

JAVASCRIPT DEVELOPMENT

Sasha Vodnik, Instructor

HELLO!

1. Submit your homework and create a pull request
2. Pull changes from the `svodnik/JS-SF-8-resources` repo to your computer
3. Open the `starter-code` folder in your code editor

JAVASCRIPT DEVELOPMENT

ADVANCED JQUERY & TEMPLATING

LEARNING OBJECTIVES

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

- Manipulate the DOM by using jQuery selectors and functions.
- Register and trigger event handlers for jQuery events.
- Use chaining to place methods on selectors.
- Use event delegation to manage dynamic content.
- Use implicit iteration to update elements of a jQuery selection
- Build content programmatically using ES6 template literals

AGENDA

- jQuery methods
- jQuery events
- jQuery best practices
- Template literals

ADVANCED JQUERY & TEMPLATING

WEEKLY OVERVIEW

WEEK 6

Advanced jQuery & templating / Ajax & APIs

WEEK 7

Asynchronous JavaScript & Callbacks / Advanced APIs

WEEK 8

Project 2 Lab / Context and this

EXIT TICKET FEEDBACK

1. How is Javascript faster if it requires more code written than JQuery?
2. -What use case would lead you to declaring a variable to store a selector in jQuery?
-Javascript is still confusing for me. Takes a while for me to write. Creating pointers for everything and variables that stand for something else, just doesn't come easily. But maybe it's because I'm so used to jquery by now.
3. Can we have questions and examples that aren't the exact same thing we just went over?

EXIT TICKET FEEDBACK

4. When would you use `css: hover` vs. javascript `mouseover`?
5. Is it faster to add jQ to your `main.js` file and minify that (reducing the number of HTTP requests) or to use a CDN for jQ?

ADVANCED JQUERY & TEMPLATING

HOMework REvIEW

HOMEWORK — GROUP DISCUSSION



EXERCISE

TYPE OF EXERCISE

- ▶ Groups of 3

TIMING

4 min

1. Share your solutions for the DOM 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.

ADVANCED JQUERY & TEMPLATING

JQUERY REVIEW

JQUERY

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

Selector

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

JQUERY OBJECTS — FINDING ELEMENTS: SOME EXAMPLES

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

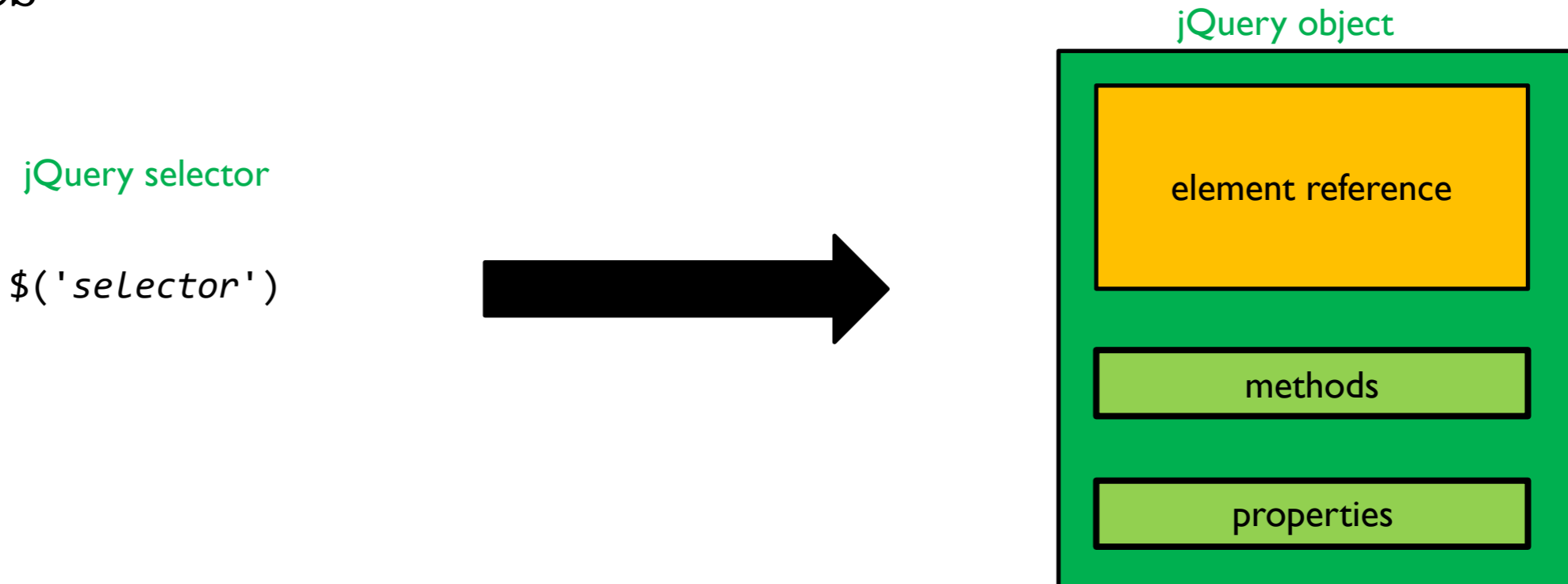
JQUERY OBJECTS

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



JQUERY OBJECTS

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



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 ');
```

JQUERY

PART 2 — ADD A METHOD

USING JQUERY TO MANIPULATE THE DOM

1

Select an element/elements

2

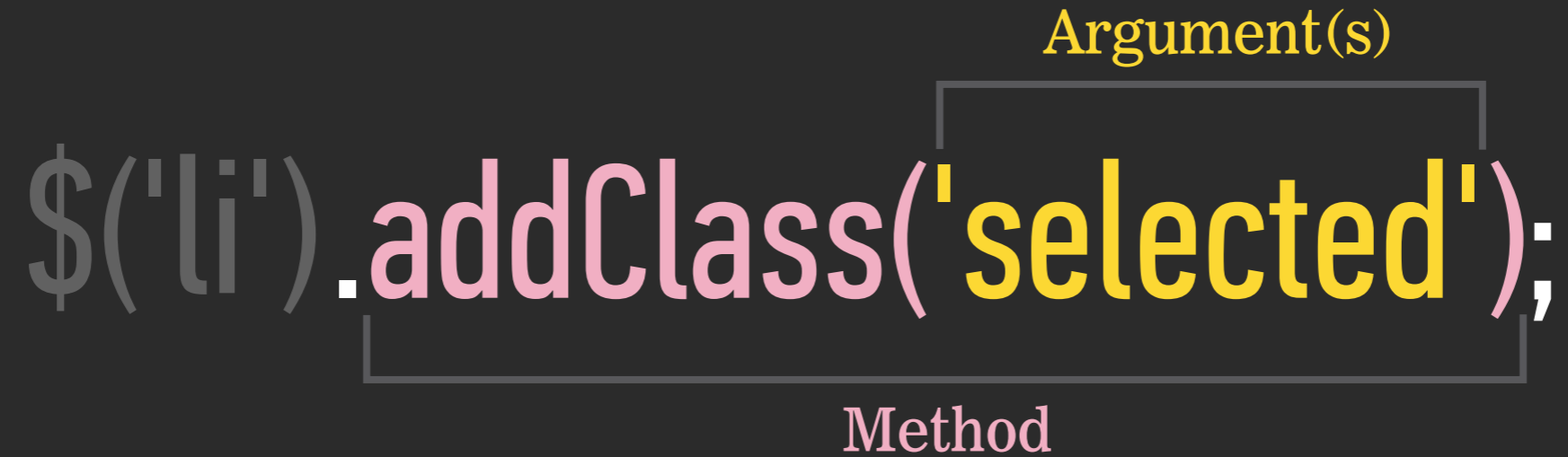
Work with those elements

JQUERY — WORKING WITH THOSE ELEMENTS

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

Argument(s)

Method

The image shows the jQuery code snippet `$('.li').addClass('selected');` with two annotations. A bracket above the string `'selected'` is labeled "Argument(s)" in yellow text. A bracket below the `addClass` method name is labeled "Method" in pink text. The `addClass` text is also highlighted in pink, and the string `'selected'` is highlighted in yellow.

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:

**FIND
ELEMENTS**

**GET/SET
CONTENT**

**ADD
EFFECTS/
ANIMATION**

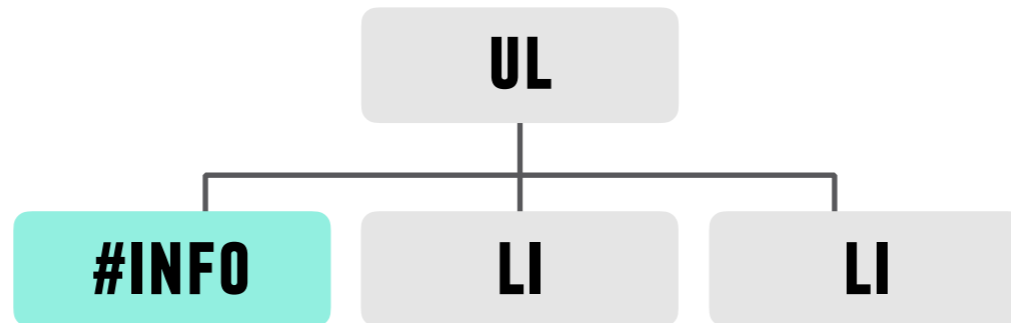
**CREATE
EVENT
LISTENERS**



See your handout or the [jQuery docs](#) for list!

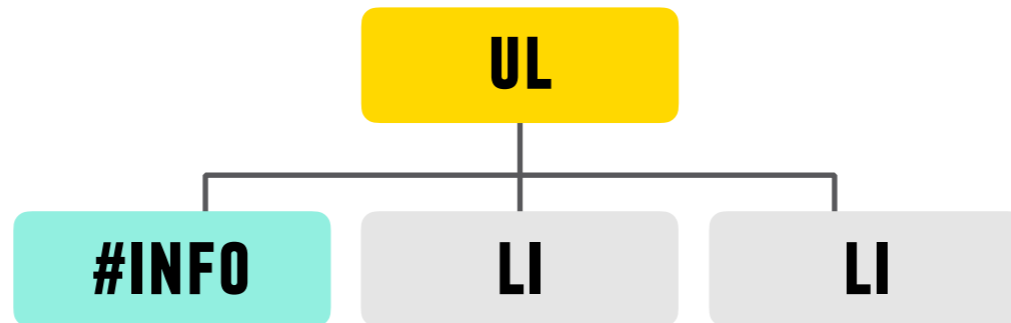
TRaversing the DOM?

```
$('#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
<code>.find()</code> <i>finds all descendants</i>	<code>\$('#h1').find('a');</code>
<code>.parent()</code>	<code>\$('#box1').parent();</code>
<code>.siblings()</code>	<code>\$('#p').siblings('.important');</code>
<code>.children()</code>	<code>\$('#ul').children('li');</code>

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:

**FIND
ELEMENTS**

**GET/SET
CONTENT**

**ADD
EFFECTS/
ANIMATION**

**CREATE
EVENT
LISTENERS**



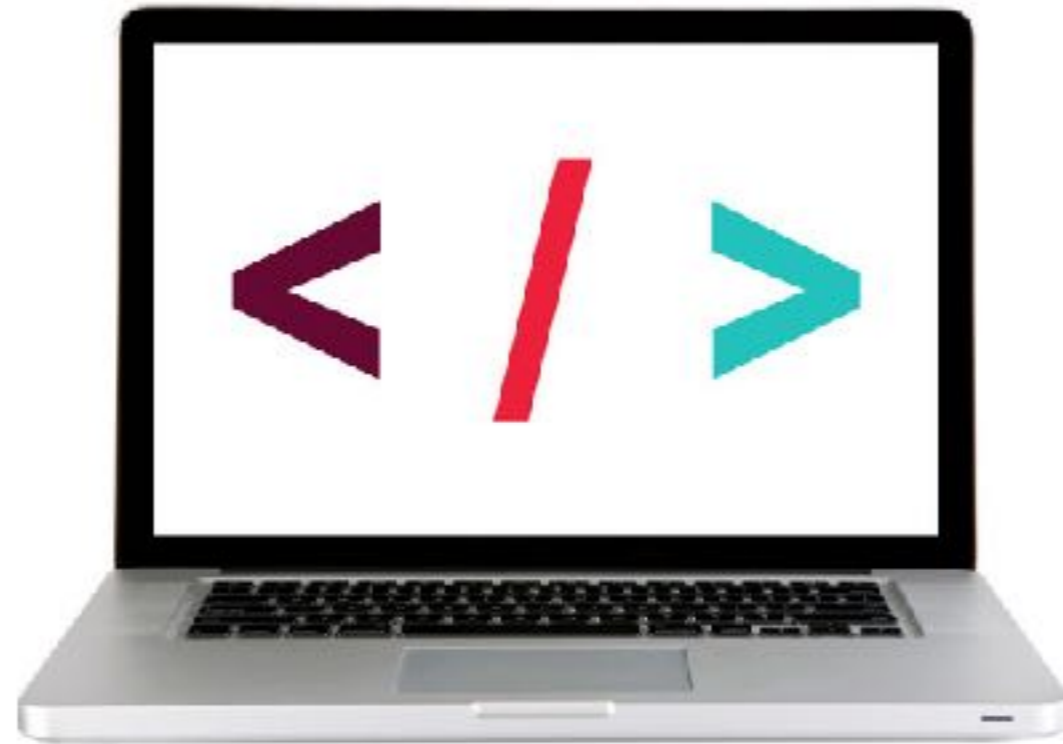
See your handout or the [jQuery docs](#) for list!

Get/change content of elements and attributes

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

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

LET'S TAKE A CLOSER LOOK



Get/change content of elements and attributes

METHODS	EXAMPLES
<code>.addClass()</code>	<code>\$('.p').addClass('success');</code>
<code>.removeClass()</code>	<code>\$('.p').removeClass('my-class-here');</code>
<code>.toggleClass()</code>	<code>\$('.p').toggleClass('special');</code>

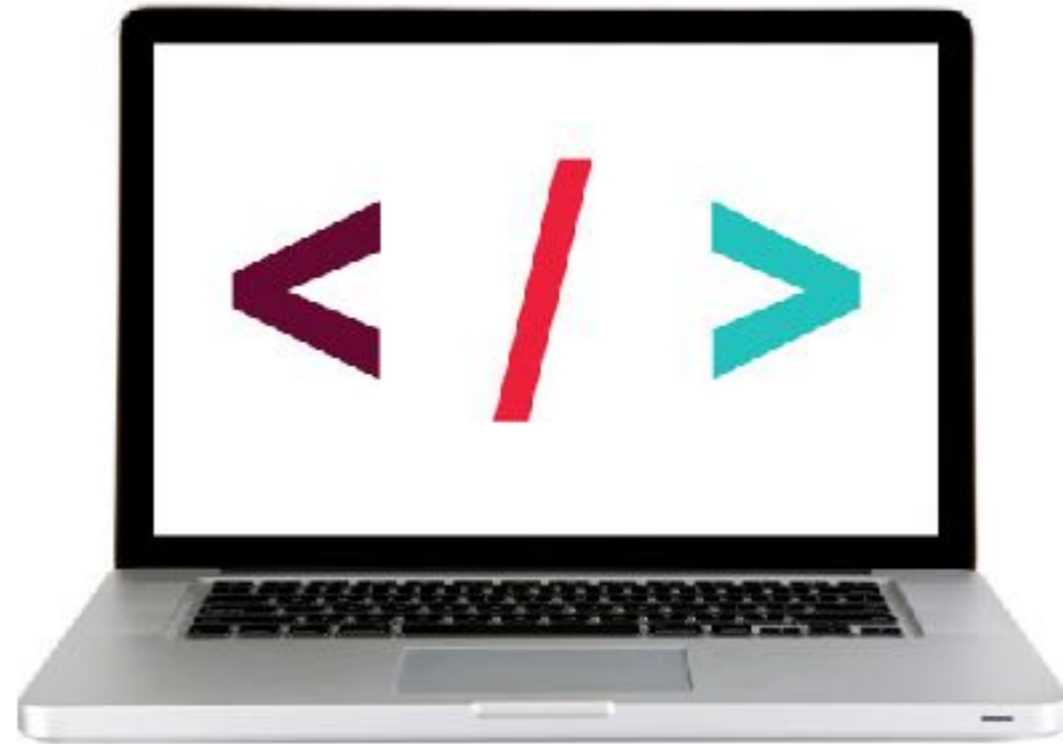
What goes in the parentheses?
The **classes** you want to change.

```
$('.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

Exercise is in 1-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:

**FIND
ELEMENTS**

**GET/SET
CONTENT**

**ADD
EFFECTS/
ANIMATION**

**CREATE
EVENT
LISTENERS**



See your handout or the [jQuery docs](#) for list!

JQUERY METHODS — EFFECTS/ANIMATION

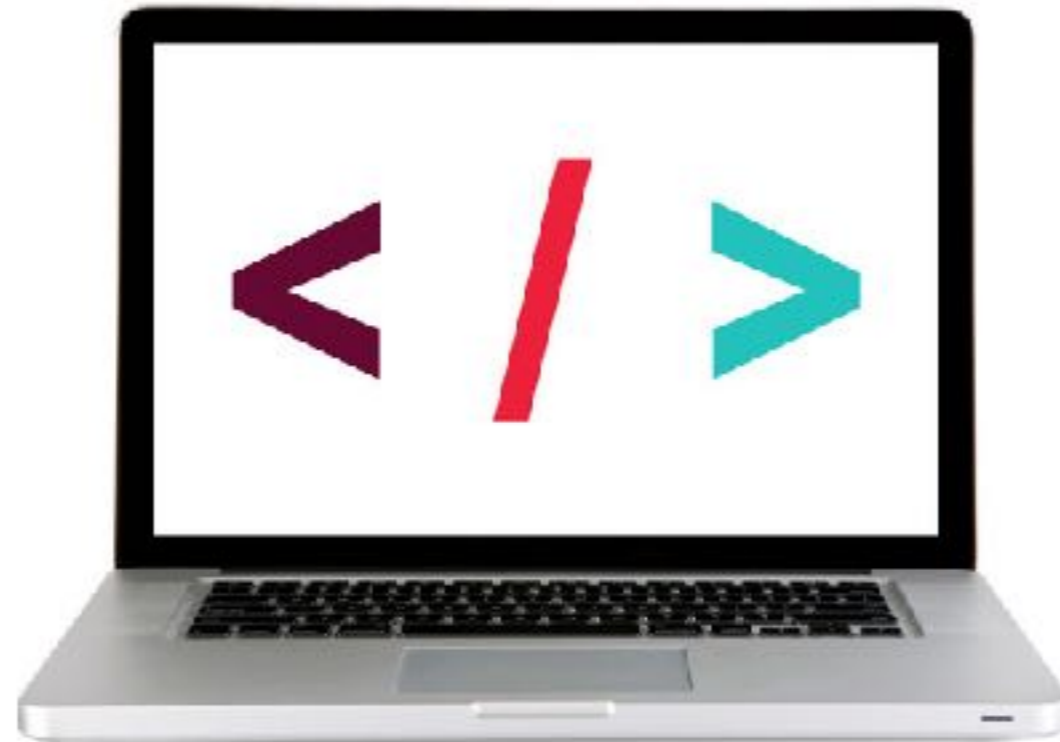
**ADD
EFFECTS/
ANIMATION**

Add effects and animation to parts of the page

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

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:

**FIND
ELEMENTS**

**GET/SET
CONTENT**

**ADD
EFFECTS/
ANIMATION**

**CREATE
EVENT
LISTENERS**



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  
});
```

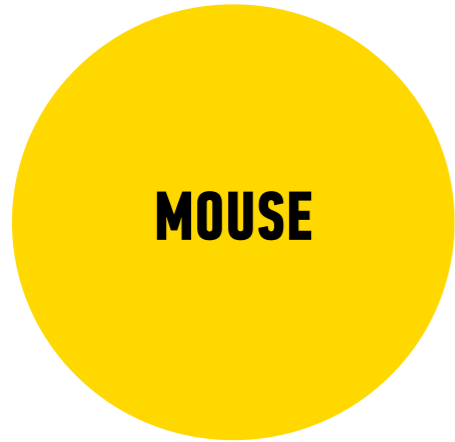

method for all events

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

JQUERY METHODS — EVENTS!

**CREATE
EVENT
LISTENERS**

```
                                type of event  
                                ┌──────────┐  
$( 'li' ).on( 'click', function() {  
    // your code here  
});
```



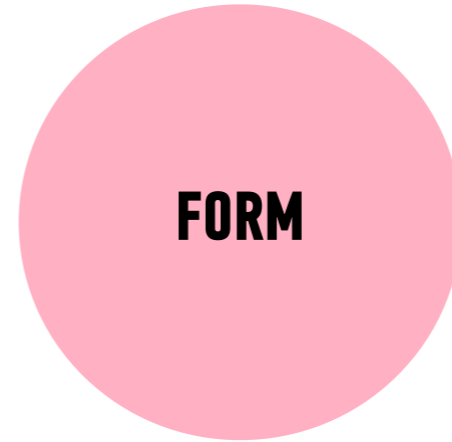
MOUSE

click
dblclick
mouseenter
mouseleave



KEYBOARD

keypress
keydown
keyup



FORM

submit
change
focus
blur



DOCUMENT

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

```
selector      method for      type of  
              all events   event  
┌──────────┐ ┌──┐ ┌──────────┐  
$( 'li' ).on( 'click', function() {  
    // do something  
});
```

function to run
when event is
triggered

ACTIVITY



EXERCISE

KEY OBJECTIVE

- ▶ Utilize jQuery to access and manipulate DOM elements.

TYPE OF EXERCISE

- ▶ Individual/Partner

TIMING

5 min

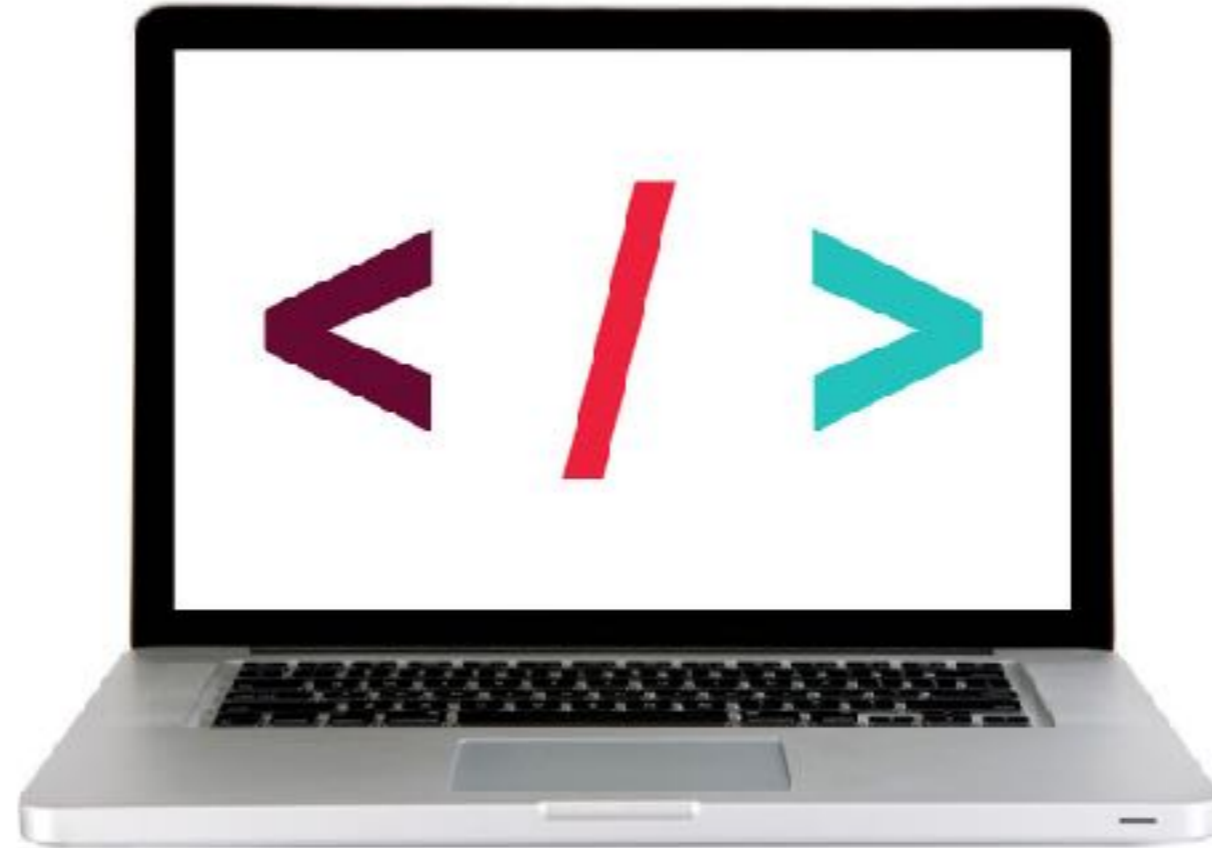
Continue with 01-jquery-exercise

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

INTRO TO JQUERY



EXERCISE



OBJECTIVE

- ▶ Manipulate the DOM by using jQuery selectors and functions.

LOCATION

- ▶ `starter-code > 3-jquery-todo-list`

TIMING

until 9:20

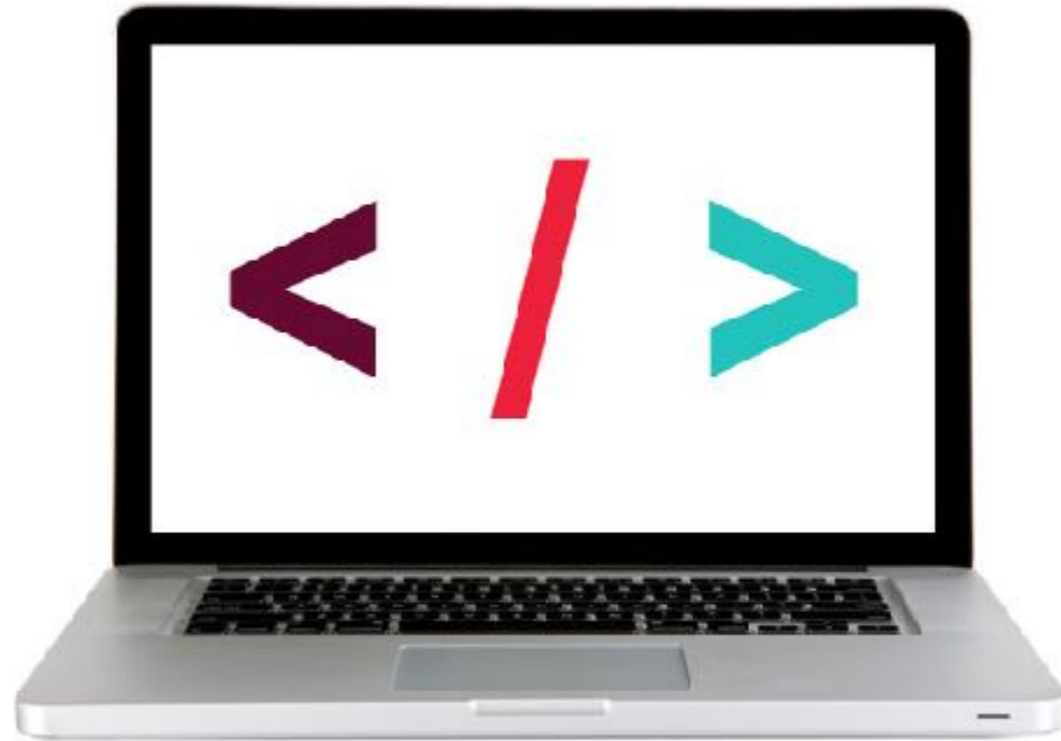
1. The HTML document contains an empty unordered list. It also contains a text input box and a Create button. Write jQuery to enable users to add elements to the to do list.
2. BONUS: Use jQuery to add a "complete task" link at the end of each to-do item when it is added to the list.

BEST PRACTICES

JQUERY

METHOD CHAINING

ACTIVITY — METHOD CHAINING



METHOD CHAINING!!!

JQUERY METHODS — METHOD CHAINING

```
$( )
```

```
.slideUp( )
```

```
'li'
```

```
'slow'
```

JQUERY METHODS — METHOD CHAINING

```
$( 'li' )
```

```
.slideUp( 'slow' )
```

JQUERY METHODS — METHOD CHAINING

```
$( 'li' ).slideUp( 'slow' );
```

JQUERY METHODS — METHOD CHAINING

```
$( )
```

```
.addClass( )
```

```
'li'
```

```
'.complete'
```

```
'complete'
```

JQUERY METHODS — METHOD CHAINING

```
$( 'li' )
```

```
.addClass( 'complete' )
```

JQUERY METHODS — METHOD CHAINING

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

JQUERY METHODS — METHOD CHAINING

```
$( )
```

```
.html( )
```

```
'li'
```

```
300
```

```
'<li>Feed cat</li>'
```

JQUERY METHODS — METHOD CHAINING

```
$( 'li' )
```

```
.html( '<li>Feed cat</li>' )
```

JQUERY METHODS — METHOD CHAINING

```
$( 'li' ).html( '<li>Feed cat</li>' );
```

JQUERY METHODS — METHOD CHAINING

`$()`

`.show()`

`.siblings()`

`'h3'`

`'p'`

JQUERY METHODS — METHOD CHAINING

```
$( 'h3' )
```

```
.show()
```

```
.siblings( 'p' )
```

JQUERY METHODS — METHOD CHAINING

```
$('h3').siblings('p').show();
```

JQUERY METHODS — METHOD CHAINING

```
$( )
```

```
.slideUp( )
```

```
.find( )
```

```
' .item '
```

```
300
```

```
' h2 '
```

JQUERY METHODS — METHOD CHAINING

```
$('.item')
```

```
.slideUp(300)
```

```
.find('h2')
```

JQUERY METHODS — METHOD CHAINING

```
$('.item').find('h2').slideUp(300);
```

JQUERY METHODS — METHOD CHAINING

`$()`

`.fadeOut()`

`.children()`

`'#main'`

`'slow'`

`'p'`

JQUERY METHODS — METHOD CHAINING

```
$('#main')
```

```
.fadeOut('slow')
```

```
.children('p')
```

JQUERY METHODS — METHOD CHAINING

```
$('#main').children('p').fadeOut('slow')
```

EXERCISE - CHAINING



EXERCISE

OBJECTIVE

- ▶ Use chaining to place methods on selectors.

LOCATION

- ▶ `starter-code > 4-best-practices-exercise`

TIMING

3 min

1. In your browser, open `index.html` and test the functionality.
2. Open `main.js` in your editor and complete items 1 and 2.
3. In your browser, reload `index.html` and verify that the functionality is unchanged.

JQUERY

IMPLICIT ITERATION

IMPLICIT ITERATION

explicit iteration

selects a jQuery collection

.each() method works like a forEach loop

```
$('li').each(function() {  
  $(this).removeClass('current');  
});
```



not necessary for element collections

implicit iteration

selects a jQuery collection

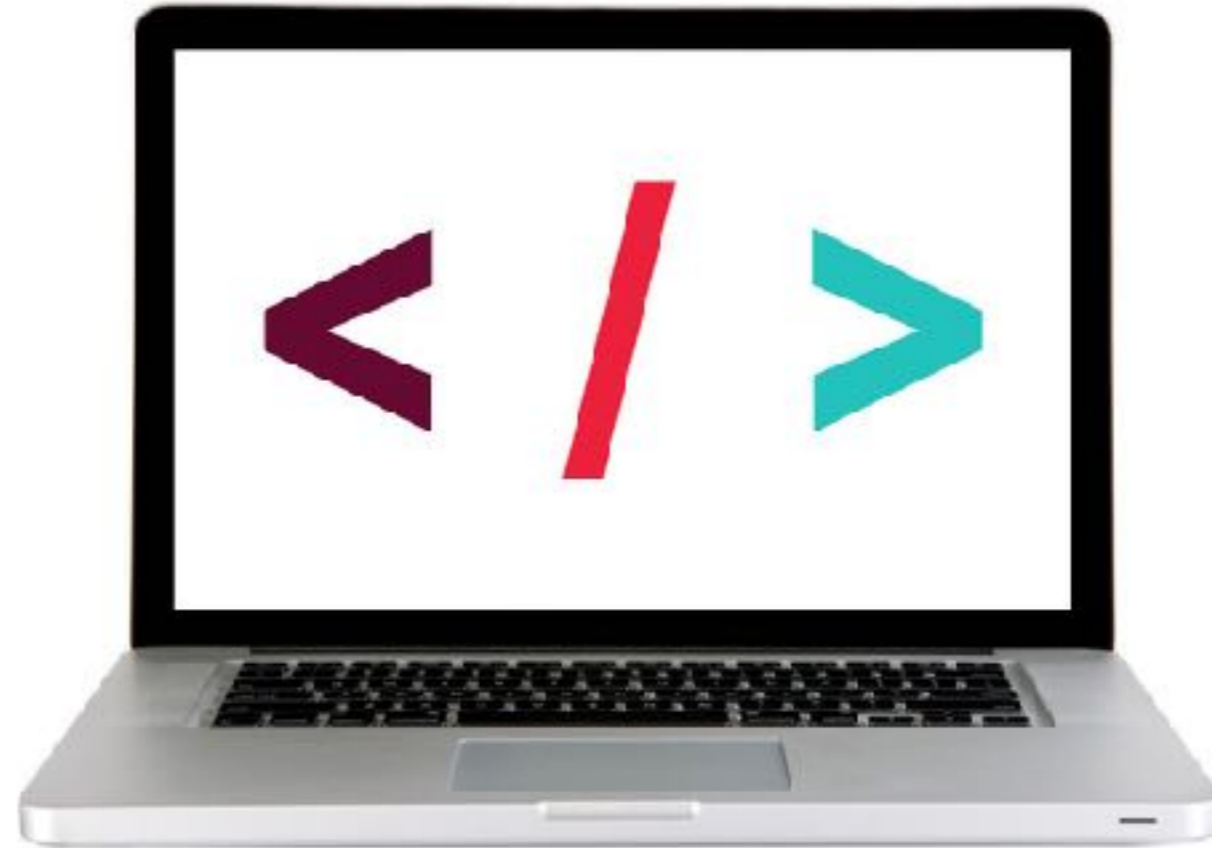
applying any method to a jQuery collection iterates through each element

```
$('li').removeClass('current');
```



less code = best practice!

INTRO TO JQUERY



LET'S TAKE A CLOSER LOOK

EXERCISE – IMPLICIT ITERATION



OBJECTIVE

- ▶ Use implicit iteration to update elements of a jQuery selection.

LOCATION

- ▶ `starter-code > 4-best-practices-exercise`

TIMING

5 min

1. Return to `main.js` in your editor and complete item 3.
2. In your browser, reload `index.html` and verify that the functionality is unchanged.

JQUERY

EVENT DELEGATION

WITHOUT EVENT DELEGATION

add an event listener to each li in the DOM

1. load page

2. set event listener on list items

3. add a new list item

```
$( 'li' ).on( 'click', function() {  
    addClass( 'selected' )  
});
```

- item1
- item2
- item3

- item1 click event
- item2 click event
- item3 click event

- item1 click event
- item2 click event
- item3 click event
- item4

click event is not automatically applied to the new li element



WITH EVENT DELEGATION

1. load page

- item1
- item2
- item3

2. set event listener on parent of list items

selector changed from 'li' to 'ul'

new argument 'li' added to on() method

```
$('#ul').on('click', 'li', function(){  
  addClass('selected')  
});
```

- item1 click event
- item2 click event
- item3 click event

add an event listener to the ul element that applies to all of its li descendants

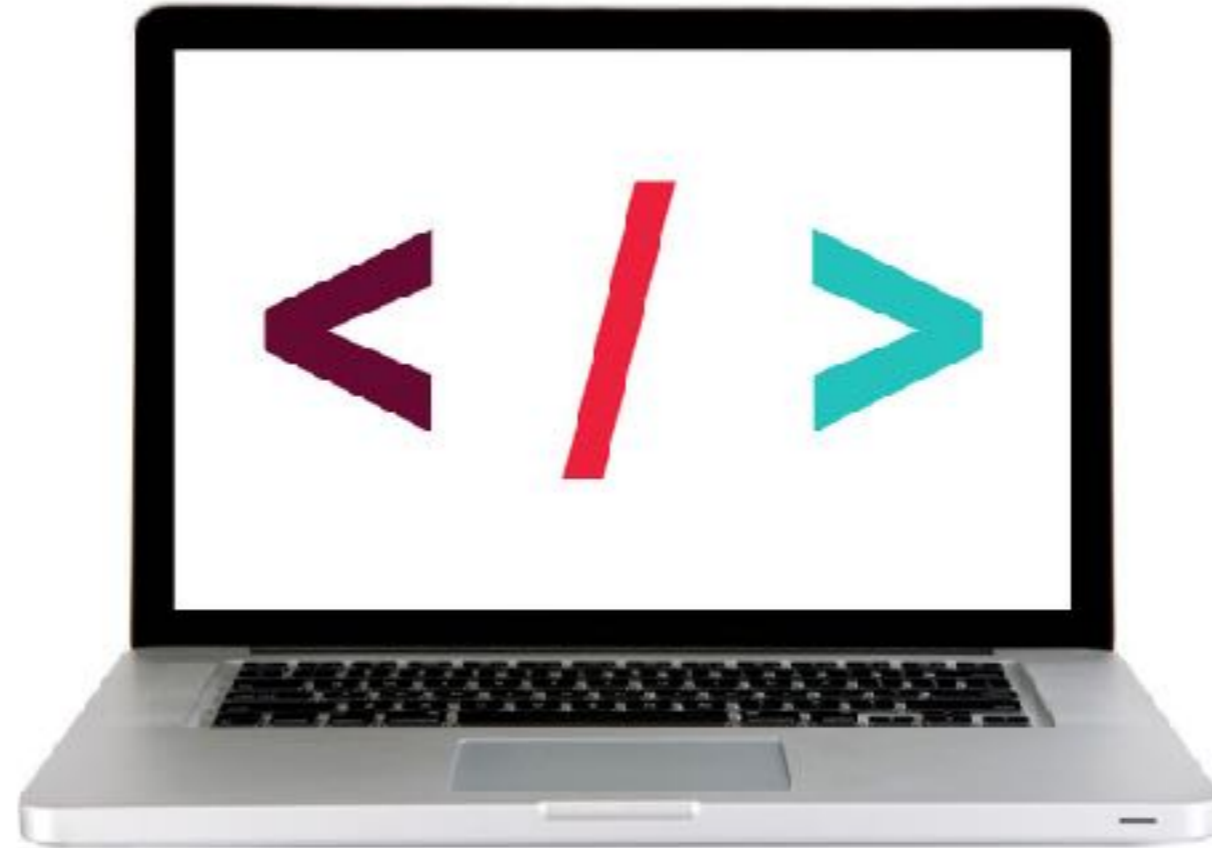
3. add a new list item

- item1 click event
- item2 click event
- item3 click event
- item4 click event

click event IS automatically applied to the new li element!



INTRO TO JQUERY



LET'S TAKE A CLOSER LOOK

EXERCISE – EVENT DELEGATION



EXERCISE

OBJECTIVE

- ▶ Use event delegation to manage dynamic content.

LOCATION

- ▶ `starter-code > 4-best-practices-exercise`

TIMING

10 min

1. Return to `main.js` in your editor and complete items 4a and 4b.
2. In your browser, reload `index.html` and verify that when you add a new item to the list, its “cross off” link works.
3. BONUS: When the user mouses over each item, the item should turn grey. Don't use CSS hovering for this.
4. BONUS: Add another link, after each item, that allows you to delete the item.

JQUERY

ATTACHING MULTIPLE EVENTS WITH A SINGLE ON() STATEMENT

ATTACHING MULTIPLE EVENTS WITH A SINGLE .ON() STATEMENT

- ▶ We could write a separate .on() statement for each event on an element:

```
var $listElement = $('#contents-list');

$listElement.on('mouseenter', 'li', function(event) {
    $(this).siblings().removeClass('active');
    $(this).addClass('active');
});

$listElement.on('mouseleave', 'li', function(event) {
    $(this).removeClass('active');
});
```

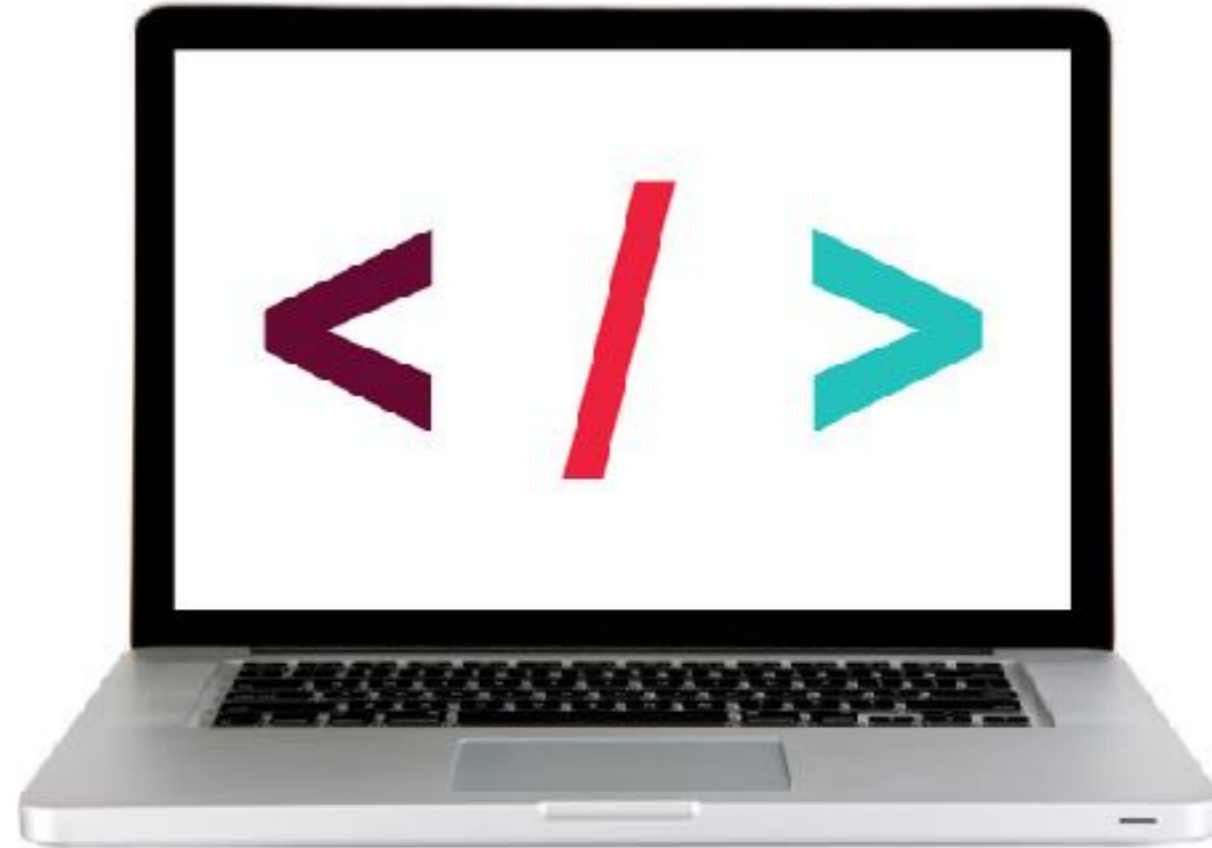
ATTACHING MULTIPLE EVENTS WITH A SINGLE .ON() STATEMENT

- Grouping all the events for an element in a single .on() statement makes our code more organized and is faster

```
var $listElement = $('#contents-list');

$listElement.on('mouseenter mouseleave', 'li', function(event) {
    if (event.type === 'mouseenter') {
        $(this).siblings().removeClass('active');
        $(this).addClass('active');
    } else if (event.type === 'mouseleave') {
        $(this).removeClass('active');
    }
});
```

INTRO TO JQUERY



LET'S TAKE A CLOSER LOOK

EXERCISE - ATTACHING MULTIPLE EVENTS



EXERCISE

LOCATION

▶ `starter-code > 6-multiple-events-exercise`

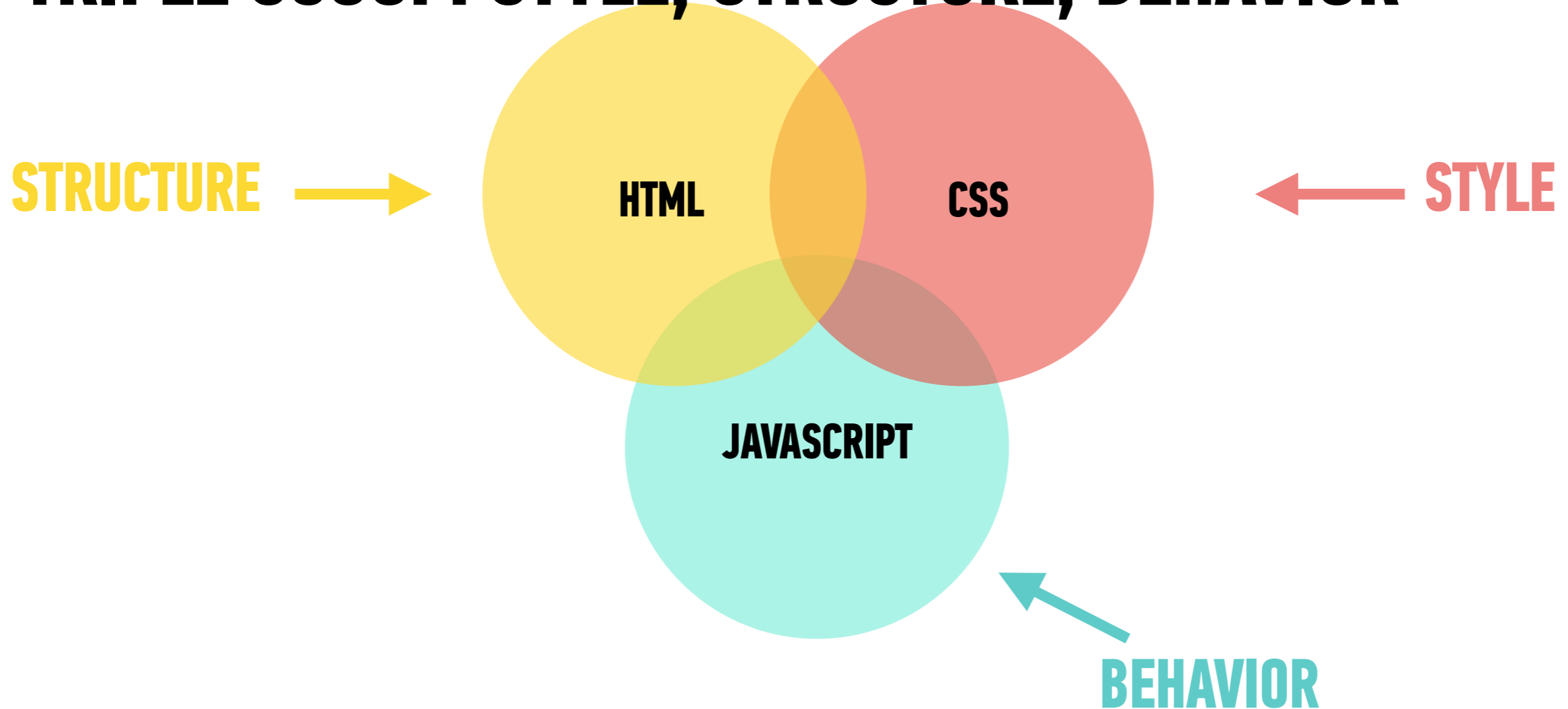
TIMING

5 min

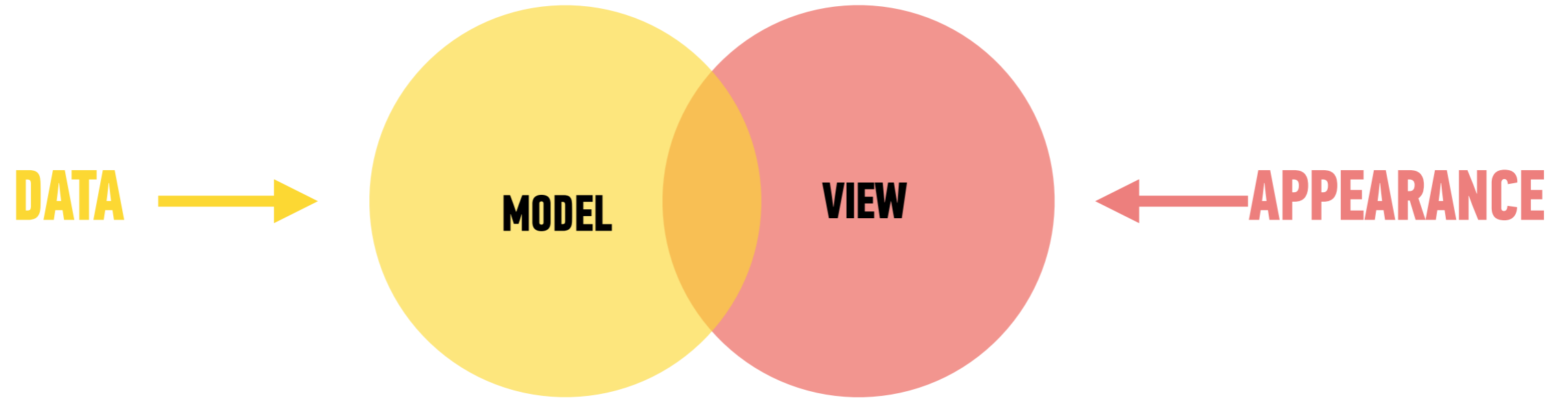
1. In your browser, open `index.html`. Move the mouse over each list item and verify that the sibling items turn gray.
2. In your editor, open `main.js` and refactor the two event listeners near the bottom of the file into a single event listener for multiple events.
3. In your browser, reload `index.html` and verify that the functionality is unchanged.

TEMPLATING

TRIPLE SCOOP: STYLE, STRUCTURE, BEHAVIOR



MODEL VS VIEW



TEMPLATE LITERALS

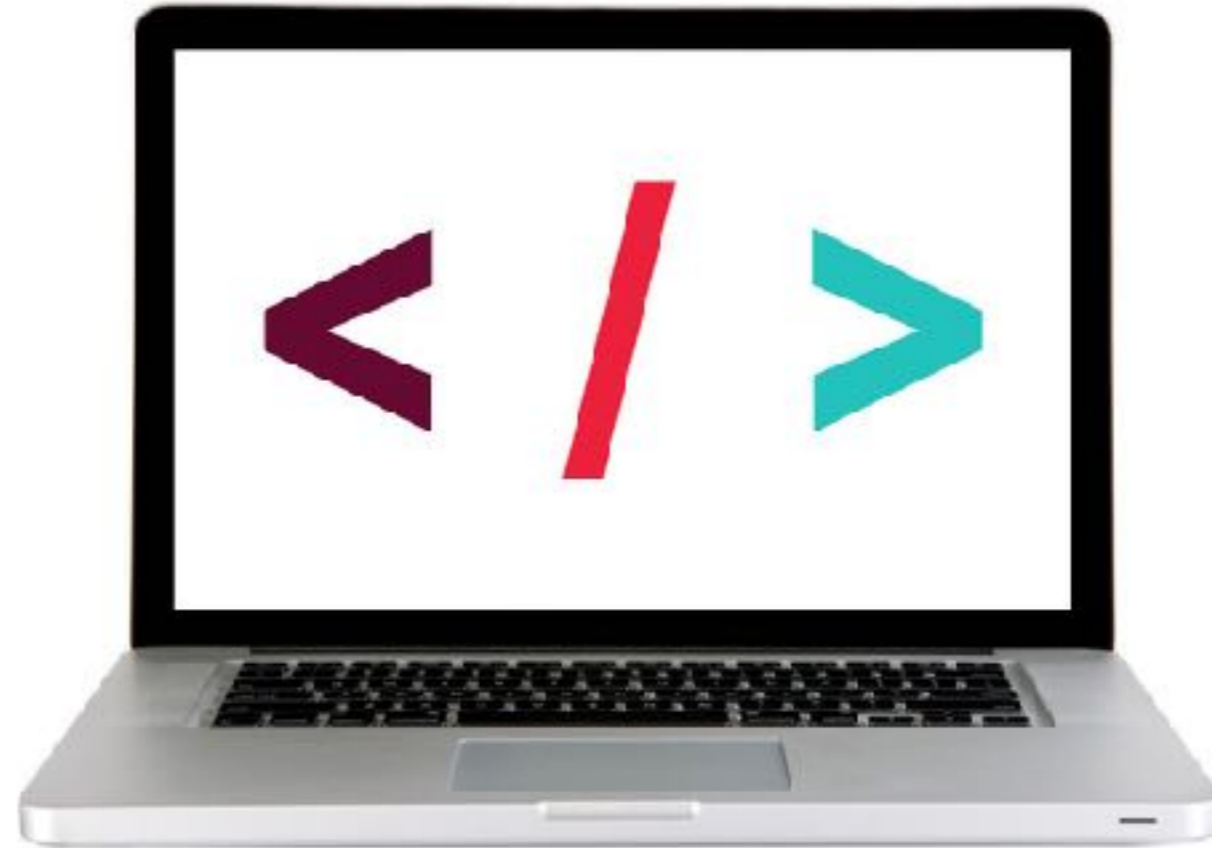
template literal starts and ends with a backtick

```
conditionsPara.innerHTML = `${state.degCInt} C / ${state.degFInt} F`;
```

variable reference starts with a dollar sign

variable reference surrounded by curly braces

INTRO TO JQUERY



LET'S TAKE A CLOSER LOOK

EXERCISE - TEMPLATING



EXERCISE

LOCATION

▶ starter-code > 9-templating-lab

TIMING

10 min

1. Create a template literal and use it to display the data in the favorite object.
2. Use the HTML structure shown in main.js.
3. **BONUS:** create a template literal that displays the contents of the 'favorites' object at the bottom of main.js.

LEARNING OBJECTIVES – REVIEW

- Manipulate the DOM by using jQuery selectors and functions.
- Register and trigger event handlers for jQuery events.
- Use chaining to place methods on selectors.
- Use event delegation to manage dynamic content.
- Use implicit iteration to update elements of a jQuery selection
- Build content programmatically using ES6 template literals

NEXT CLASS PREVIEW

Ajax & APIs

- Identify all the HTTP verbs & their uses.
- Describe APIs and how to make calls and consume API data.
- Access public APIs and get information back.
- Implement an Ajax request with Fetch.
- Create an Ajax request using jQuery.
- Reiterate the benefits of separation of concerns – API vs. Client.

Exit Tickets!

(Class #9)

Q&A