GENERAL ASSEMBLY

JAVASCRIPT DEVELOPMENT

Sasha Vodnik, Instructor

HELLO!

1. Pull changes from the `svodnik/jsd6` repo to your computer
2. Open the `starter-code` folder in your code editor

INTRO TO JQUERY

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.
- Implement advanced jQuery events
- Use event delegation to manage dynamic content.
- Use implicit iteration to update elements of a jQuery selection, and use chaining to place methods on selectors.

AGENDA

- jQuery intro
- jQuery lab
- jQuery events
- Event delegation

EXIT TICKET FEEDBACK

- “Today felt rushed.”
- “Don’t feel prepared for homework right now.”
- “Walking around/music during the exercises is VERY distracting to me”
- “It’s exciting to be making things happen in the browser!!”

WEEKLY OVERVIEW

WEEK 5	holiday / Intro to jQuery
WEEK 6	Intro to APIs / Asynchronous JS & callbacks
WEEK 7	Advanced APIs / Project 2 Lab

Checkin and questions

- The **most significant thing I’ve learned** about the DOM is ___________.
- My **biggest outstanding question** about the DOM is ___________.

EXERCISE — CATCH PHRASE

EXERCISE

TYPE OF EXERCISE

- Groups of 3

TIMING

5 min

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

HOMEWORK REVIEW

REVIEW

EVENTS

- Commonly used events
  - `load`
  - `click`
- full list at https://developer.mozilla.org/en-US/docs/Web/Events

`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();
  ...
}
```

Think about events that can trigger a change in a web page.

- Events you’ve experienced (such as `click`)
- Events you imagine would be useful
- Events you’ve heard of (even if you don’t know exactly how they work)

JQUERY

WHAT IS JQUERY?

- jQuery is a JavaScript library.
- It provides an application programming interface (API)
- Makes it easier to write code to select elements, manipulate the DOM, and perform other tasks.
- jQuery is the most widely used JavaScript library on the web

HOW IS JQUERY USEFUL?

- jQuery lets us select DOM elements using CSS syntax
- We can do this with JavaScript using the following methods:
    - `querySelector()`
    - `querySelectorAll()`
- But jQuery syntax is less verbose

JQUERY OBJECTS

- Selecting elements with jQuery returns a jQuery object
- This is essentially the results of our search, wrapped in an object with a set of properties and methods, which let you more easily
    - change styling
    - add event listeners for specific events
    - write brand new content into the web page
- You can also work with the results using array notation

HOW DO WE USE JQUERY?

- jQuery is stored in an external .js file just like any other JavaScript code we include in our web pages
- We reference it in a `script` element in our HTML, such as

```
<script src=“http://code.jquery.com/jquery-3.1.0.min.js”></script>
```

REFERENCE JQUERY FILE BEFORE YOUR OWN FILES!

- The jQuery code must be loaded before any other .js files that use jQuery syntax; otherwise, they will not work, so:

```
<script src="http://code.jquery.com/jquery-3.1.0.min.js"></script>
<script src="js/app.js"></script>
```

REFERENCING THE JQUERY LIBRARY

the jQuery file is loaded first, telling the browser how to interpret jQuery syntax

our JavaScript file uses jQuery syntax, so we load it after the jQuery JavaScript file

```
<html>
  <head>
 </head>
  <body>
    <h1>JavaScript resources</h1>
    <script src="jquery-3.1.0.js"></script>
    <script src="script.js"></script>
  </body>
</html>
```

SELECTING AN ELEMENT WITH JQUERY

- Simply specify the CSS selector within $( )

```
$(‘#item’) // select item by id
$(‘.open’) // select item(s) by class
$(‘h2’)    // select item(s) by tag
```

CREATING VARIABLES WITH JQUERY

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

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

CREATE A NEW ELEMENT WITH JQUERY

```
var $summary = $(‘<p>’);
```

- Even when an element takes a closing tag, we only have to specify the opening tag to create an instance of that element with jQuery

EXERCISE — JQUERY METHODS

EXERCISE

TYPE OF EXERCISE

- Pairs

TIMING

2 min	1. Look up your method on http://api.jquery.com 2. Be prepared to describe what your method does

CREATE AND APPEND A TEXT NODE

- .text()

```
$summary.text(“It all comes down to this.”);
```

APPEND AN ELEMENT

- `.append()`

```
$section1.append($summary);
```

RUN CODE ONLY AFTER THE DOM HAS LOADED

```
$(document).ready(function() {
  ...
}
```

- More commonly used shorthand version:

```
$(function() {
  ...
}
```

SPECIFY AN EVENT HANDLER

- `.on()`

```
$button.on(‘click’, function(event) {

});
```

- more at https://learn.jquery.com/events/

WORK WITH CSS STYLES

- jQuery lets us access and change specific CSS styles for an element.
- However, the best practice is to do all work with CSS in the stylesheet (.css file) and use jQuery only to change the class value assigned to an element with `.addClass()` and `.removeClass()`

```
$phoneBox.addClass(‘placeholderText’);
$phoneBox.removeClass(‘placeholderText’);
```

CHANGE THE HTML CONTENT OF AN ELEMENT

- `.html()`

```
$summary.html(‘Everything you need to know.’);
```

```
$sidebar.html(‘<img src=“sparrow.jpg” alt=“a
sparrow”’);
```

EXERCISE — JQUERY

EXERCISE

OBJECTIVE

- Manipulate the DOM by using jQuery selectors and functions, and register and trigger event handlers for jQuery events.

TYPE OF EXERCISE

- Individual

TIMING

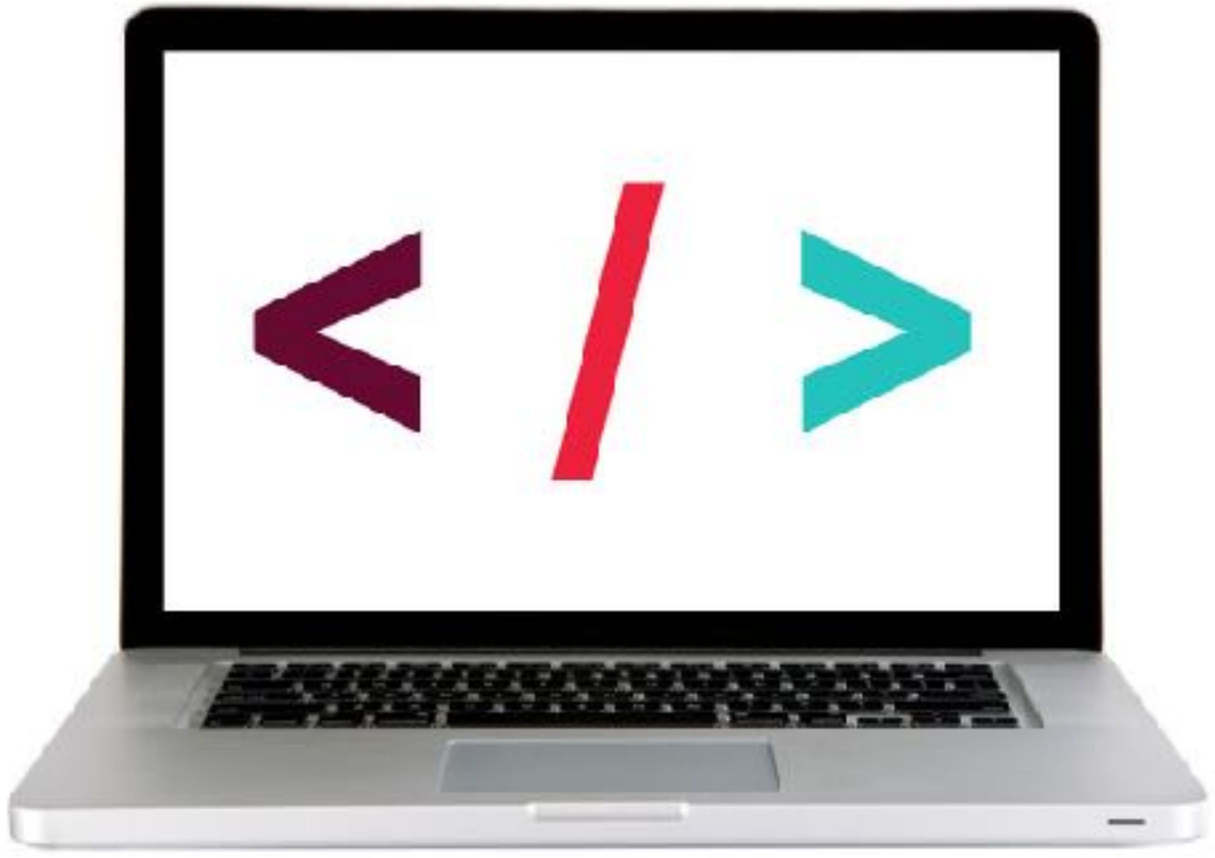
2 min

1. Write jQuery code to select the element with the ID value logo
2. Write jQuery code to select all p elements
3. Write jQuery code to specify an event handler for the click event for the element referenced by the $button variable

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

EXERCISE

OBJECTIVE

- Manipulate the DOM by using jQuery selectors and functions.

LOCATION

- `starter-code > 1-jquery—exercise`

TIMING

20 min

1. The starter code displays a to do list. You can type a new item in the box, then click the Create button to add the item to the list.
2. Using jQuery, add a "complete task" link at the end of each to-do item.
3. When clicked, the link will cross out the current item (hint: add a class to the list that sets the text-decoration property to line-through)
4. Each new item added by the user needs to also have the "complete task" link at the end.

JQUERY & EVENTS

DOM EVENTS WE'VE USED SO FAR

`load`	the page has finished loading
`click`	a mouse button has been clicked and released when the pointer is over an element

A SELECTION OF OTHER DOM EVENTS

Mouse Events

```
click
dblclick
mouseenter
mouseleave
```

Form Events

```
submit
change
focus
blur
```

Keyboard Events

```
keypress
keydown
keyup
```

Document/Window Events

```
load
resize
scroll
unload
```

EVENT OBJECT

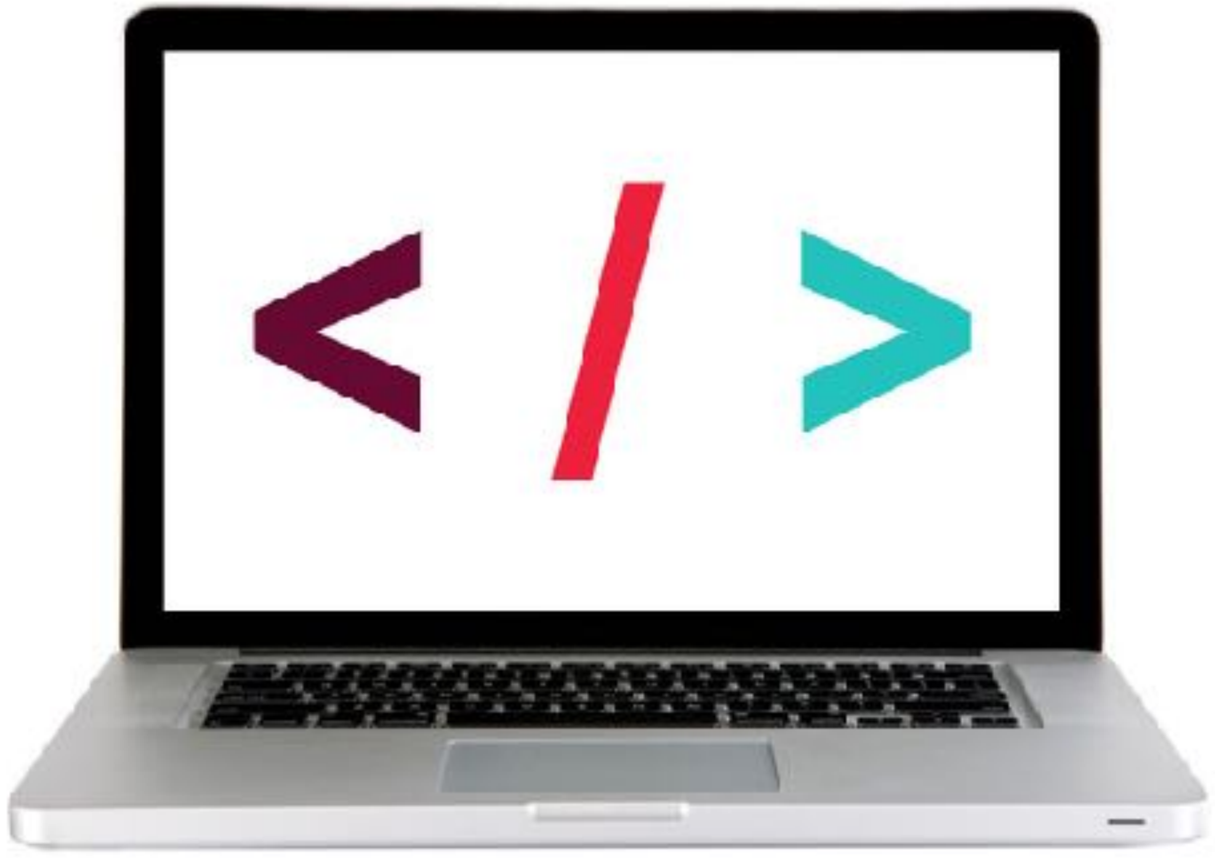
- Generic code to respond to an event:

```
$("#clickButton").on('click', function() {
  // do something
};
```

- We can modify this code to capture an object describing the event when it happens:

```
$("#clickButton").on('click', function(event) {
  // do something
  // can reference the event object with the parameter name
};
```

INTRO TO JQUERY



LET'S TAKE A CLOSER LOOK

BEST PRACTICES

CHAINING

- jQuery lets us attach one or more methods to a selector, so we can combine multiple actions into a single statement

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

becomes

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

EXERCISE - CHAINING

EXERCISE

OBJECTIVE

- Use chaining to place methods on selectors.

LOCATION

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

TIMING

5 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.

EXPLICIT ITERATION

- We can use the jQuery `.each()` method to iterate through a jQuery collection

```
var $listItems = $(‘li’); // collection
var $qed = $(‘<span>’).html('&there4;');
$listItems.each(function() {
 $(this).append($qed);
});
```

- This works just like a `for()` loop in vanilla JavaScript

IMPLICIT ITERATION

- On a selector that returns a jQuery collection, chain a method
- This method is applied iteratively to each element in the jQuery collection, *but without needing to explicitly write code that iterates*
- This is known as **implicit iteration**

```
var $listItems = $(‘li’); // collection
var $qed = $(‘<span>’).html('&there4;');
$listItems.append($qed);
```

EXERCISE - IMPLICIT ITERATION

EXERCISE

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 items 3 and 4.
2. In your browser, reload index.html and verify that the functionality is unchanged.

EVENT DELEGATION

- When the page loads, we can set events on a set of elements
- However, if we add a sibling element later, the event is not set on it

```
var $listItems = $(‘#contents-list li');

$listItems.on('mouseenter', function(event) {
  $(this).siblings().removeClass(‘active’);
  $(this).addClass('active');
});
```

EVENT DELEGATION

- We can ensure that events are attached to elements added to the DOM later by selecting the parent element and specifying the child elements within the `on()` method arguments
- This is known as **event delegation**

Selector changed from `‘#contents-list li’`

New argument `‘li’` added to `on()` method

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

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

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 5a and 5b.
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.

ATTACHING MULTIPLE EVENTS WITH A SINGLE EVENT HANDLER

- We could write a separate event handler 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 EVENT HANDLER

- Grouping all the events for an element in a single event handler 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');
  }
});
```

EXERCISE – ATTACHING MULTIPLE EVENTS

EXERCISE

LOCATION

▸ `starter-code > 5-multiple-events-exercise`

TIMING

10 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.

LEARNING OBJECTIVES - REVIEW

- Manipulate the DOM by using jQuery selectors and functions.
- Register and trigger event handlers for jQuery events.
- Implement advanced jQuery events
- Use event delegation to manage dynamic content.
- Use implicit iteration to update elements of a jQuery selection, and use chaining to place methods on selectors.

NEXT CLASS PREVIEW

Ajax and 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 vanilla JS.
- Implement a jQuery Ajax client for a simple REST service.
- Describe the benefits of separation of concerns – API vs. Client.

Exit Tickets!

Q&A