Intro to jQuery

An Easier Way?

Key Questions

- What is jQuery
- What is a library?
- Why would you use jQuery?
- Why would you not use jQuery?

What is jQuery?

jQuery is a DOM manipulation library

What is jQuery?

It comes with a bunch of useful methods to things like:

- Select Elements
- Manipulate Elements
- Create Elements
- Add Event Listeners
- Animate Elements
- Add Effects
- Make HTTP Requests(AJAX)

Why Use jQuery?

- Fixes "broken" DOM API
- Brevity and Clarity
- Ease of use
- Cross-Browser Support
- AJAX
- Lot's of people use jQuery!

Why Not Use jQuery?

- The DOM API is no longer "broken"
- It doesn't do anything you can't do on your own
- It's an unnecessary dependency
- Performance
- Lot's of people are moving away from jQuery!

Either way, it's worth knowing.

Adding jQuery

Download jQuery and link to it locally:

```
<script type="text/javascript" src="jquery.js"></script>
```

OR

Link to a CDN(a hosted copy)

```
<script type="text/javascript" src="https://code.jquery.com/jquery-2.1.4.js"></script>
```

Quick Preview

Here's a 30 second preview of what jQuery looks like:

```
//when a user clicks the button with id 'trigge
$('#trigger').click(function(){

    //change the body's background to yellow
    $('body').css("background", "yellow");

    //fade out all img's over 3 seconds
    $('img').fadeOut(3000, function() {

    //remove imgs from page when fadeOut is don
    $(this).remove();
  });
});
```

jQuery Selectors

Objectives

- Select elements with *$()*
- Use *.css()* to style elements

Selecting with jQuery

$("selectorGoesHere")

Selecting with jQuery is very similar to *querySelectorAll*, in that we provide a CSS style selector and jQuery will return all matching elements

Selecting with jQuery

$("selectorGoesHere")

We **select** and then manipulate

```
//to select all img tags
$("img")

//to select all elements with class 'sale'
$(".sale")

//to select element with id "bonus"
$("#bonus")

//to select all a tags inside of li's
$("li a")
```

Manipulating Style

The .css() method is jQuery's interface to styling.

```
$(selector)
.css(property, value)
```

Manipulating Style

.css(property, value)

We **select** and then manipulate

```
//select elem with id "special" and give it a bord
$("#special").css("border", "2px solid red");


//we can also pass in an object with styles
 var styles = {
    backgroundColor : "pink",
    fontWeight: "bold"
 };

$("#special").css(styles);
```

Manipulating Style

.css(property, value)

We can style multiple elements at once

```
//select all li's and make them yellow
$("li").css("color", "yellow");

//select all elements with class "big"
//and give them an orange border
$(".big").css("border", "1px dashed orange");
```

Exercise

Use the following starter HTML:

```
<div>Div 1</div>
<div class="highlight">Div 2</div>
<div id="third">Div 3</div>
<div class="highlight">Div 4</div>
```

- Correctly include jQuery
- Select all divs and give them a purple background
- Select the divs with *class* "highlight" and make them 200px wide
- Select the div with *id* "third" and give it a orange border
- Bonus: Select the first div only and change its font color to pink

Common jQuery Methods

Part 1

Objectives

- val()
- text()
- attr()
- html()
- addClass()
- removeClass()
- toggleClass()

jQuery Events

Making things Interactive

Objectives

- click()
- keypress()
- on()

Click()

jQuery's *click()* method is a quick and easy way to add a click listener to element(s)

```
//prints when item with id 'submit' is clicke
$('#submit').click(function(){
  console.log("Another click");
});

//alerts when ANY button is clicked
$('button').click(function(){
  alert("Someone clicked a button");
});
```

keypress()

jQuery's *keypress()* method is a quick and easy way to add a keypress listener to element(s)

```
//listen for any keypress in any text inpu
$('input[type="text"]').keypress(function()
  alert("text input keypress!");
});
```

on()

jQuery's *on()* works similarly to *addEventListener* It lets you specify the type of event to listen for.

```
//prints when item with id 'submit' is clicke
$('#submit').on('click', function(){
  console.log("Another click");
});

//alerts when ANY button is clicked
$('button').on('click', function(){
  console.log("button clicked!");
});
```

on()

It's not just for click events. *on()* supports all types of events

```
//double click event
$('button').on('dblclick', function(){
  alert("DOUBLE CLICKED!");
});

//drag start event
$('a').on('dragstart', function(){
  console.log("DRAG STARTED!");
});

//keypress event
$('input[type="text"]').on('keypress', function(){
  alert("key press in an input!")
});
```

Why Use On()?

In most cases, *click()* and *on('click')* will both get the job done.
HOWEVER, there is one key difference:

- *click()* only adds listeners for existing elements
- *on()* will add listeners for all potential future elements
- This will all make sense in the next video!