

jQuery Review

Slides heavily influenced by materials found at
teaching-materials.org

Review: DOM and Data!

Our favorite Data Types

```
var myString = "This is a string!";  
var myInteger = 1;  
var myBoolean = true;  
var myArray = ["string", 1, myVar, true];  
var myObject = {  
    name: "Pamela",  
    adjective: "Cool",  
    roles: ["hacker", "teacher", "coder"]  
}
```

Traversing the DOM with JavaScript

```
document.getElementById('presentation');  
document.getElementsByClassName('slide');  
document.getElementsByTagName('body');  
document.querySelectorAll('a');  
document.querySelector('img');
```

What's a library?

A collection of reusable methods for a particular purpose.

A math library might have functions like:

```
math.sum(array);  
math.pow(num, num);  
math.factorial(num);
```

What's a library?

- Include a script tag to the library on your page
- Call functions from that library

```
<body>

<script src="http://imagine-it.org/math.js"></script>
<script>
    var answer = math.sum(2, 2);
    alert(answer);
</script>
</body>
```

What is jQuery?

- Javascript is a Library that aims to:
 - Reduce amount of code you need to write
 - Ease DOM manipulation and events
 - Eliminate cross browser incompatibilities
 - Make AJAX simpler
- Used by over 50% of ALL websites!
 - Includes Google, Microsoft, Yahoo, Netflix, etc...

jQuery: the most popular library

Write less code for common tasks.

Abstract on top of cross-browser differences.

- Data Manipulation
- DOM Manipulation
- Events
- AJAX
- Effects & Animation
- HTML Templating
- Widgets/Theming
- Graphics/Charts
- App Architecture



Open source, big community

Getting jQuery

jQuery comes in two forms

- Development - Fully documented version
<http://code.jquery.com/jquery-2.0.2.js>
- Production - Minified
<http://code.jquery.com/jquery-2.0.2.min.js>

Then include using a `<script>` tag on the page.

```
<html>  
<body>
```

```
<h1>Hi!</h1>
<script src="jquery.min.js"></script>
<script>
  // Your code here
</script>
</body>
</html>
```

Getting jQuery

Better yet... use a CDN (Content Delivery Network)

Insert the following into your page

```
<html>
<body>
  <h1>Hi!</h1>
  <script src="//ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
  </script>
  <script>
    // Your code here
  </script>
</body>
```


`</html>`

- Advantages of CDN?
- Why `//ajax.googleapis...` and not `http://` or `https://` ?

jQuery: Why?

No library:

```
var elems = document.getElementsByTagName("img");
for (var i = 0; i < elems.length; i++) {
  elems[i].style.display = "none";
}
```

jQuery:

```
$('.img').hide();
```

jQuery: Why?

No library:

```
var p = document.createElement('p');
p.appendChild(document.createTextNode('Welcome!'));
p.style.cssFloat = 'left';
p.style.backgroundColor = 'red';
p.className = 'special';
document.querySelector('div.header').appendChild(p);
```

jQuery:

```
var newP = $('<p>Welcome!</p>');
newP.css({'float': 'left', 'background-color': 'red'});
```

```
newP.addClass('special');  
$('div.header').append(newP);
```

jQuery: The Basics

```
<p>Welcome to jQuery!</p>
```



```
$('p').addClass('special');
```



```
<p class="special">Welcome to jQuery!</p>
```

jQuery Recipe: Select, Manipulate, Admire

Step 1: Select element

```
$('#p')
```

Step 2: Use a jQuery method to manipulate

```
$('#p').addClass('special');
```

Step 3: Admire your results!

jQuery: The Basics

```
$('#p').addClass('special');
```

\$

('p')

addClass('special')

The global
jQuery function.
Can also be

Finds DOM element(s) according
to what's in the quotes.

Built-in jQuery method that adds the
specified class to the collection.

"jQuery".

Returns a "jQuery collection."

[Read the docs here.](#)

jQuery: Finding Elements

All CSS selectors are valid, plus more. [Read the docs.](#)

With this HTML..

```
<p>Welcome!</p>
```

We find it this way:

```
$('p')
```

```
<div id="main">Welcome!</div>
```

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

```
<p class="intro">Welcome!</p>
```

```
$('.intro')
```

```
<div id="main">  
<p class="intro">Welcome!</p>  
</div>
```

```
$('#main .intro')
```

jQuery: Reading Elements

If we start with this HTML...

```
<a id="yahoo" href="http://www.yahoo.com" style="font-size:20px">Yahoo!</a>
```

We can find it...

```
$('a#yahoo');
```

We can store it...

```
var myLink = $('a#yahoo');
```

...And we can find out lots of things about it:

```
myLink.html();
```



```
'Yahoo!'
```

```
myLink.attr('href');
```



```
'http://www.yahoo.com'
```

```
myLink.css('font-size');
```



```
'20px'
```

jQuery: Changing Elements

If we start with this HTML:

```
<a href="http://www.google.com">Google</a>
```


We can use this jQuery:

```
$('#a').html('Yahoo!');  
$('#a').attr('href', 'http://www.yahoo.com');  
$('#a').css({'color': 'purple'});
```

And we'll get this:

```
<a href="http://www.yahoo.com" style="color:purple">Yahoo</a>
```

jQuery Recipe: Create, Manipulate & Inject

Step 1: Create element and store a reference

```
var p = $('<p>');
```

Step 2: Use a method to manipulate (optional)

```
p.addClass('special');
```

Step 3: Inject into your HTML

```
$('body').append(p);
```

jQuery: Create and Store

Pass in any HTML string and jQuery will create it and return it as a

collection.

With this jQuery...

```
$('#<p>');
```

→

```
<p></p>
```

```
$('#<p>Welcome!</p>');
```

→

```
<p>Welcome!</p>
```

```
$('#<p class="intro">Welcome!</p>');
```

→

```
<p class="intro">Welcome!</p>
```

Just like with the DOM API, we can store a reference to our new element in memory...

```
var myParagraph = $('#<p class="intro">Welcome!</p>');
```

jQuery: Manipulate

Now that we've stored a reference, we can make further revisions to our element.

```
var myParagraph = $('<p class="intro">Welcome!</p>');
```

```
myParagraph.css('font-size', '4em');
```



jQuery: Inject

Now, we can take our stored reference to myParagraph and inject it somewhere!

```
$( 'body' ).append(myParagraph);
```

```
$( 'body' ).prepend(myParagraph);
```



regular DOM nodes to jQuery objects

```
var paragraphs = $('p'); // an array
```

```
var myParagraph = paragraphs[0]; // a regular DOM node
```

```
var $myParagraph = $(paragraphs[0]); // a jQuery Object
```

We can also loop through our array...

```
for(var i = 0; i < paragraphs.length; i++) {  
    var element = paragraphs[i];  
    var paragraph = $(element);  
    paragraph.html(paragraph.html() + ' wowee!!!!');  
};
```

jQuery Events

jQuery makes events easier

- click
- keypress
- focus
- change

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

jQuery Events

One of the most common events is the ready method

```
$(document).ready(function () {  
    // your script goes here  
});
```

or

```
$(function () {  
    // your script goes here  
});
```

Waits until the DOM is ready before the function is called.

Why do this?

<http://api.jquery.com/category/events/>

jQuery Effects

jQuery makes effects easier

- `hide()`, `show()`, `toggle()`
- `slideDown()`, `slideUp()`
- ...

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

jQuery Chaining

Elements returned by `$(selector)` can be chained

```
$(selector).method1().method2().method3();
```

```
$("#p1").css("color", "red").slideUp(2000).slideDown(2000);
```

Execute the callback function for each item

```
$(selector).each(callback);
```

this will refer to one DOM element at a time that matches the selector inside the callback



Practice Time

