GENERAL ASSEMBLY

JAVASCRIPT DEVELOPMENT

Sasha Vodnik, Instructor

HELLO!

1. Pull changes from the `svodnik/JS-SF-16-resources` repo to your computer
2. Open the `08-advanced-jquery` folder in your editor

EVENTS & JQUERY

LEARNING OBJECTIVES

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

- Create DOM event handlers using vanilla JavaScript.
- Select DOM elements and properties using jQuery.
- Manipulate the DOM by using jQuery selectors and functions.
- Use event delegation to manage dynamic content.
- Use implicit iteration to update elements of a jQuery selection

AGENDA

- Events
- jQuery
- Event delegation
- Implicit iteration

WEEKLY OVERVIEW

WEEK 5	Events & jQuery / Ajax & APIs
WEEK 6	Asynchronous JS & callbacks / Advanced APIs
WEEK 7	Project 2 lab / Prototypal inheritance

EXIT TICKET QUESTIONS

1. Will we have future exercises where we write the HTML file in addition to editing the JS file?
2. What's Vanilla JS?
3. why would you create html elements in vanilla js if you can just type it in the html file
4. which resource I can use outside the class to learn more about DOM.

HOMEWORK REVIEW

HOMEWORK — GROUP DISCUSSION

EXERCISE

TYPE OF EXERCISE

- Groups of 3

TIMING

4 min

1. Share your solutions for the homework.
2. Share one thing you found challenging. If you worked it out, share how; if not, brainstorm with your group how you might approach it.

EXERCISE — CATCH PHRASE

EXERCISE

TYPE OF EXERCISE

- Pairs

TIMING

5 min

1. Describe the term on one of your slips of paper **without saying the term itself** until your partner guesses the term.
2. Take turns so everyone gets a chance to give clues.

EVENTS

EVENT LISTENERS

selecting element

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

element reference

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

EVENT LISTENERS

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

method to add event listener

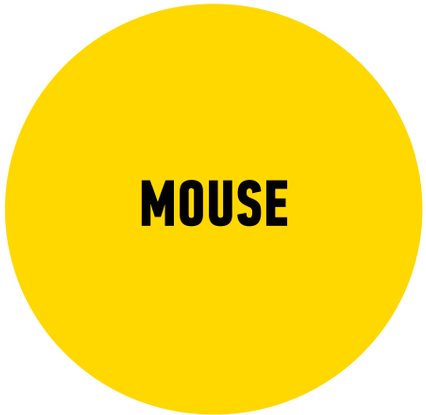
```
button.addEventListener('click', () => {
  // your code here
});
```

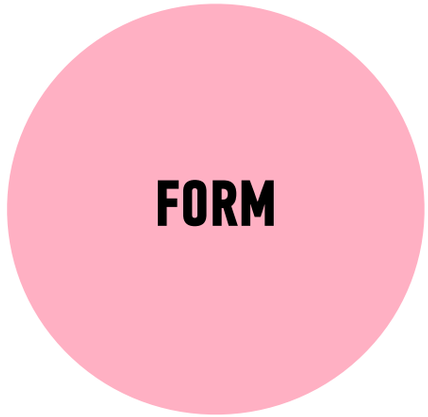
EVENT LISTENERS

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

type of event

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





click

dblclick

mouseenter

mouseleave

keypress

keydown

keyup

submit

change

focus

blur

resize

scroll

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

EVENT LISTENERS

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

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

function to run when event is triggered

EVENT LISTENERS

element reference

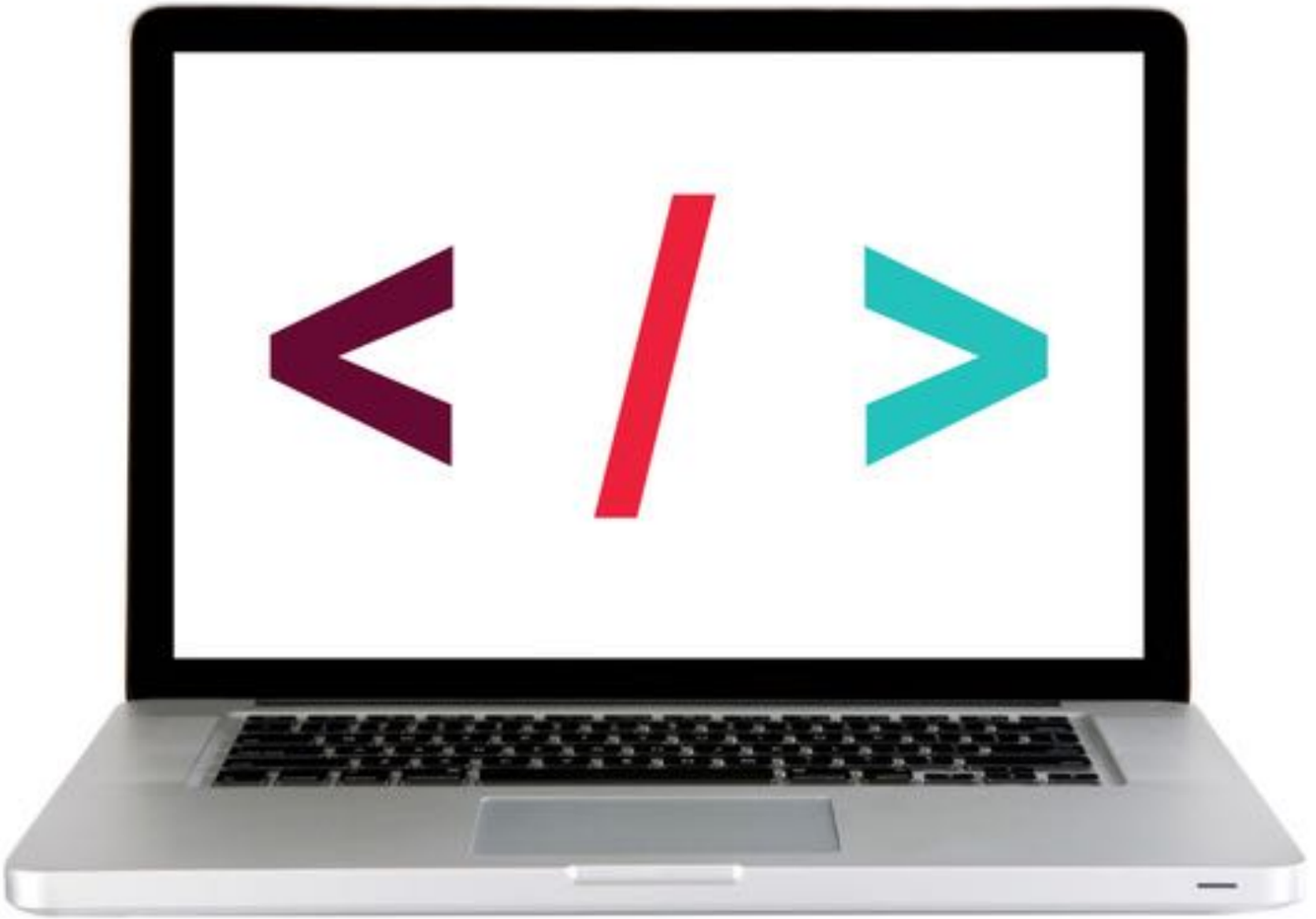
method to add event listener

type of event

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

function to run when event is triggered

LET'S TAKE A CLOSER LOOK



ACTIVITY

KEY OBJECTIVE

- Create DOM event handlers using vanilla JavaScript

TYPE OF EXERCISE

- Individual/Partner

TIMING

8 min `01-events-exercise`

1. Add event listeners to the 3 buttons at the top of the page.
2. Clicking each button should hide the block below it with the corresponding color.
3. Use handout/slides as a guide for syntax

WORKING WITH EVENT OBJECTS

`preventDefault()`

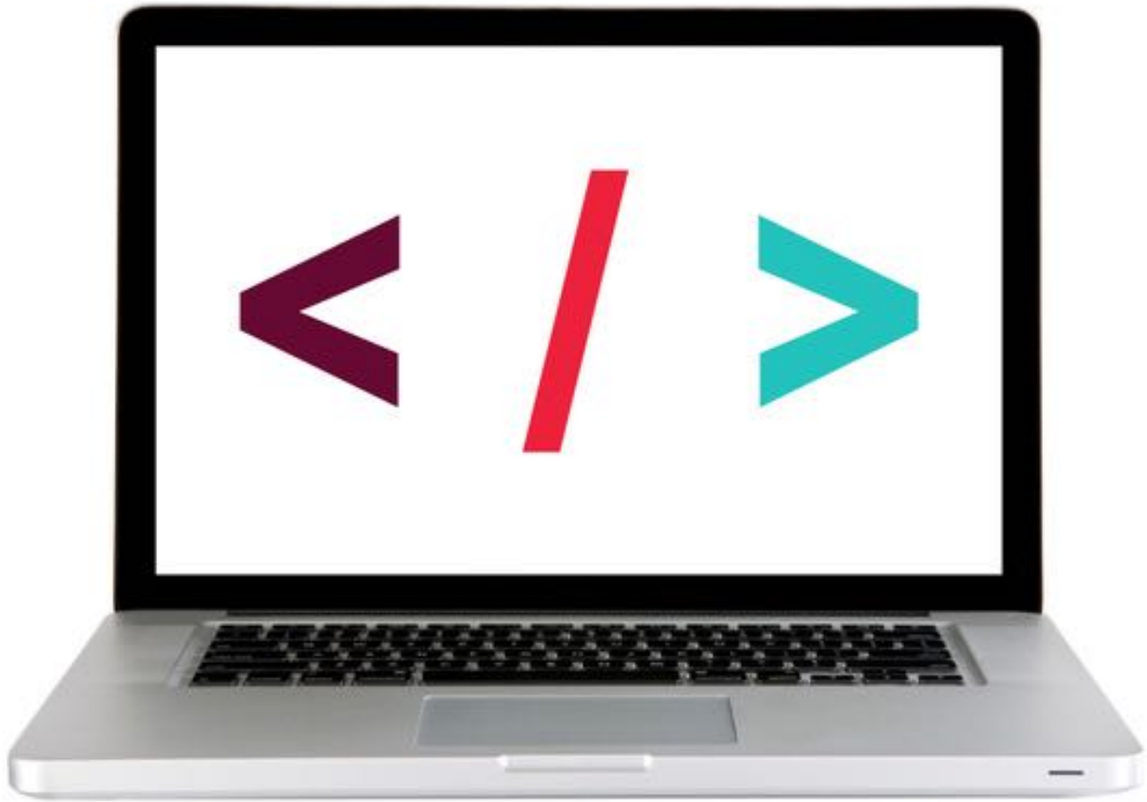
- Prevents element from executing default behavior in response to an event

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.on('click',(event) => {
  event.preventDefault();
  ...
)};
```

EVENTS & JQUERY



LET'S TAKE A CLOSER LOOK

EXERCISE

LOCATION

▸ `starter-code > 03-event object—exercise`

TIMING

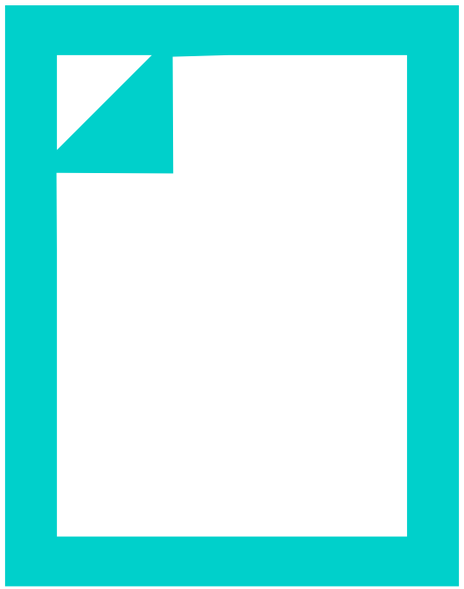
2 min

1. Update the code to prevent the form from submitting when the button is clicked.
2. Test your code in the browser and check the URL to verify that the form is not being submitted.

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').textContent = "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

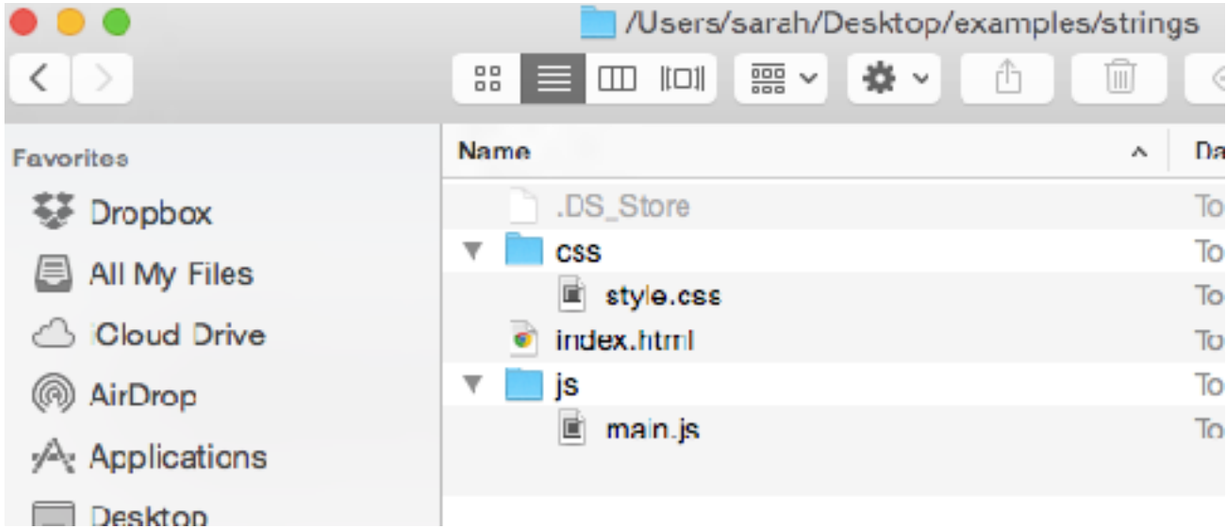
PURE JAVASCRIPT

- 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*).

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>
```

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!!!!

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>
```

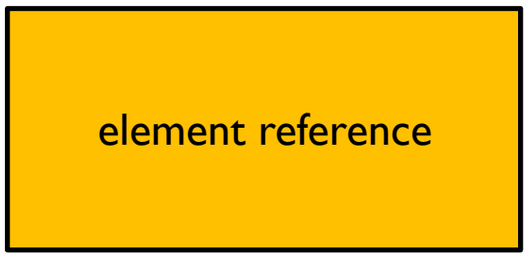
JQUERY OBJECTS

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

vanilla JavaScript selector

`querySelector('selector')`

plain element reference

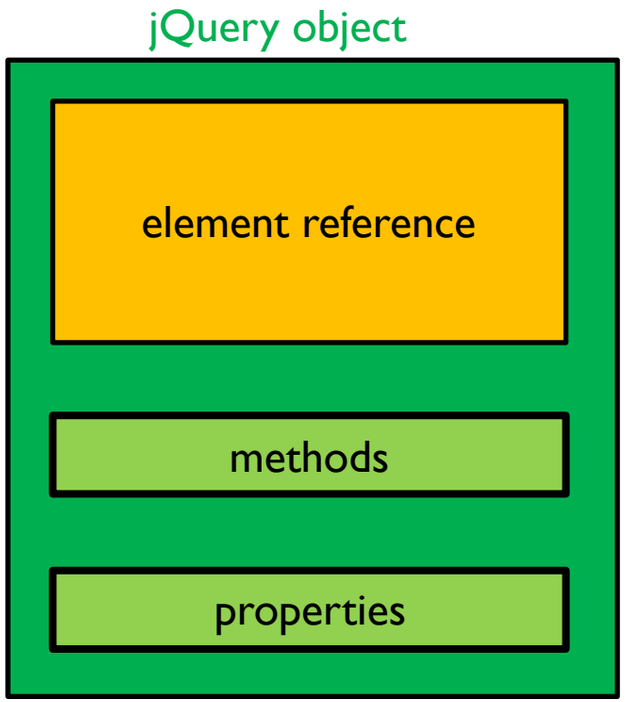


JQUERY OBJECTS

- 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

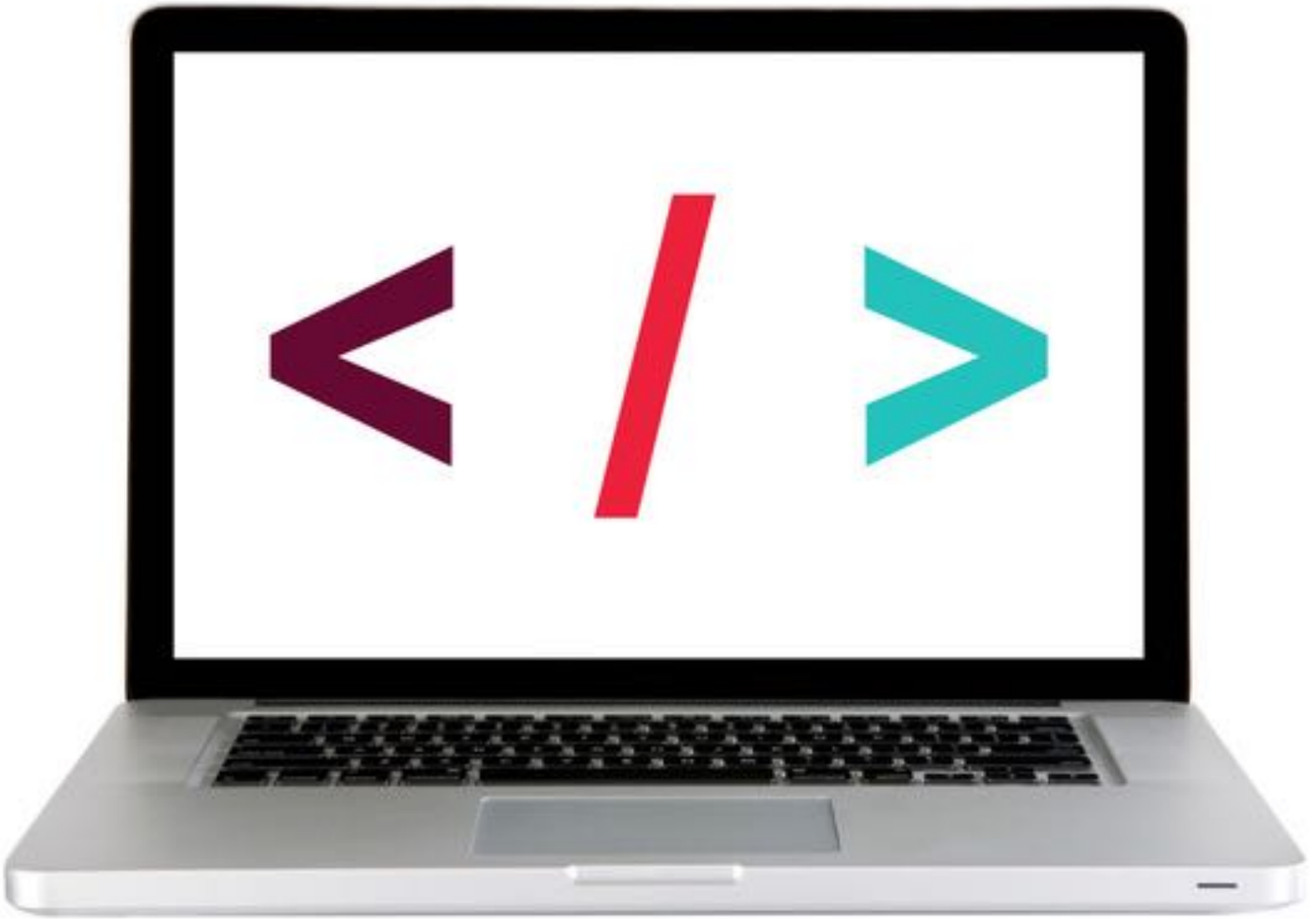
include $ at start of variable name to indicate that its value is a jQuery object

```
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

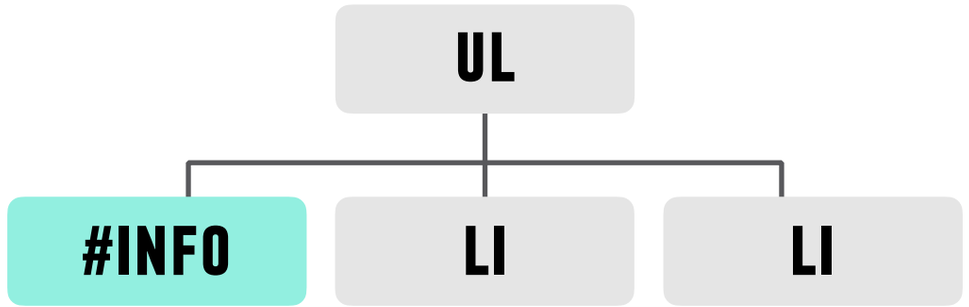
JQUERY METHODS — WORKING WITH THOSE ELEMENTS

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

See your handout or the jQuery docs for list!

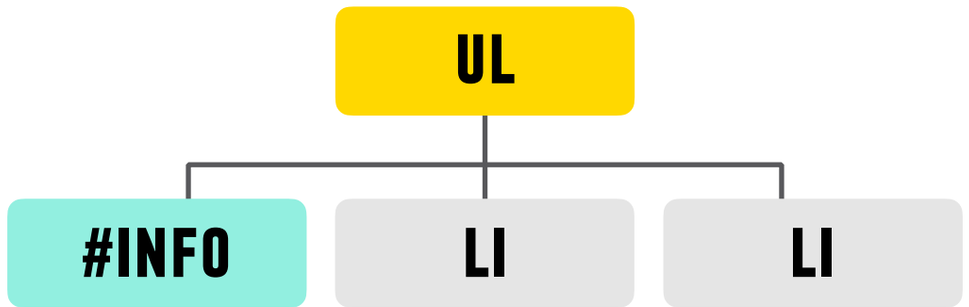
TRAVERSING THE DOM?

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



TRAVERSING THE DOM?

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



JQUERY METHODS — TRAVERSING THE DOM

FIND ELEMENTS

- 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

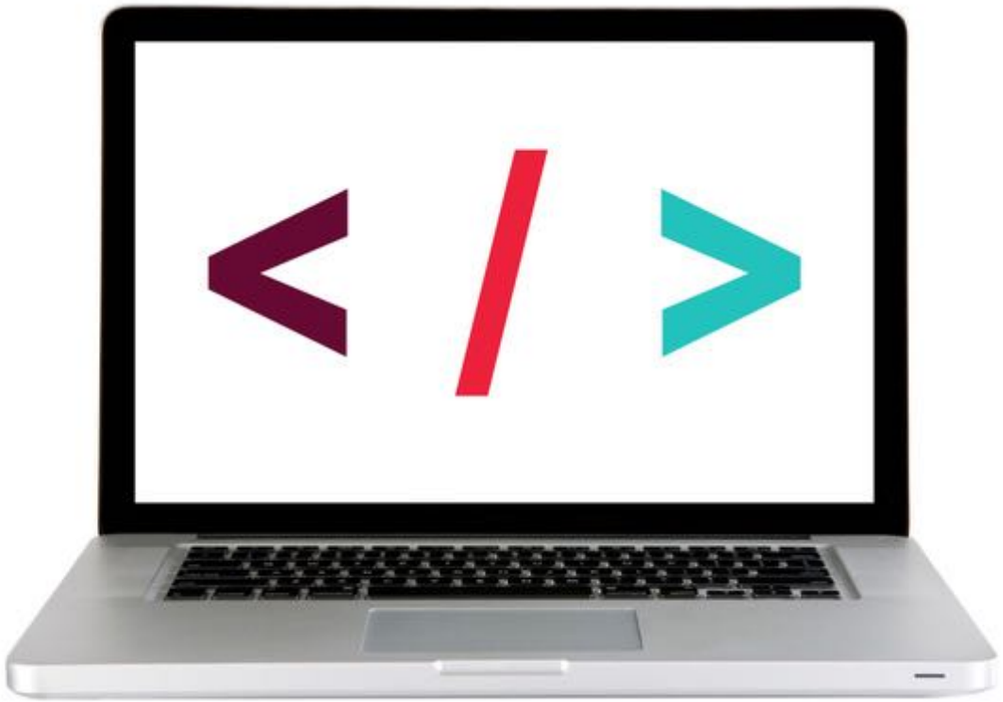
GET/SET CONTENT

Get/change content of elements and attributes

METHODS	EXAMPLES
.html()	$(‘h1').html('Content');
.text()	$(‘h1’).text('Just text content!’);
.attr()	$('img').attr('src', ‘images/bike.png');

What goes in the parentheses?
The **content** 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.

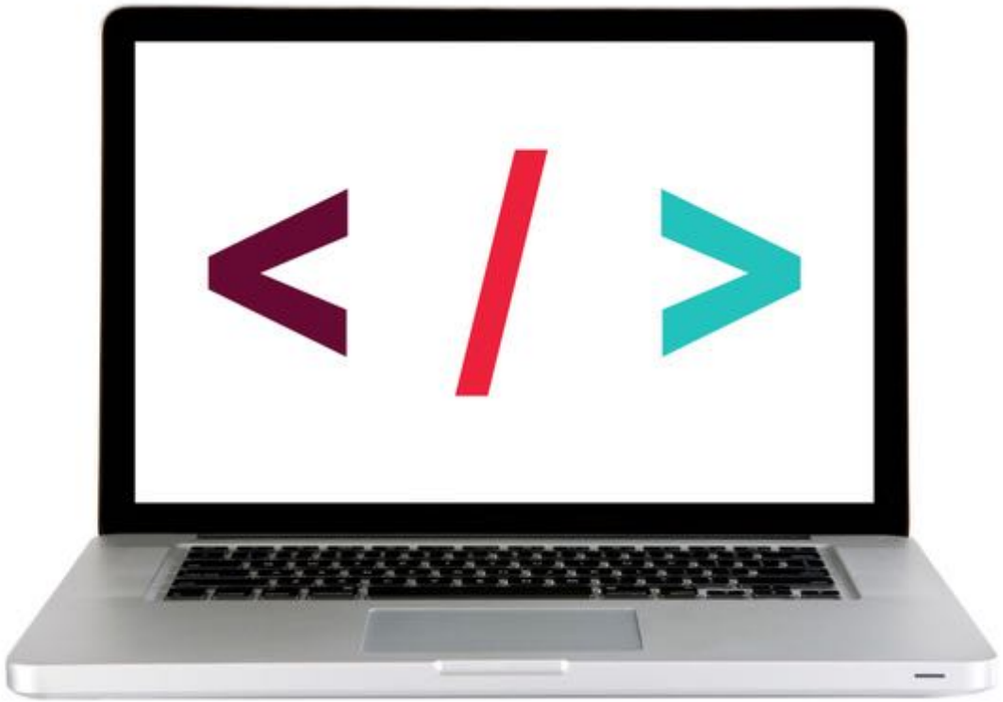
JQUERY METHODS — GETTING/SETTING CONTENT

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 `05-jquery-statements-exercise`

1. Follow the instructions under part 1 in main.js
2. Use handout/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

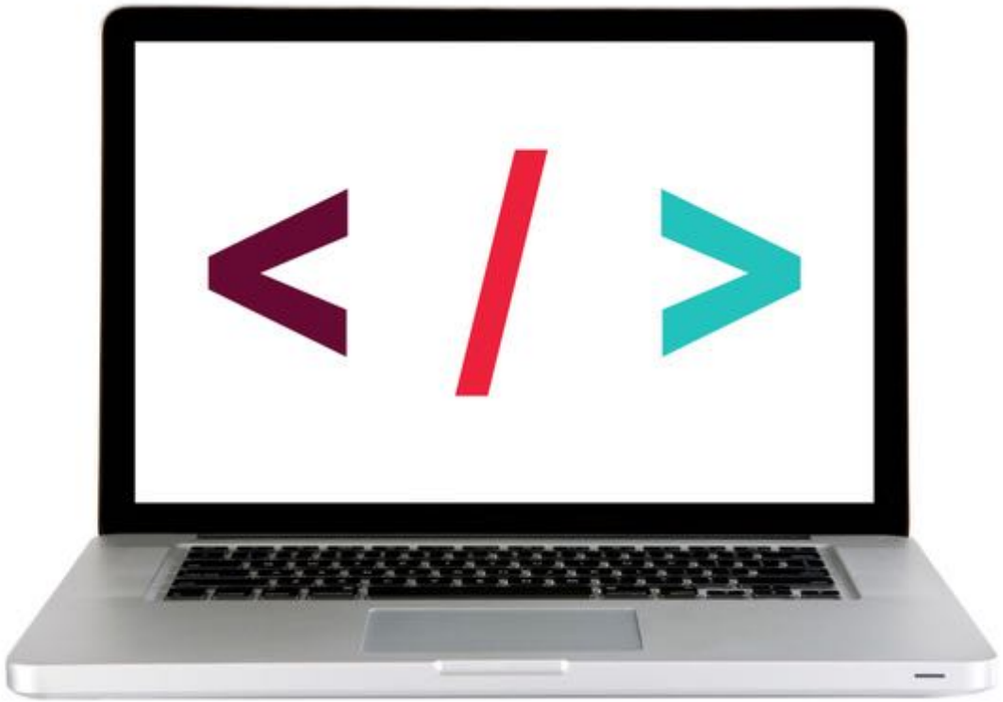
ADD EFFECTS/ ANIMATION

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!

EVENTS

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', () => {
  // your code here
});
```

JQUERY METHODS — EVENTS!

CREATE EVENT LISTENERS

method for all events

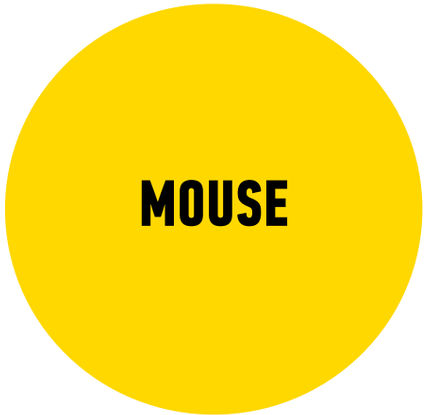
```
$('li').on('click', () => {
  // your code here
});
```

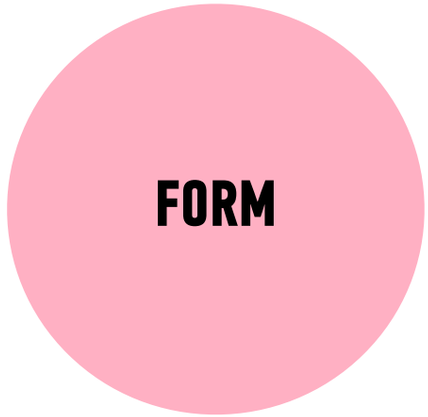
JQUERY METHODS — EVENTS!

CREATE EVENT LISTENERS

type of event

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





click

dblclick

mouseenter

mouseleave

keypress

keydown

keyup

submit

change

focus

blur

resize

scroll

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

JQUERY METHODS — EVENTS!

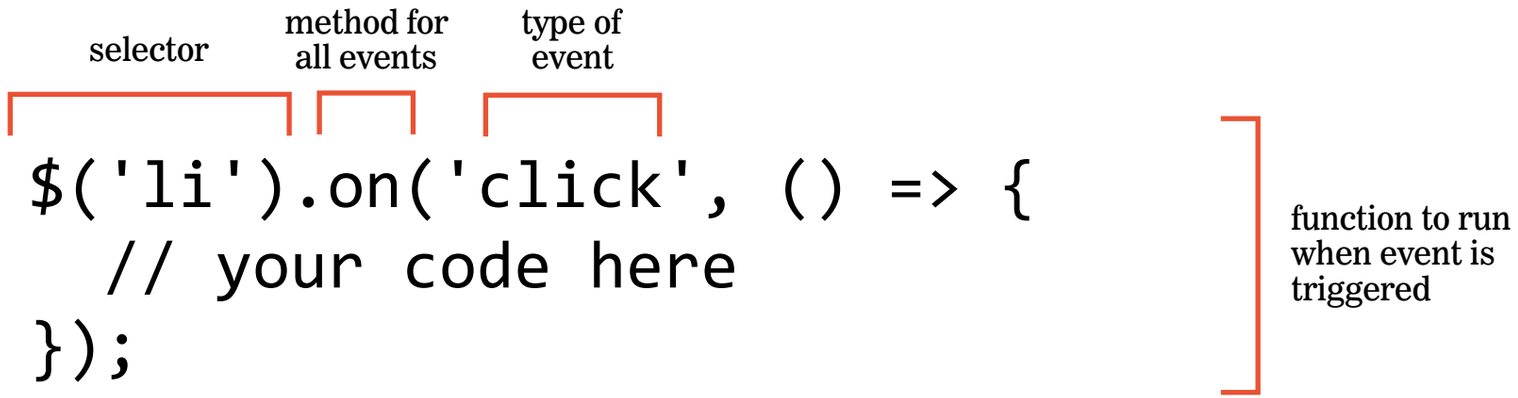
CREATE EVENT LISTENERS

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

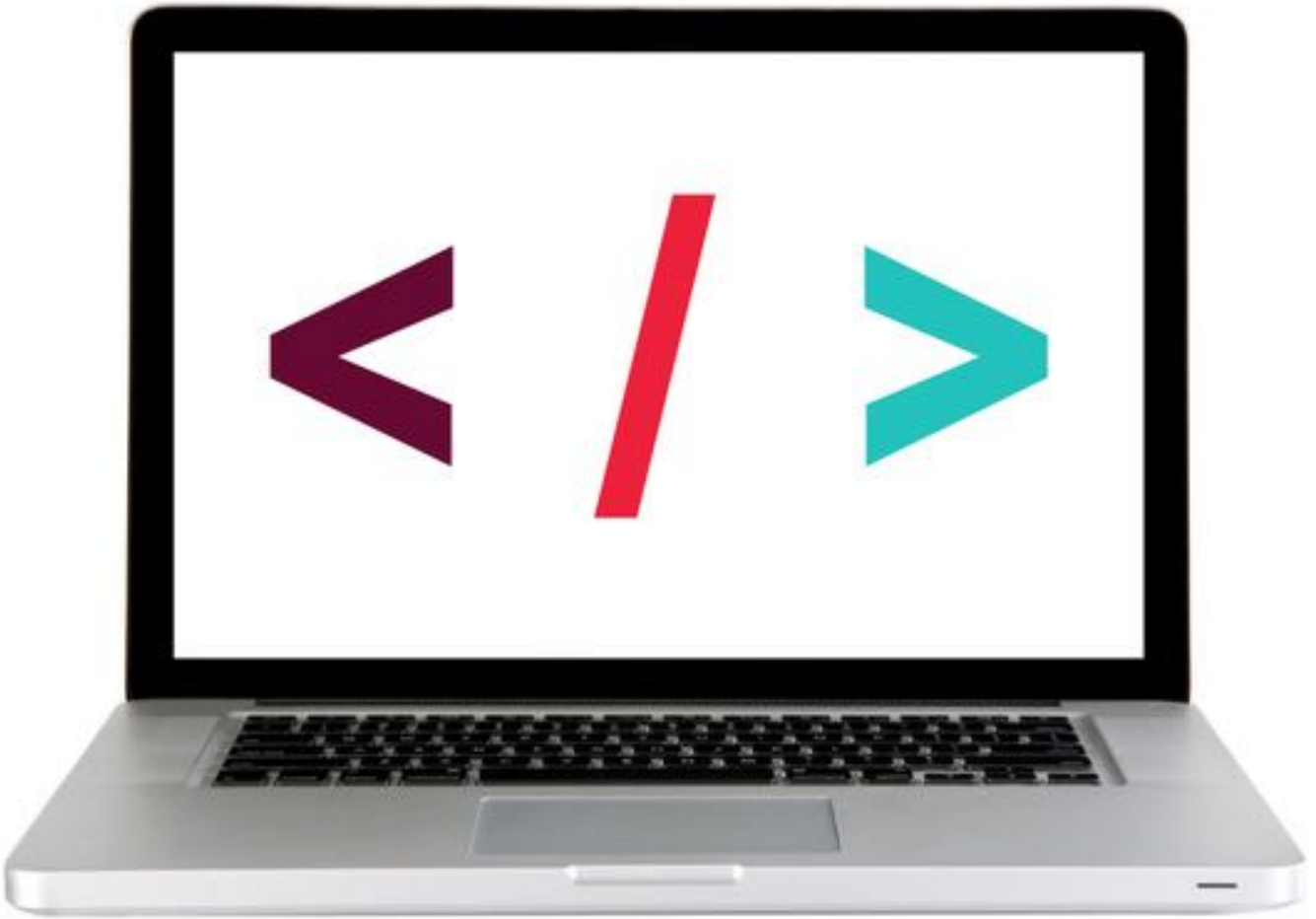
function to run when event is triggered

JQUERY METHODS — EVENTS!

CREATE EVENT LISTENERS



LET'S TAKE A LOOK



ACTIVITY

EXERCISE

KEY OBJECTIVE

- Utilize jQuery to access and manipulate DOM elements.

TYPE OF EXERCISE

- Individual/Partner

TIMING

5 min Continue with `05-jquery-statements-exercise`

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

CREATING & APPENDING DOM NODES WITH JQUERY

document.ready()

- specifies code to run only after the DOM has finished loading
- Syntax:

```
$(document).ready(() => {
  // code goes here
});
```

- Shorthand version (best practice):

```
$(() => {
  // code goes here
});
```

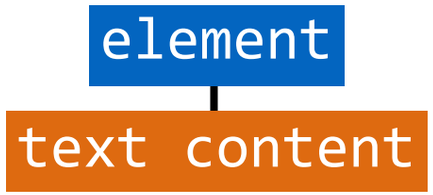
Adding content to the DOM

1. create a new element with `$('<element>')`

`element`

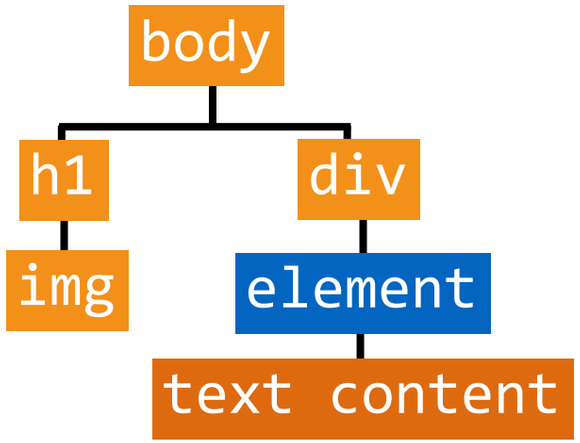
Adding content to the DOM

1. create a new element with `$('<element>')`
2. add new content to that element with a method like `.text()`, `.html()`, or `.attr()`



Adding content to the DOM

1. create a new element with `$('<element>')`
2. add new content to that element with a method like `.text()`, `.html()`, or `.attr()`
3. attach the new element to the DOM with `.append()`



$('<element>')

- Creates a new element

```
$('<li>'); // creates an li element
```

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

```
let item1 = $('<li>');
let item2 = $('<li>');
```

`.text()` or `.html()`

- Creates and adds text content as the child of an element
- Easiest to add method to same statement that creates element

```
let item1 = $('<li>').text('banana');
let item2 = $('<li>').text('apple');
```

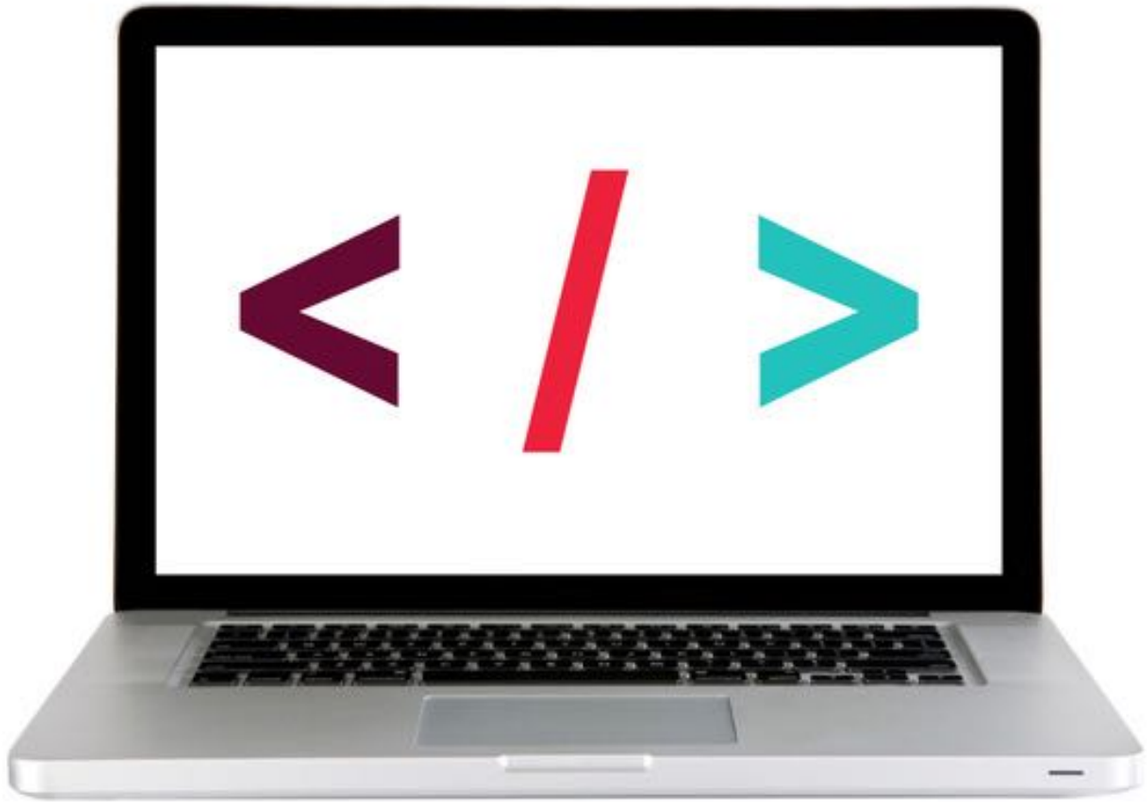
```
let item1 = $('<li>').html('<strong>Every</strong> dinosaur');
let item2 = $('<li>').html('Books (<em>not</em> ebooks)');
```

.append()

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

```
const list = $('ul');  // selects ul element
list.append(item1);    // adds item1 li to list ul
list.append(item2);    // adds item2 li to list ul
```

EVENTS & JQUERY



LET'S TAKE A CLOSER LOOK

JQUERY BEST PRACTICES

METHOD CHAINING

CHAINING

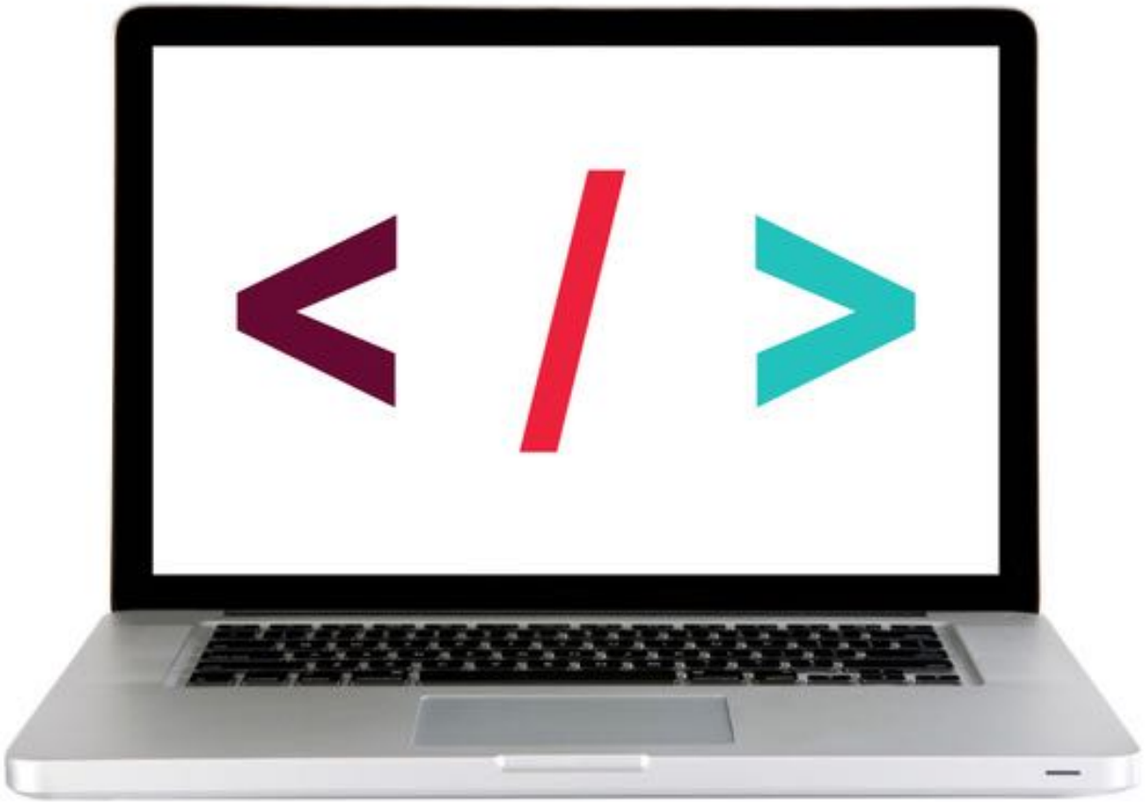
without chaining:

```
let $mainCaption = $('<p>');
let $captionWithText = $mainCaption.text('Today');
let $fullCaption = $captionWithText.addClass('accent');
```

with chaining:

```
let $fullCaption = $('<p>').text('Today').addClass('accent');
```

ADVANCED JQUERY



LET'S TAKE A CLOSER LOOK

IMPLICIT ITERATION

ADVANCED JQUERY

IMPLICIT ITERATION

explicit iteration

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

jQuery `.each()` method works like a `forEach` loop

not necessary for
element collections

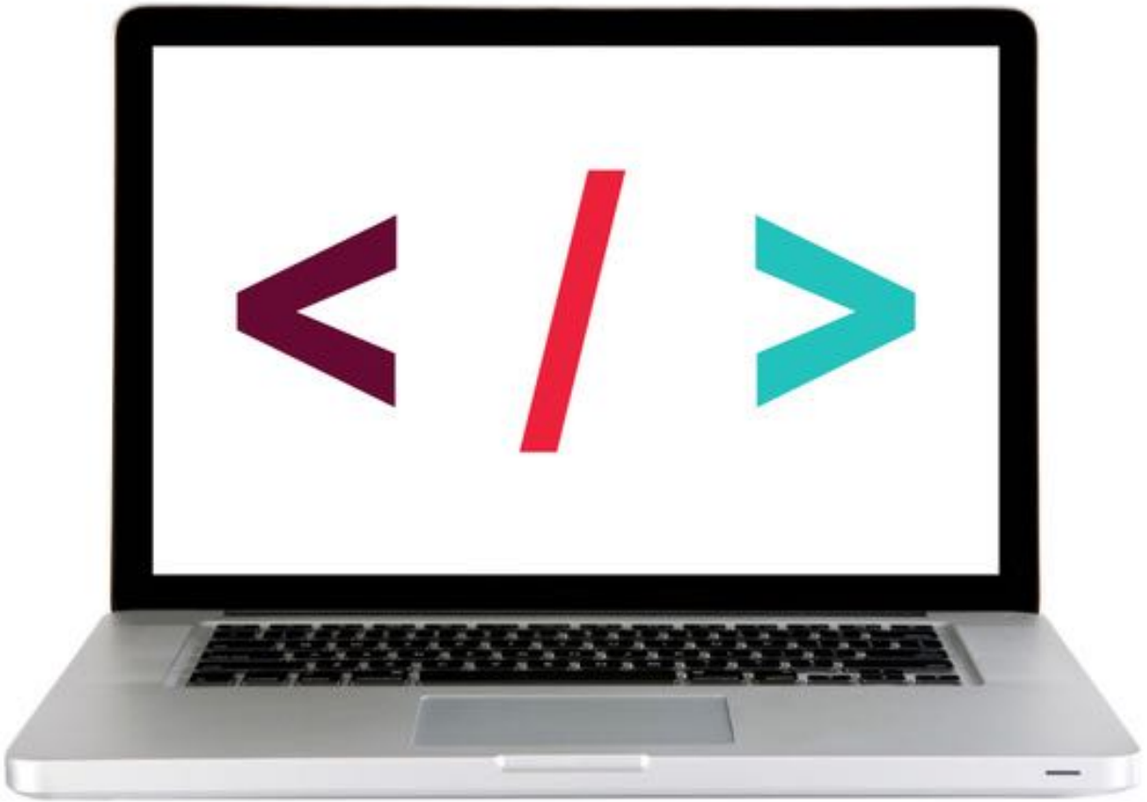
implicit iteration

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

applying any method to a jQuery collection iterates through each element!

less code = best practice!

ADVANCED JQUERY



LET'S TAKE A CLOSER LOOK

EXERCISE - IMPLICIT ITERATION

OBJECTIVES

- Use chaining to place methods on selectors.
- Use implicit iteration to update elements of a jQuery selection.

LOCATION

- `starter-code > 09-best-practices-exercise`

TIMING

5 min

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

EVENT DELEGATION

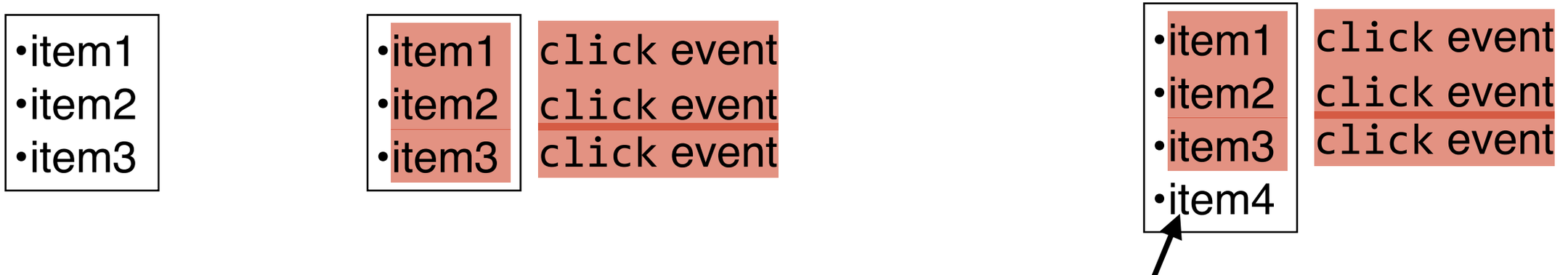
WITHOUT EVENT DELEGATION

1. load page

2. set event listener on list items

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

3. add a new list item



WITH EVENT DELEGATION

1. load page

- item1
- item2
- item3

2. set event listener on *parent of* list items

selector changed to parent

new second argument specifies children

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

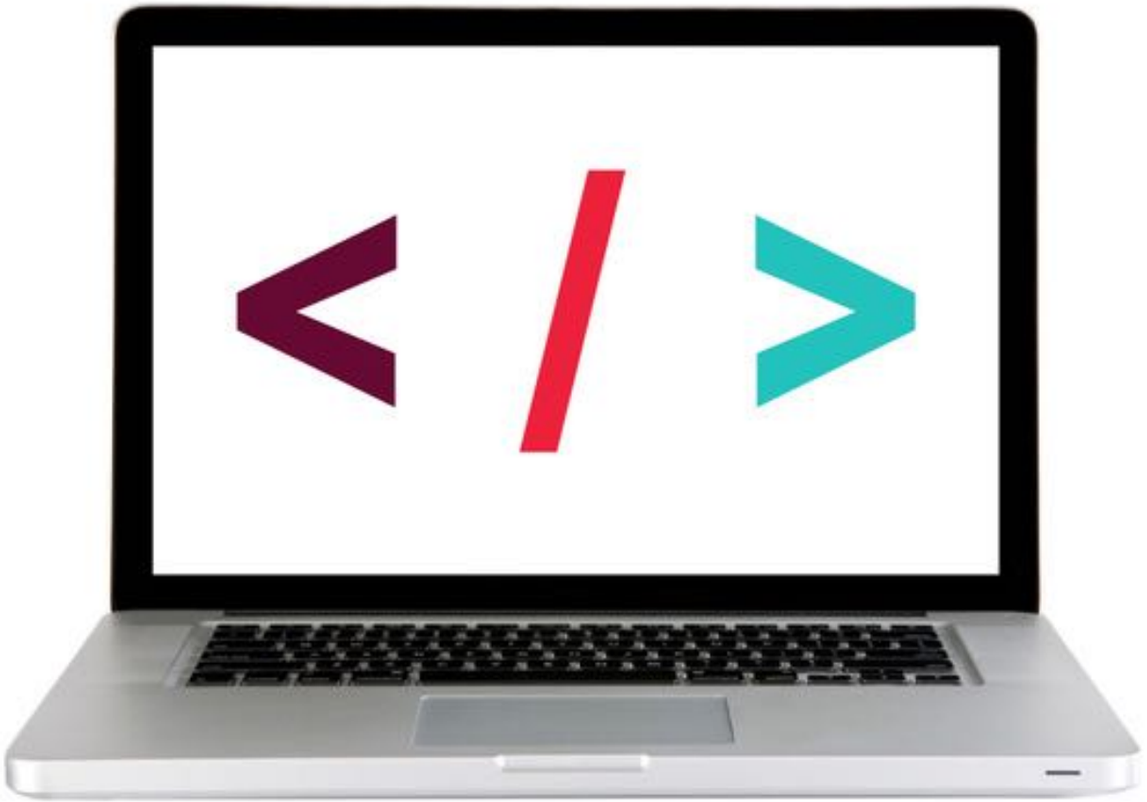
•item1	click event
•item2	click event
•item3	click event

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!

ADVANCED JQUERY



LET'S TAKE A CLOSER LOOK

EXERCISE - EVENT DELEGATION

OBJECTIVE

- Use event delegation to manage dynamic content.

LOCATION

- `starter-code > 09-best-practices-exercise`

TIMING

10 min

1. Return to main.js in your editor and complete item 4.
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 1: When the user mouses over each item, the item should turn grey. Don't use CSS hovering for this.
4. BONUS 2: Add another link, after each item, that allows you to delete the item.

ATTACHING MULTIPLE EVENTS WITH A SINGLE ON() STATEMENT

ATTACHING MULTIPLE EVENTS WITH A SINGLE .ON() STATEMENT

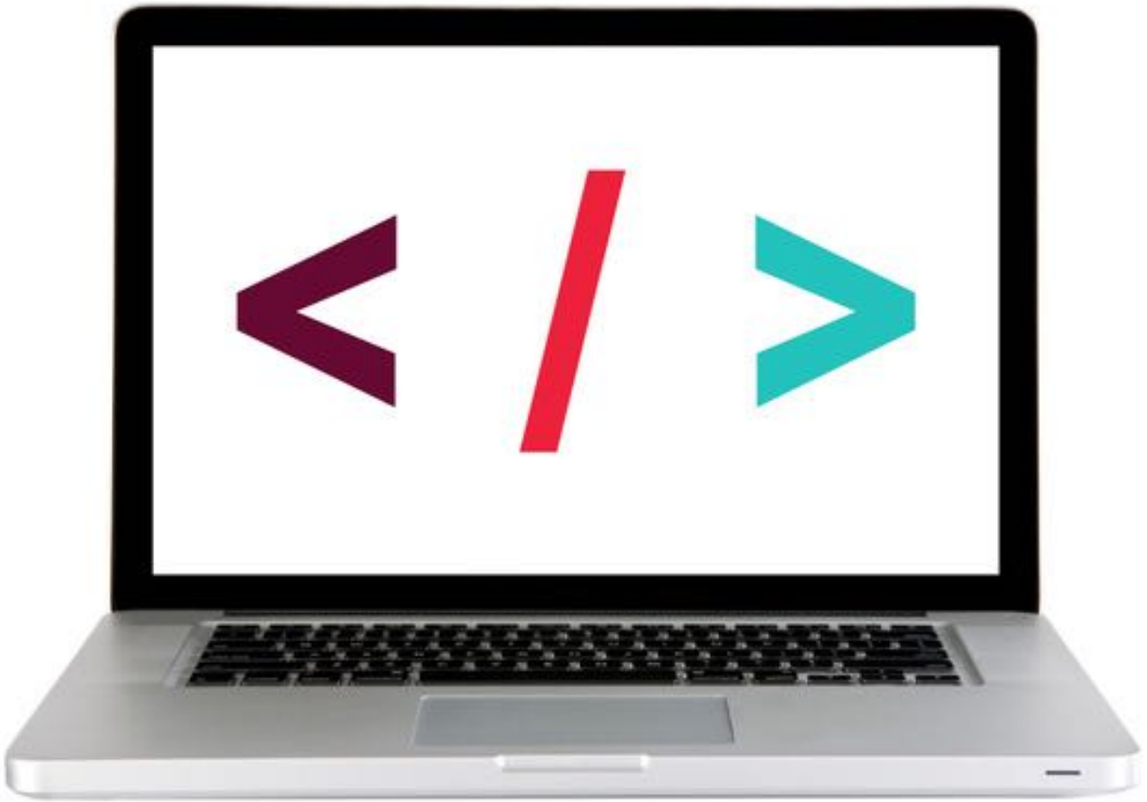
- We could write a separate `.on()` statement for each event on an element:

```
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

```
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');
  }
});
```

ADVANCED JQUERY



LET'S TAKE A CLOSER LOOK

EXERCISE - ATTACHING MULTIPLE EVENTS

LOCATION

▸ `starter-code > 10-multiple-events-exercise`

TIMING

4 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.

Exit Tickets!

(Class #8)

LEARNING OBJECTIVES - REVIEW

- Create DOM event handlers using vanilla JavaScript.
- Select DOM elements and properties using jQuery.
- Manipulate the DOM by using jQuery selectors and functions.
- Use event delegation to manage dynamic content.
- Use implicit iteration to update elements of a jQuery selection

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.

Q&A