GENERAL ASSEMBLY

JAVASCRIPT DEVELOPMENT

Sasha Vodnik, Instructor

HELLO!

1. Pull changes from the `svodnik/JS-SF-14-resources` repo to your computer
2. Open the `08-advanced-jquery` folder in your code editor

JAVASCRIPT DEVELOPMENT

ADVANCED JQUERY

LEARNING OBJECTIVES

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

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

AGENDA

- Events with jQuery
- Adding DOM nodes with jQuery
- Event delegation
- Implicit iteration

WEEKLY OVERVIEW

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

EXIT TICKET QUESTIONS

1. How to select the descendant of a descendant using jquery.
2. Is Jquery so accepted that vanilla JS manual work is not as needed? I understand knowing it but is it accepted to know the shortcuts?
3. The coding work alongs are a bit fast paced sometimes, not sure if I should work through them with the instructor or just watch
4. What are some common real-world practices with JQuery?

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 — 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:

FIND ELEMENTS

GET/SET CONTENT

ADD EFFECTS/ ANIMATION

CREATE EVENT LISTENERS

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('<strong>Content</strong>');
.text()	$(‘h1’).text('Just text content!’);
.attr()	$('img').attr('src', ‘images/bike.png');

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

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 — 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

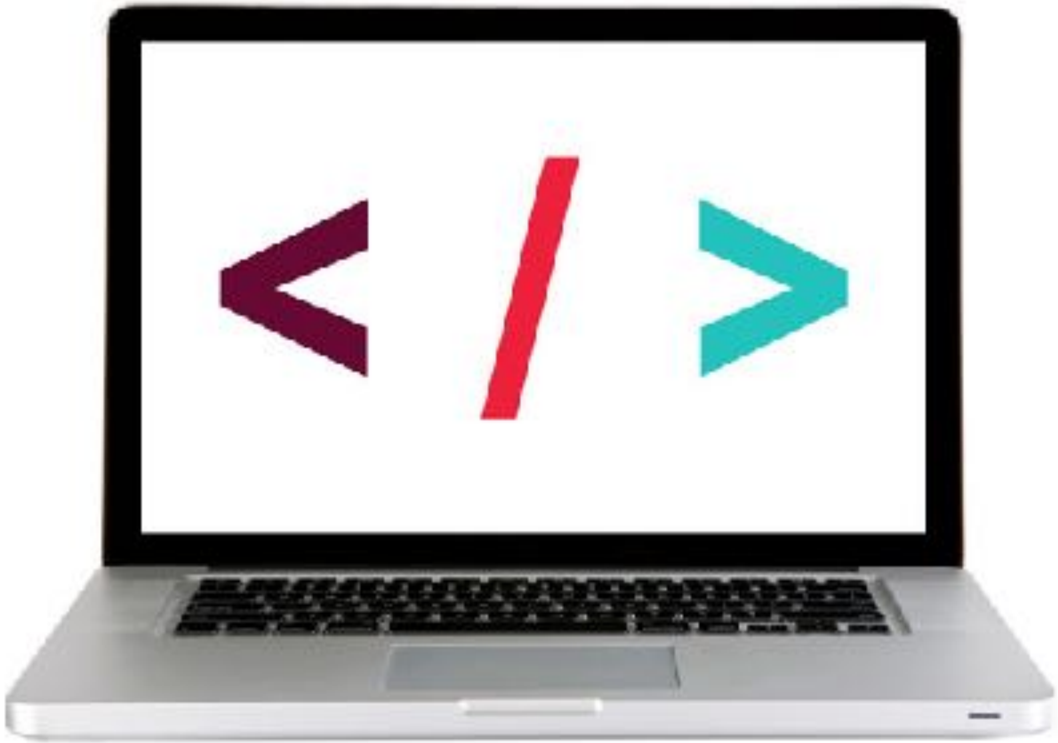
ADD EFFECTS/ ANIMATION

Add effects and animation to parts of the page

METHODS	EXAMPLES
`.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



ACTIVITY

EXERCISE

KEY OBJECTIVE

- Utilize jQuery to access and manipulate DOM elements.

TYPE OF EXERCISE

- Individual/Partner

TIMING

5 min Continue with `01-jquery-statements-exercise`

1. Follow the instructions under Part 2 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:

FIND ELEMENTS

GET/SET CONTENT

ADD EFFECTS/ ANIMATION

CREATE EVENT LISTENERS

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', 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
});
```

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!

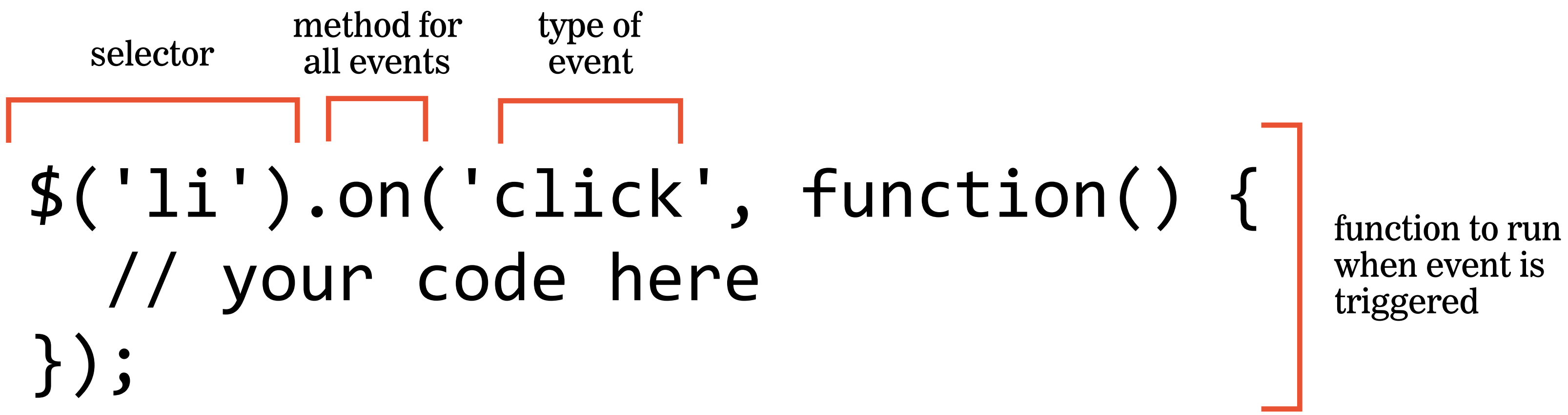
CREATE EVENT LISTENERS

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

function to run when event is triggered

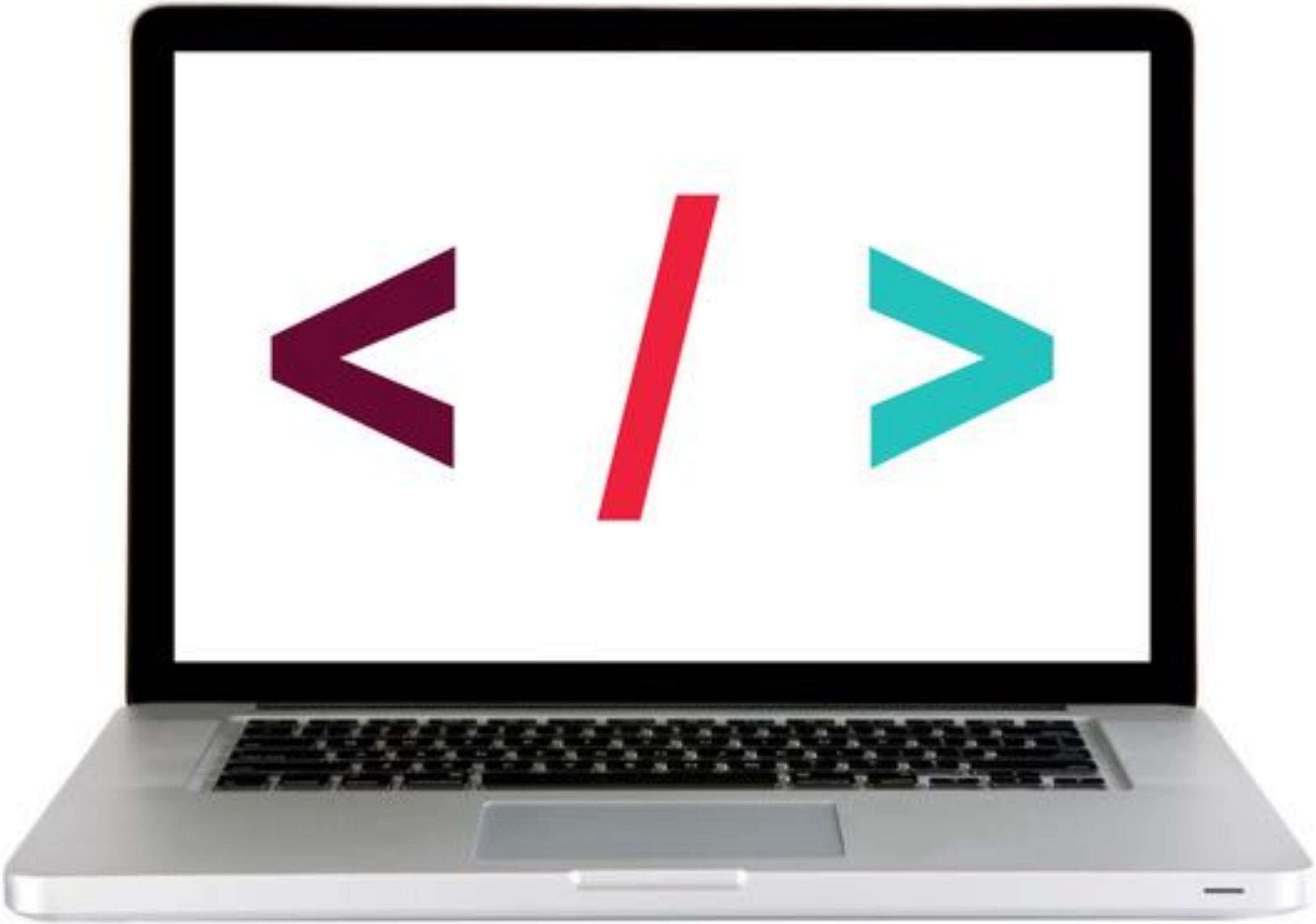
JQUERY METHODS — EVENTS!

CREATE EVENT LISTENERS



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

LET'S TAKE A LOOK



ACTIVITY

EXERCISE

KEY OBJECTIVE

- Create DOM event handlers to respond to user actions

TYPE OF EXERCISE

- Individual/Partner

AS A CLASS

6 min `03-jquery-events-exercise` folder

1. Rewrite the vanilla JavaScript code to use jQuery instead.
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(function() {
  // code goes here
});
```

- Shorthand version (best practice):

```
$(function() {
  // code goes here
});
```

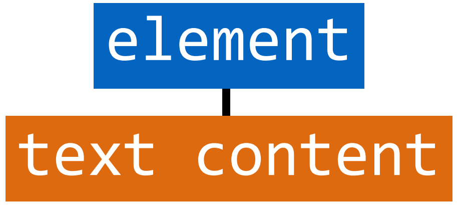
Adding content to the DOM

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

element

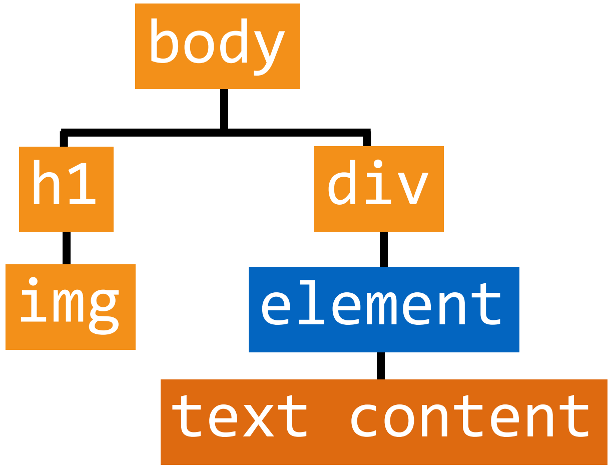
Adding content to the DOM

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



Adding content to the DOM

1. create a new element with `$(‘<element>’)`
2. add new content to that element with `.text()` or `.html()`
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’);
```

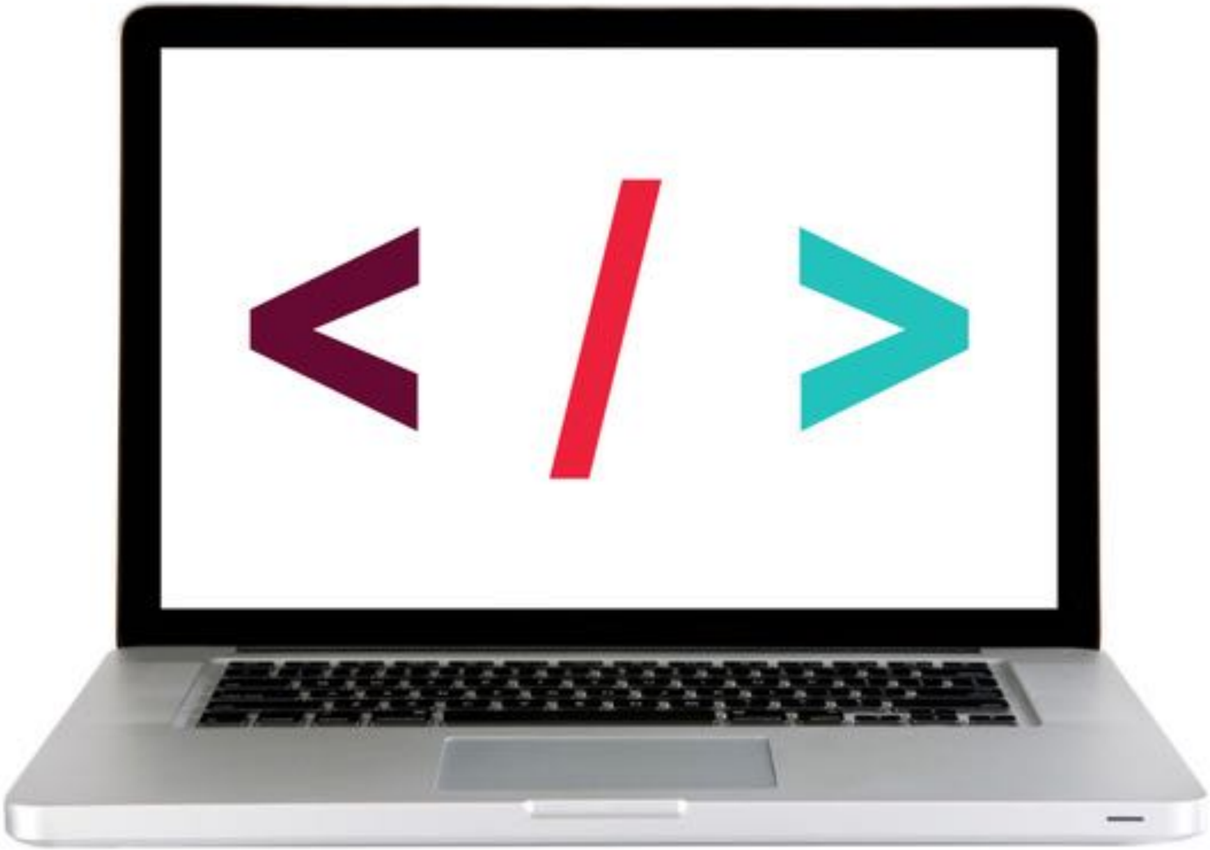
```
let item1 = $(‘<li>’).html(‘<strong>Every</strong> dinosaur’);
let item2 = $(‘<li>’).html(‘Books (<em>not</em> ebooks)’);
```

.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

EXERCISE - ADD CONTENT TO A WEB PAGE USING JQUERY

EXERCISE

LOCATION

- `starter-code > 05-create-append—exercise`

TIMING

10 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.

WORKING WITH EVENT OBJECTS

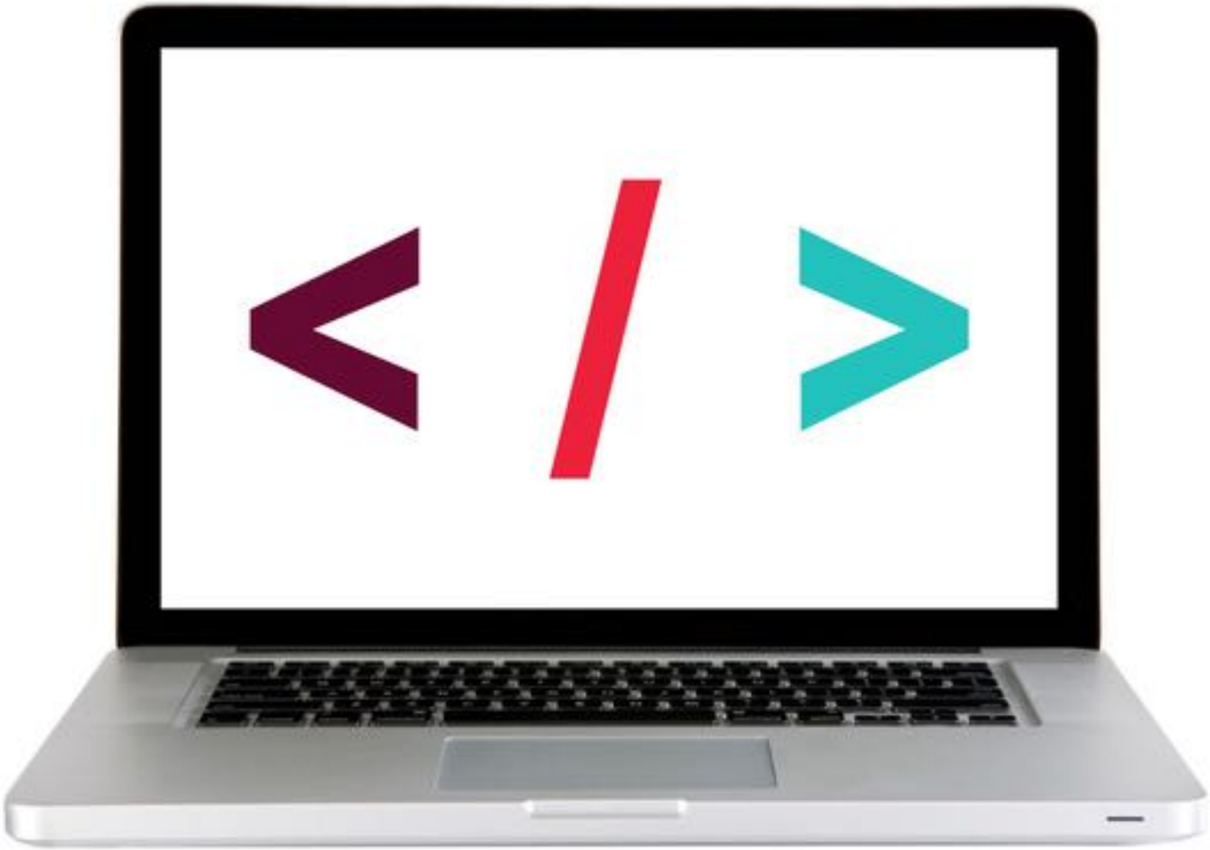
`preventDefault()`

- 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’, function(event) {
  event.preventDefault();
  ...
)};
```



LET'S TAKE A CLOSER LOOK

EXERCISE

EXERCISE

LOCATION

▸ `starter-code > 07-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.

EXERCISE

EXERCISE

OBJECTIVE

- Manipulate the DOM and create DOM event handlers using jQuery

LOCATION

- `starter-code > 08-bottles-lab`

TIMING

15 min

1. Use jQuery to write the lyrics of 99 Bottles of Beer to the browser window.
2. Start by creating both an HTML file and a JavaScript file.
    - In your HTML file, you'll want to include an unordered list (`<ul>`) that will contain all of your lyrics.
    - Each line of the song should appear inside of a list item (`<li>`) within that unordered list.
3. BONUS: Refactor your code to replace all jQuery with vanilla JavaScript.

BEST PRACTICES

METHOD CHAINING

CHAINING

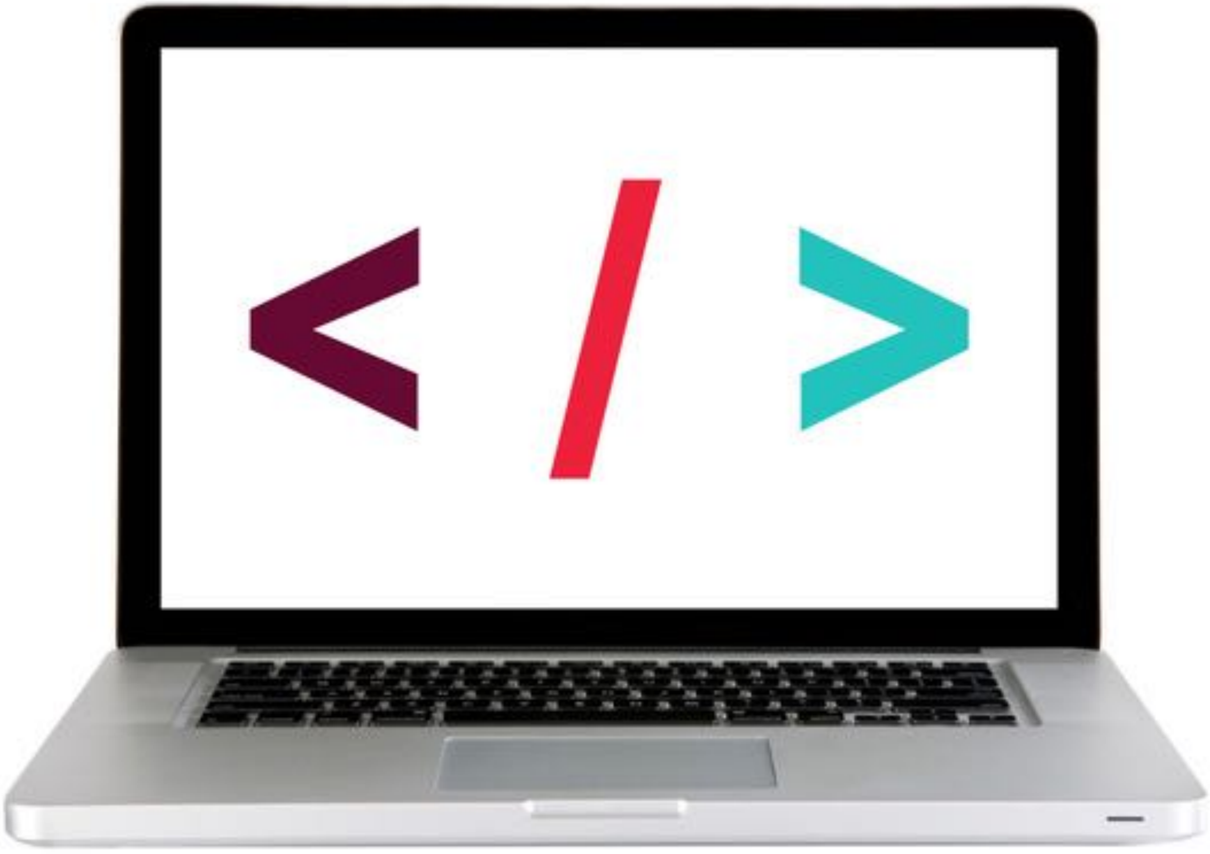
without chaining:

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

with chaining:

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

ADVANCED JQUERY



LET'S TAKE A CLOSER LOOK

EXERCISE – CHAINING

EXERCISE

OBJECTIVE

- Use chaining to place methods on selectors.

LOCATION

- `starter-code > 10-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.

IMPLICIT ITERATION

IMPLICIT ITERATION

explicit iteration

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

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

✗ not necessary for element collections

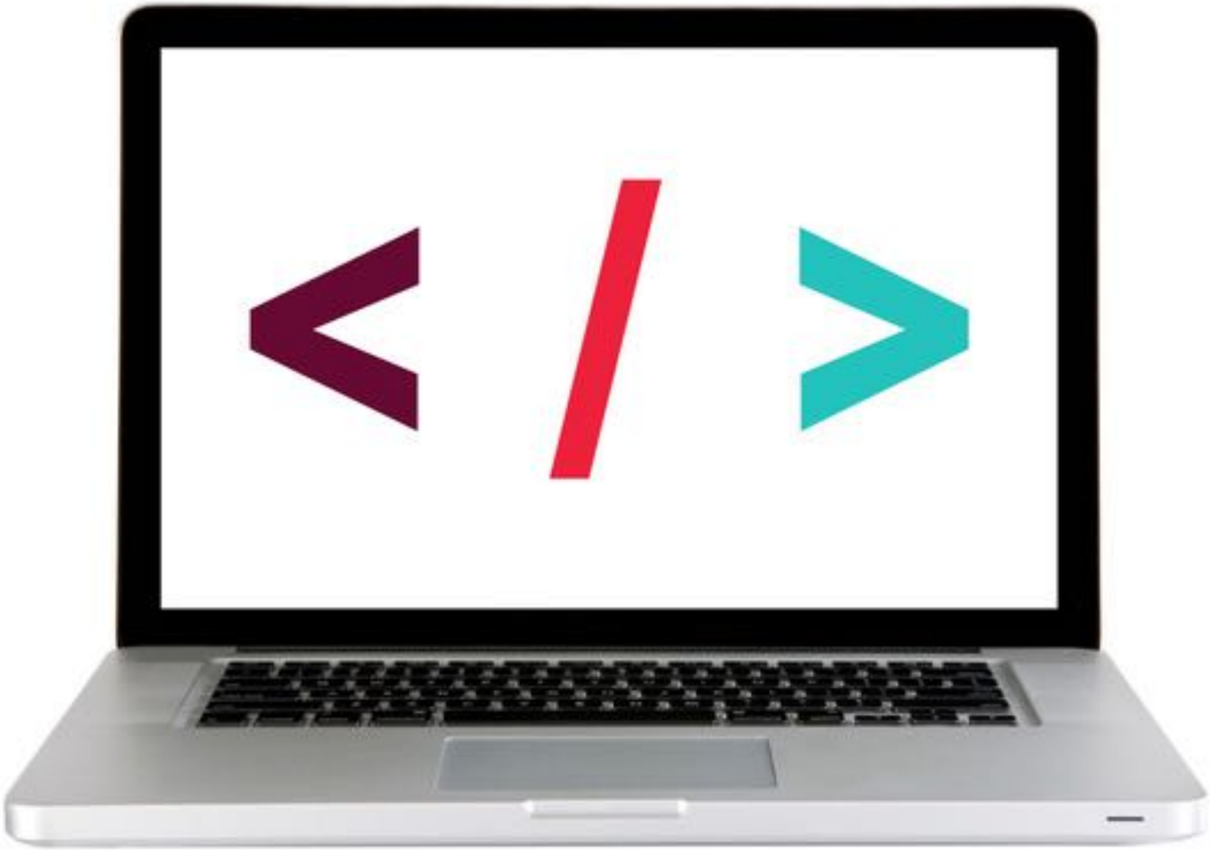
implicit iteration

```
$(‘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

EXERCISE

OBJECTIVE

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

LOCATION

- `starter-code > 10-best-practices-exercise`

TIMING

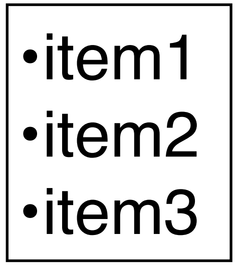
5 min

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.

EVENT DELEGATION

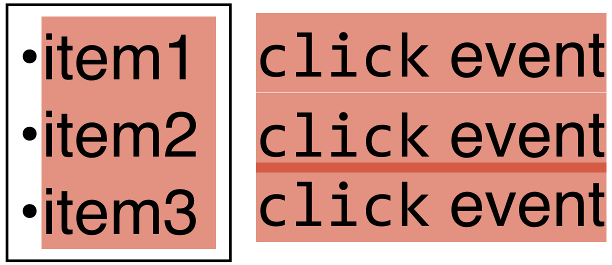
WITHOUT EVENT DELEGATION

1. load page

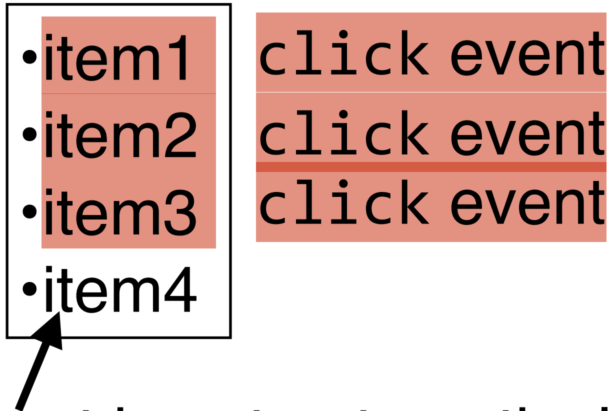


2. set event listener on list items

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

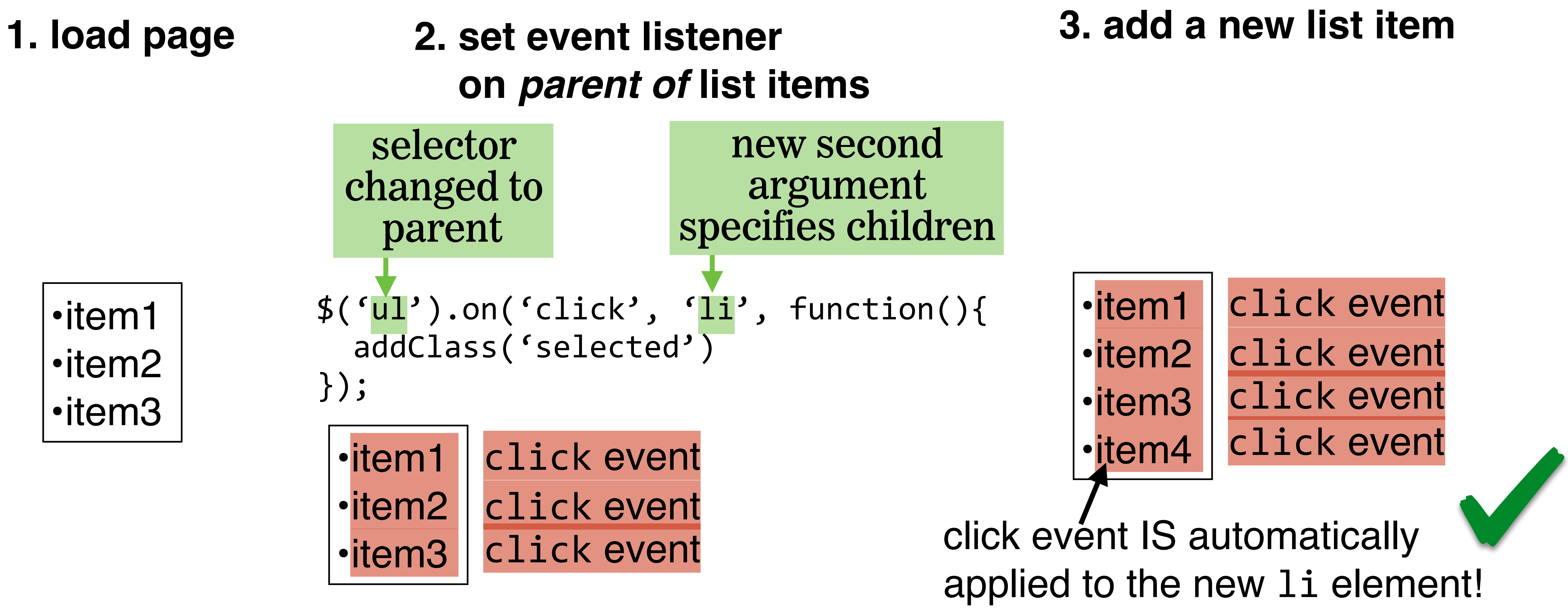


3. add a new list item



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 to parent

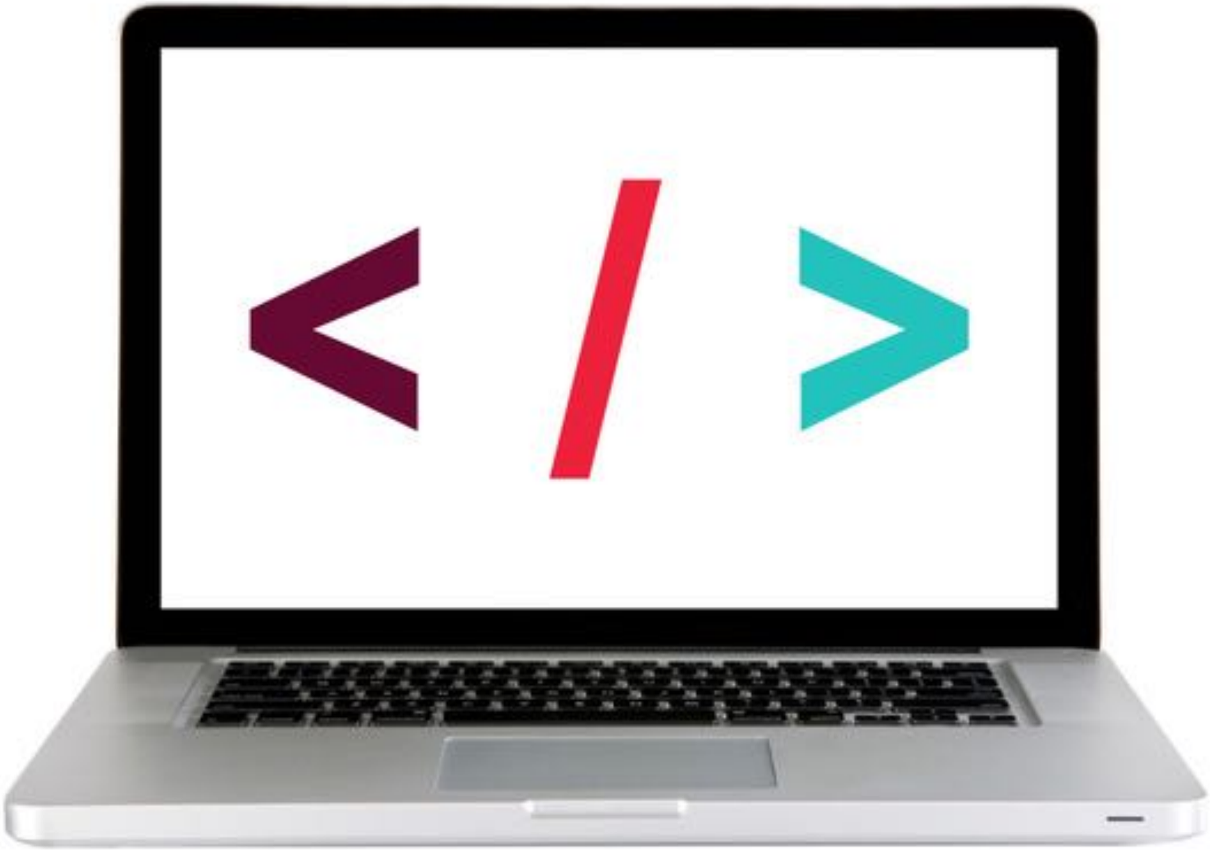
new second argument specifies children

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

3. add a new list item

click event IS automatically applied to the new `li` element! ✓

ADVANCED JQUERY



LET'S TAKE A CLOSER LOOK

EXERCISE – EVENT DELEGATION

EXERCISE

OBJECTIVE

- Use event delegation to manage dynamic content.

LOCATION

- `starter-code > 10-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

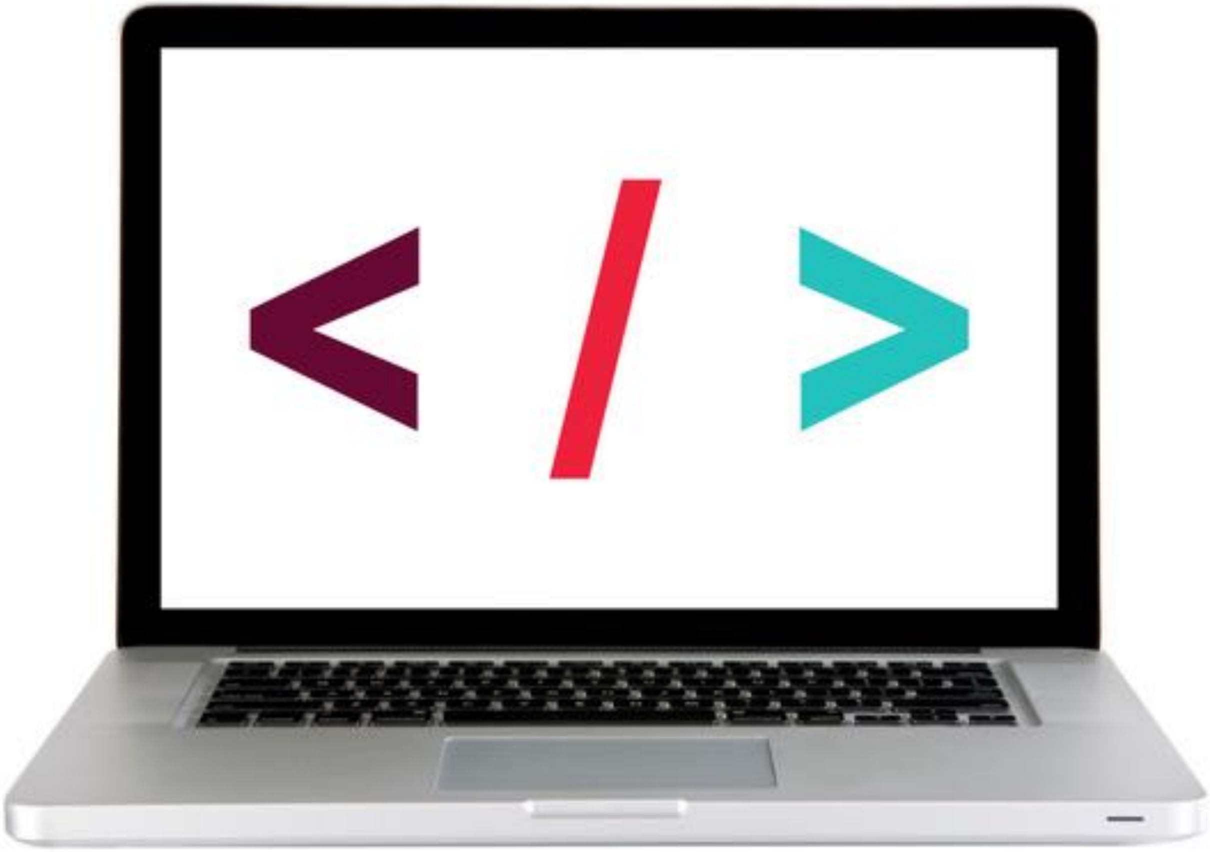
- We could write a separate `.on()` statement for each event on an element:

```
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

EXERCISE

LOCATION

▸ `starter-code > 11-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.

Exit Tickets!

(Class #8)

LEARNING OBJECTIVES - REVIEW

- Manipulate the DOM by using jQuery selectors and functions.
- Create DOM event handlers using jQuery.
- 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