Interactive Web Development

Getting Started

Dr Russ Ross

Dixie State University—Computer and Information Technologies

Spring 2015
Reading: *Learning jQuery 1.3*, Chapter 1
What jQuery does

jQuery is a platform library for client-side JavaScript that helps you do the following:

- Access elements in a document
- Modify the appearance of a web page
- Alter the content of a document
- Respond to a user’s interaction
- Animate changes being made to a document
- Retrieve information from a server without refreshing the page
- Simplify common JavaScript tasks
Why jQuery works well

jQuery employs a few strategies to make a wide range of tasks easier:

▶ Leverage knowledge of CSS
▶ Support extensions
▶ Abstract away browser quirks
▶ Always work with sets
▶ Allow multiple actions in one line

By consistent application of these ideas, the library is relatively slim, while still being powerful and predictable.
An example

Consider the following HTML:

```html
<html>
  <head>
    <title>Through the looking glass</title>
    <link rel="stylesheet" href="alice.css"/>
    <script src="jquery.js" type="text/javascript"></script>
    <script src="alice.js" type="text/javascript"></script>
  </head>
  <body>
    <h1>Through the Looking-Glass</h1>
    ...
    <div class="poem">
      <div class="poem-stanza">
        <div>sevot yhtils eht dna , gillirb sawT’</div>
        ...
      </div>
    </div>
    ...
  </body>
</html>
```
The CSS

Note that the link to the jQuery library comes **before** the link to our code.

We have a stylesheet that contains the following:

```css
.highlight {
    font-style: italic;
    border: 1px solid #888;
    padding: 0.5em;
    margin: 0.5em 0;
    background-color: #ffc;
}
```
Adding jQuery

Our custom code goes in the file alice.js:

```javascript
$(document).ready(function () {
    $('.poem-stanza').addClass('highlight);
});
```

The most fundamental operation in jQuery is selecting part of the document. The `$(())` construct achieves this, typically taking a string as its parameter. Here we use a CSS selector to identify all elements with the class `poem-stanza`. Note that this may find more than one element with the given class.

The `$(())` function is a factory for the **jQuery object**. What we get back is an object that represents all selected elements on the page.
Injecting the new class

```javascript
$(document).ready(function () {
  $('.poem-stanza').addClass('highlight');
});
```

The name of the `.addClass()` method is self-descriptive. It adds a CSS class to any element or elements that we have selected. Through implicit iteration, the action is repeated for all selected elements. `.removeClass()` would let remove a class from selected elements.

The class is only added to elements that do not already have it, or removed only from elements that do have it.
Executing the code

The $() statement and .addClass() call are enough to perform the desired action. However, if we simply executed that code in a script loaded in the document header, nothing would happen. The code would be executed before the page was loaded and rendered. We need to delay its execution until the page is ready.

One way to fix this is to package the code into a function so we can run it later:

```javascript
function highlightPoemStanzas() {
    $('.poem-stanza').addClass('highlight');
}
```

Then you could invoke that function when the page has loaded:

```html
<body onload="highlightPoemStanzas();">
```
Executing the code

Modifying the HTML directly to invoke JavaScript is generally a bad idea, cluttering the server-side code. This also waits for images to finish loading, which adds unnecessary delays.

jQuery lets us schedule a function to run when the DOM is ready but images are not necessarily loaded:

```javascript
$(document).ready(highlightPoemStanzas);
```

This does not require modifying the HTML at all. Creating a function that is used only once is also a bit tedious. We use a function literal instead:

```javascript
$(document).ready(function () {
    $('.poem-stanza').addClass('highlight');
});
```
Examples

Complete code for all examples from the book can be downloaded from the book’s website. I recommend downloading it and following along during classes:


I will often abbreviate examples to make them fit better on the slides.