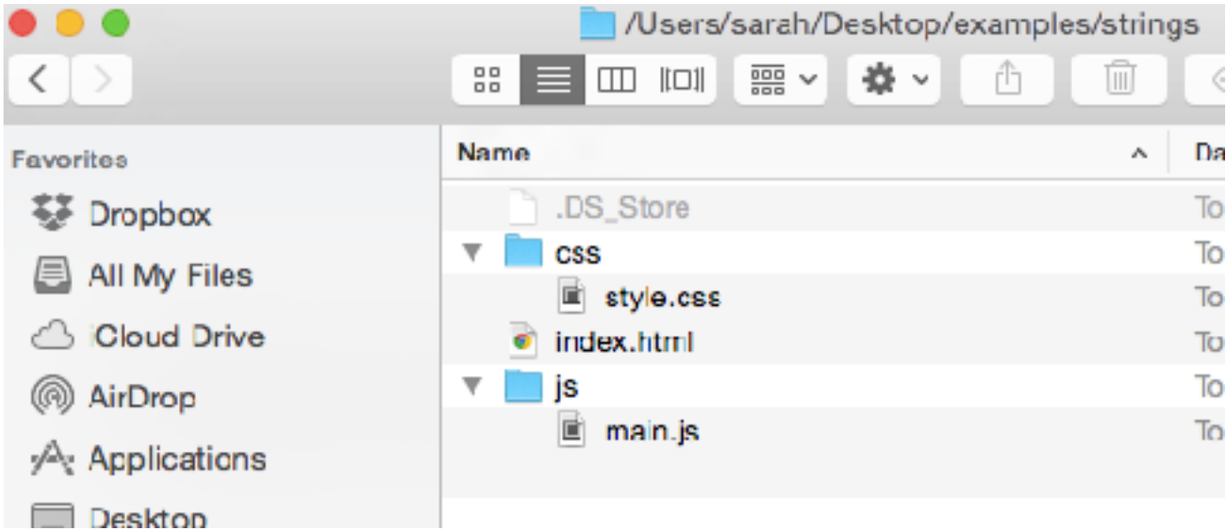
PURE JAVASCRIPT

- Better performance
- Faster

ADDING JQUERY TO YOUR PROJECT

KEEP IT ON THE UP AND UP!

- It is considered **best practice** to keep Javascript files organized in one folder.
- Usually people name this folder *scripts*, *js*, or *javascript*.



Remember - use an underscore or dash between words in folder names instead of a space. And try to avoid characters/symbols in file names (*really_cool_page.html* or *really-cool-page.html*).

STEP 1: ADD JQUERY TO YOUR WEBSITE

1. Download the jQuery script (version 3.x, compressed).
2. Add a `js` folder to your project
3. Move the jQuery file you downloaded to the `js` folder
4. Use a `<script>` tag to include the jQuery file after your HTML content and before any other JavaScript files that use it.

```
<body>
  <!-- HTML content here -->
  <script src="js/jquery-3.2.1.min.js"></script>
  <script src="js/main.js"></script>
</body>
```

STEP 2: ADD A JAVASCRIPT FILE

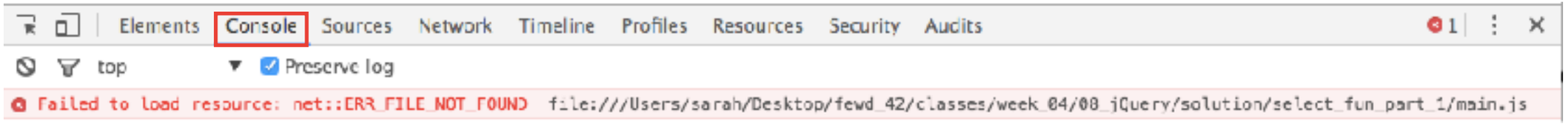
1. Create your custom JavaScript file with a .js extension (example: main.js)
2. Link to the JavaScript file from your HTML page using the `<script>` element. Add this **right before the closing `</body>` tag and after the `<script>` element for your jQuery file.**

```
<body>
  <!-- HTML content here -->
  <script src="js/jquery-3.2.1.min.js"></script>
  <script src="js/main.js"></script>
</body>
```

ORDER IS IMPORTANT!!!!

MAKE SURE YOUR JS IS HOOKED UP PROPERLY

- Open the page in Chrome, then open the console (command + option + J [Mac] or Ctrl + Alt + J [Win]) and make sure there are no errors.



This error means the file can't be found. Check your url in your <script> tag. Make sure the file exists.

PART 1 — SELECT AN ELEMENT

A JQUERY STATEMENT INVOLVES 2 PARTS

1

Select an element/elements

2

Work with those elements

INTRO TO JQUERY

1

Select an element/elements

2

Work with those elements

JQUERY — SELECTING ELEMENTS

JQUERY OBJECTS — FINDING ELEMENTS: SOME EXAMPLES

	CSS	JQUERY
ELEMENT	`a { color: blue; }`	`$('a')`
ID	`#special { color: blue; }`	`$('#special')`
CLASS	`.info { color: blue; }`	`$('.info')`
NESTED SELECTOR	`div span { color: blue; }`	`$('div span')`

```
<button id="form-submit">Submit</button>

<li class="circle">One</li>

<h1>Color Scheme Switcher</h1>
```

JQUERY OBJECTS

- Selecting elements with vanilla JavaScript returns an element reference (`querySelector`) or a collection of element references (`querySelectorAll`)

vanilla JavaScript selector

plain element reference

`querySelector('selector')`

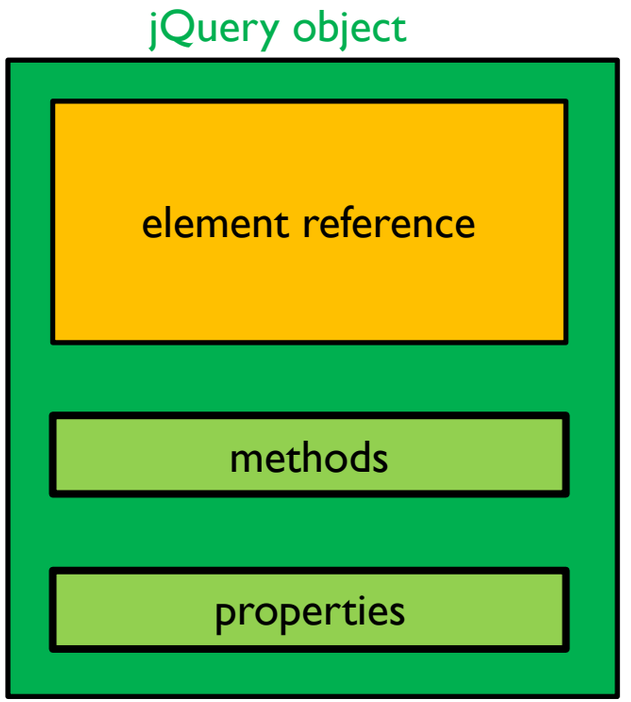
element reference

JQUERY OBJECTS

- Selecting elements with jQuery returns a **jQuery object**, which is one or more element references packaged with jQuery methods and properties

jQuery selector

```
$('selector')
```



NAMING VARIABLES WHEN USING JQUERY

- Best practice: include $ as the first character of any variable whose value is a jQuery object
- This is not required by jQuery, but helps us keep track of what parts of our code rely on the jQuery library

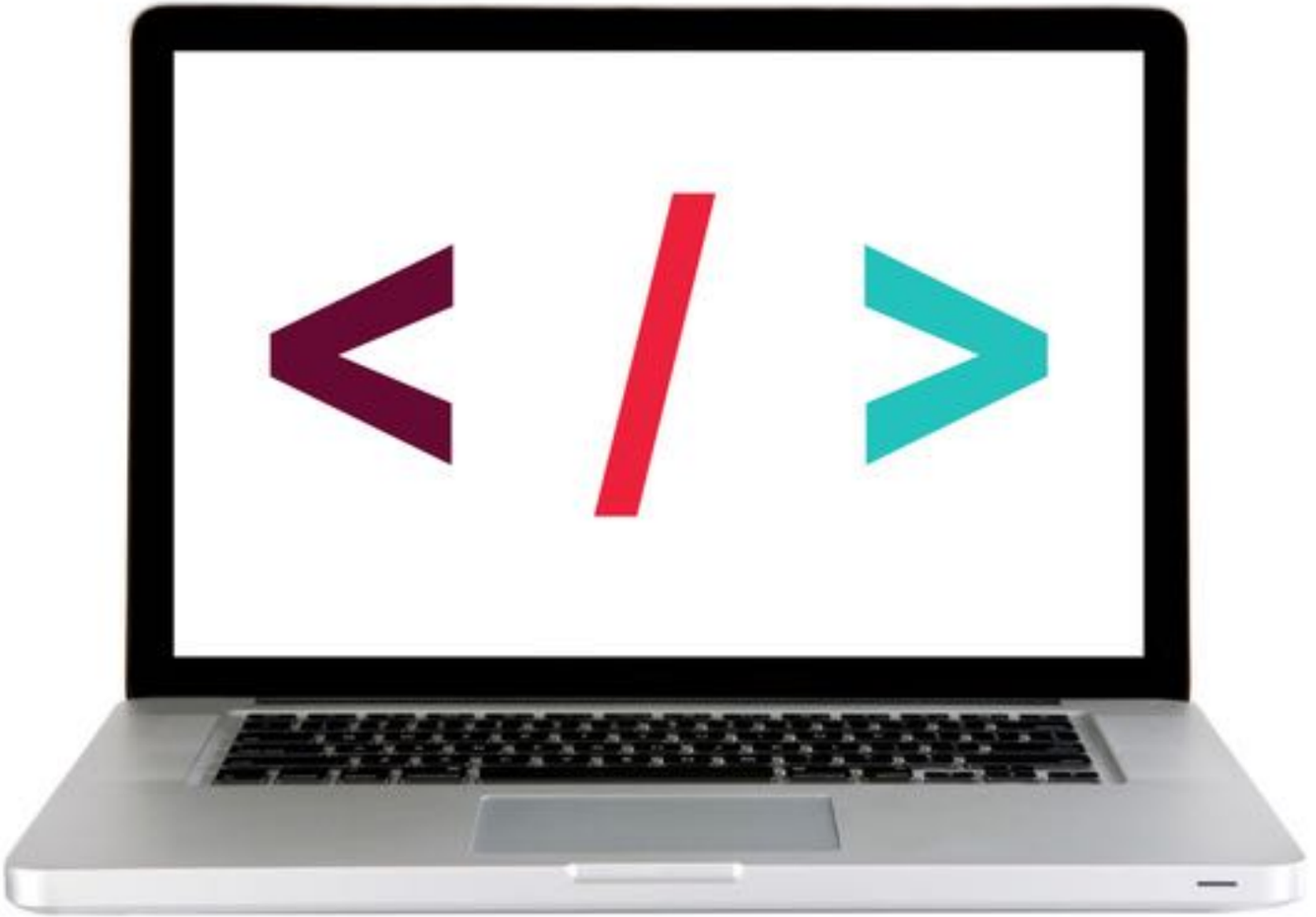
$ included at start of variable name to indicate that its value is a jQuery object

```
let $openTab = $('.open');
```

it's not an error to name the variable with out the $ — it just wouldn't give us as much information

```
let openTab = $('.open');
```

LET'S TAKE A CLOSER LOOK



PART 2 — ADD A METHOD

USING JQUERY TO MANIPULATE THE DOM

1

Select an element/elements

2

Work with those elements

JQUERY — WORKING WITH THOSE ELEMENTS

JQUERY METHODS

Be forewarned!

There are a lot of methods!

Do not feel like you need to sit down and memorize these. The important things is knowing that they're there and **being able to look them up** in the documentation.

api.jquery.com

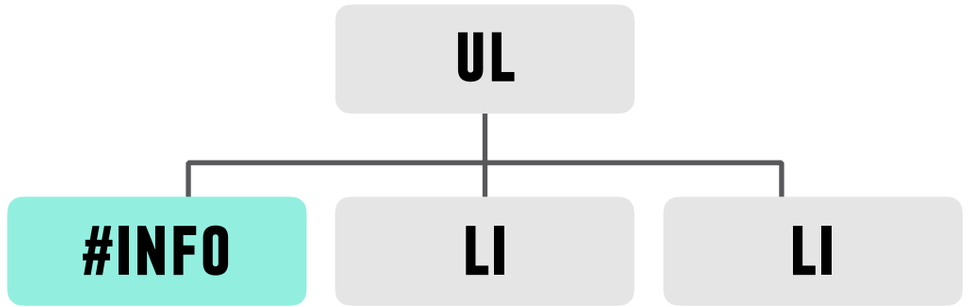
JQUERY METHODS — WORKING WITH THOSE ELEMENTS

After we've selected elements, we can use jQuery methods to:

See your handout or the jQuery docs for list!

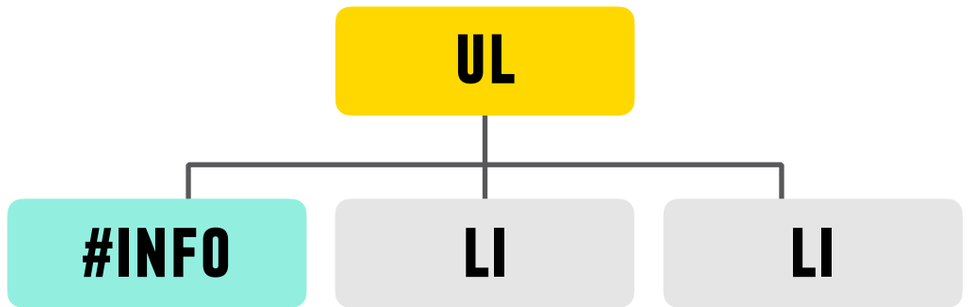
TRAVERSING THE DOM?

```
$('#info').parent();
```



TRAVERSING THE DOM?

```
$('#info').parent();
```



JQUERY METHODS — TRAVERSING THE DOM

TRAVERSE THE DOM

- Think of these as filters, or part of the selection process.
- They must come *directly after another selection*

METHODS	EXAMPLES
`.find()` *finds all descendants*	`$('h1').find('a');`
`.parent()`	`$('#box1').parent();`
`.siblings()`	`$('p').siblings('.important');`
`.children()`	`$('ul').children('li');`

What goes in the parentheses?
A **css-style selector**

JQUERY METHODS — WORKING WITH THOSE ELEMENTS

After we've selected elements, we can use jQuery methods to:

See your handout or the jQuery docs for list!

GETTING/SETTING CONTENT — PART 1

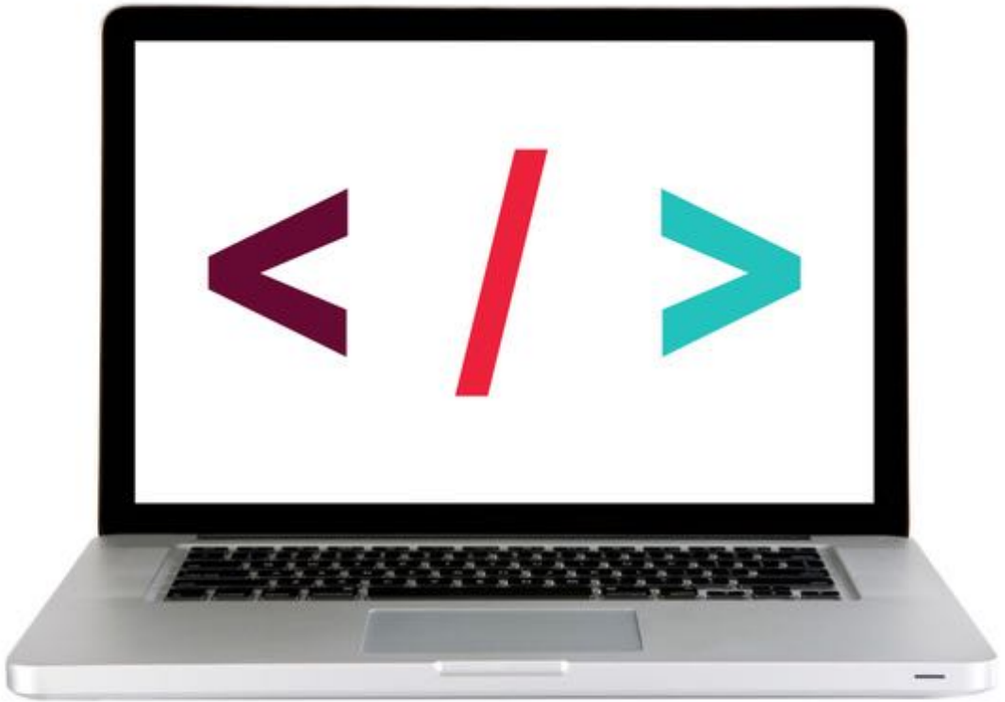
GET/SET CONTENT

Get/change content of elements and attributes

METHODS	EXAMPLES
.html()	$('h1').html('Content to insert goes here');
.attr()	$('img').attr('src', 'images/bike.png');

What goes in the parentheses?
The **html** you want to change.

LET'S TAKE A CLOSER LOOK



GETTING/SETTING CONTENT — PART 2

GET/SET CONTENT

Get/change content of elements and attributes

METHODS	EXAMPLES
`.addClass()`	`$('p').addClass('success');`
`.removeClass()`	`$('p').removeClass('my-class-here');`
`.toggleClass()`	`$('p').toggleClass('special');`

What goes in the parentheses?
The **classes** you want to change.

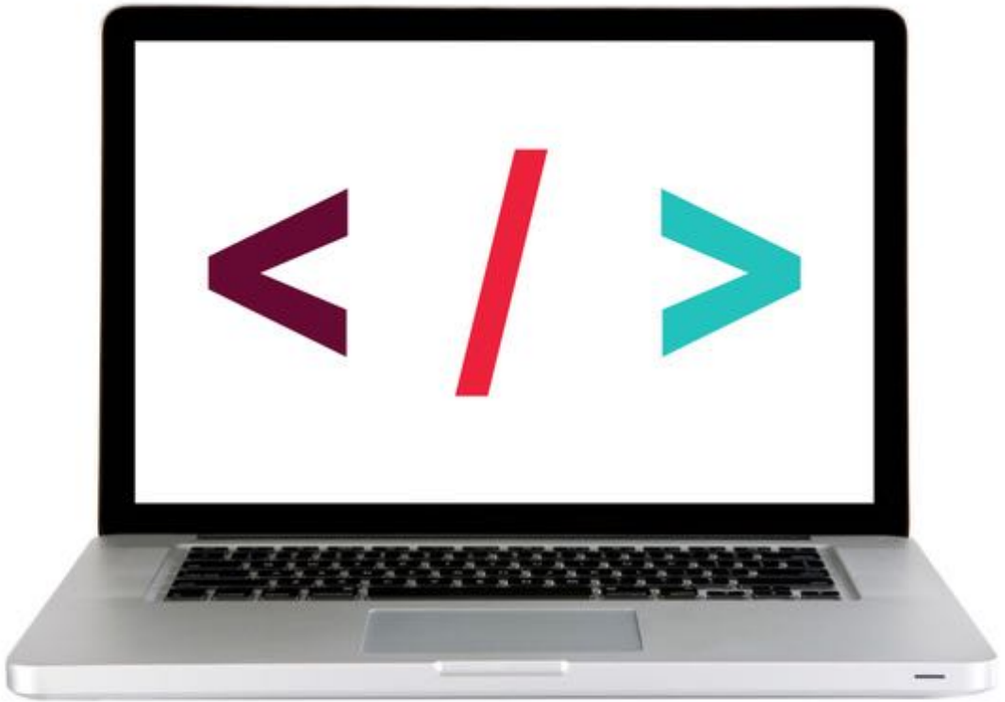
JQUERY METHODS — GETTING/SETTING CONTENT

GET/SET CONTENT

```
$('li').addClass('selected');
```

NO PERIOD!!!

LET'S TAKE A CLOSER LOOK



ACTIVITY

KEY OBJECTIVE

- Utilize jQuery to access and manipulate DOM elements.

TYPE OF EXERCISE

- Individual/Partner

TIMING

5 min `09-jquery-exercise`

1. Follow the instructions under part 1 in main.js
2. Use cheat sheet/slides as a guide for syntax

JQUERY METHODS — WORKING WITH THOSE ELEMENTS

After we've selected elements, we can use jQuery methods to:

See your handout or the jQuery docs for list!

JQUERY METHODS — EFFECTS/ANIMATION

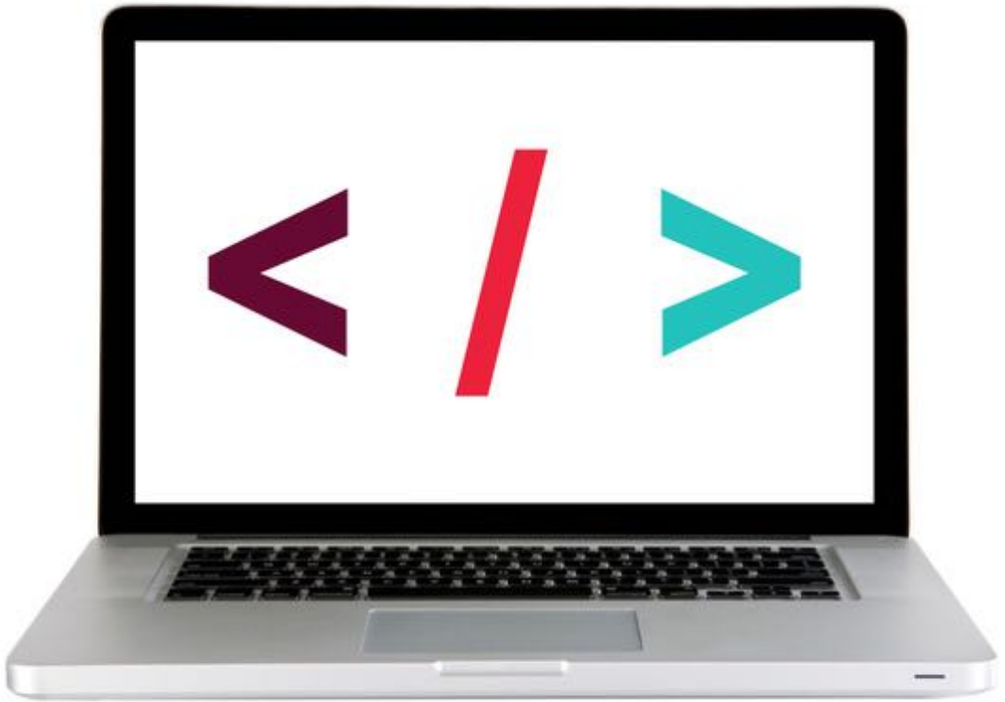
ADD EFFECTS/ ANIMATION

Add effects and animation to parts of the page

METHODS	EXAMPLES
.show()	$('h1').show();
.hide()	$('ul').hide();
.fadeIn()	$('h1').fadeIn(300);
.fadeOut()	$('.special').fadeOut('fast');
.slideUp()	$('div').slideUp();
.slideDown()	$('#box1').slideDown('slow');
.slideToggle()	$('p').slideToggle(300);

What goes in the parenthesis?
An animation speed

LET'S TAKE A CLOSER LOOK



JQUERY METHODS — WORKING WITH THOSE ELEMENTS

After we've selected elements, we can use jQuery methods to:

See your handout or the jQuery docs for list!

JQUERY METHODS — EVENTS!

CREATE EVENT LISTENERS

We can use the `on()` method to handle all events in jQuery.

JQUERY METHODS — EVENTS!

CREATE EVENT LISTENERS

selector

```
$('li').on('click', function() {
  // your code here
});
```

JQUERY METHODS — EVENTS!

CREATE EVENT LISTENERS

method for all events

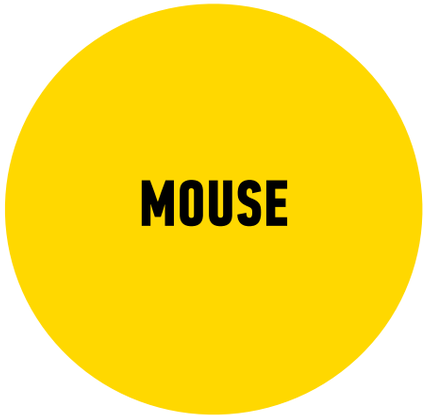
```
$('li').on('click', function() {
  // your code here
});
```

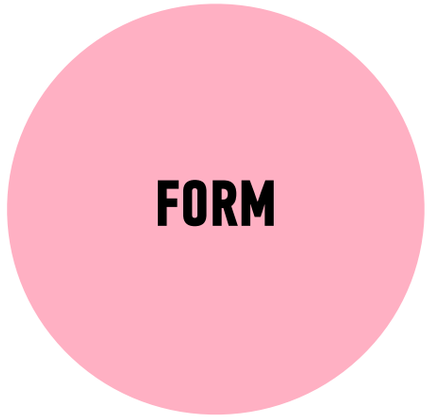
JQUERY METHODS — EVENTS!

CREATE EVENT LISTENERS

type of event

```
$('li').on('click', function() {
  // your code here
});
```





click

dblclick

mouseenter

mouseleave

keypress

keydown

keyup

submit

change

focus

blur

resize

scroll

```
$('li').on('eventGoesHere', function() {
  // your code here
});
```

JQUERY METHODS — EVENTS!

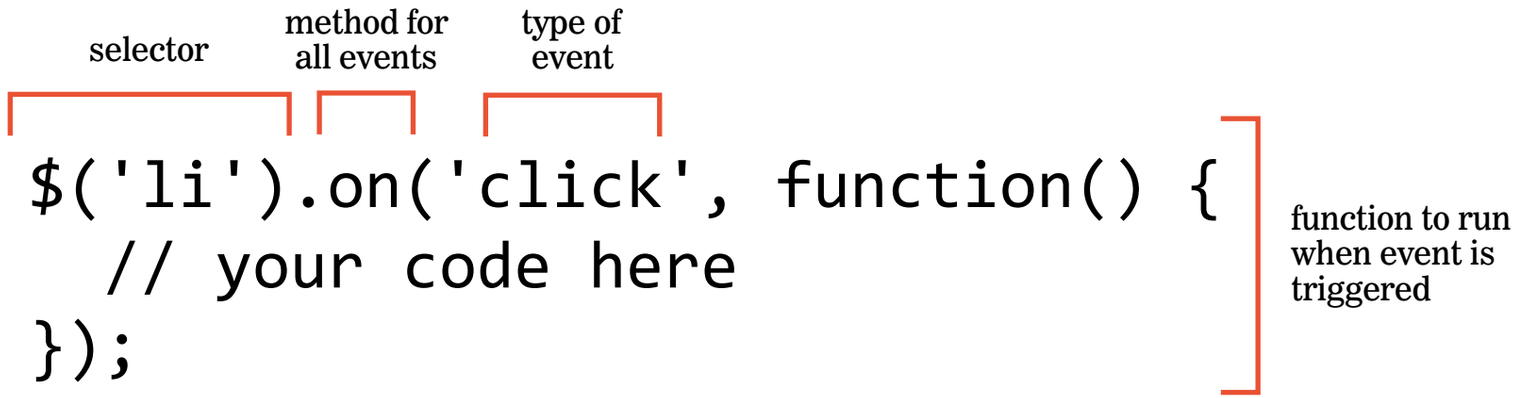
CREATE EVENT LISTENERS

```
$('li').on('click', function() {
  // your code here
});
```

function to run when event is triggered

JQUERY METHODS — EVENTS!

CREATE EVENT LISTENERS



ACTIVITY

KEY OBJECTIVE

- Utilize jQuery to access and manipulate DOM elements.

TYPE OF EXERCISE

- Individual/Partner

TIMING

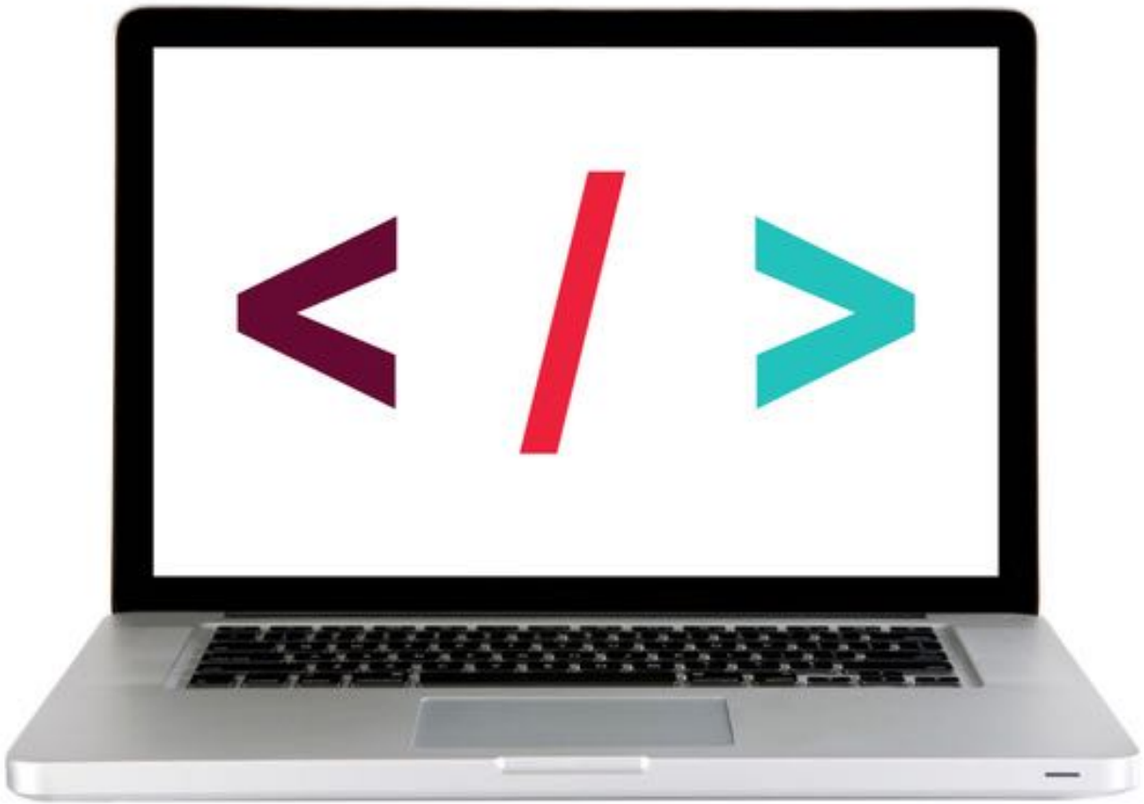
5 min Continue with `09-jquery-exercise`

1. Follow the instructions under Part 2 in main.js
2. Use cheat sheet/slides as a guide for syntax

REFACTORING

- **Refactoring** is the process of rewriting code to make it more efficient, or to incorporate new coding practices
- Rewriting code to replace vanilla JavaScript with jQuery methods is an example of refactoring

INTRO TO JQUERY



LET'S TAKE A CLOSER LOOK

EXERCISE

OBJECTIVE

- Create DOM event handlers using jQuery

LOCATION

- `starter-code > 11-refactoring-exercise`

TIMING

until 9:20

1. Refactor the favorites list you built earlier in JavaScript to use jQuery selectors and methods instead.
2. BONUS: Use jQuery to add a "complete task" link at the end of each to-do item when it is added to the list.

Exit Tickets!

(Class #7)

LEARNING OBJECTIVES - REVIEW

- Explain and use JavaScript methods for DOM manipulation.
- Create DOM event handlers to respond to user actions
- Manipulate the DOM by using jQuery selectors and functions.
- Create DOM event handlers using jQuery

NEXT CLASS PREVIEW

Advanced jQuery

- Use event delegation to manage dynamic content.
- Use implicit iteration to update elements of a jQuery selection

Q&A