LC 101

Unit 3 - jQuery

March 13, 2017

What is jQuery?

“jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.”

https://jquery.com/

What is jQuery?

- jQuery is a JavaScript library
  - Written in jQuery
  - Designed to make writing programs in JavaScript easier
    - Simplifies DOM manipulation
    - Hides browser differences
- The **`jQuery`** object is actually a function
  - It encapsulates many other functions and objects
- The dollar sign **`$`** is another reference to the **`jQuery`** object
  - Shorter than writing **`jQuery`** everywhere

Example

- DOM Manipulation in jQuery is much simpler than plain JavaScript

```
// Plain JavaScript
var div = document.createNode('div');
div.innerHTML = "Hello, World!";
var parent = document.querySelector('#parent');
parent.appendChild(div);


// jQuery
$('#parent').append('<div>Hello, World!</div>');
```

Including jQuery

- Use a **`script`** tag to include jQuery in an HTML page, just like any other js file
  - Can host it on your own server, use the jQuery home server, or a content delivery network
    - (You only need one, not all three)
  - Should use the *minified* version in production
    - Minifying shrinks the size of the js file by removing formatting and renaming internal variables
  - Should come before any of your scripts that use jQuery

```
<script src='js/jquery.min.js'><script>
<script src='//code.jquery.com/jquery-3.1.1.min.js'></script>
<script
src='//ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js'></script>
```

jQuery Selectors

- You can select elements in the DOM using syntax similar to CSS selectors

```
// $('tag') returns a jQuery collection of all elements of a tag
var divs = $('div');

// $('.class') returns a jQuery collection of all elements with a class
// Note the . before the class
var articleItems = $('.article-item');

// $('#id') returns a jQuery collection of the single element with the id
// Note the # before the id
var navItem = $('#nav');
```

Traversal Methods

```
parent = $('#abc').parent();                // immediate parent
ancestors = $('#abc').parents();            // all ancestors
ancestor = $('#abc').parents('.green');     // filtered
children = ('#abc').children();             // immediate children
descendants = $('#abc').find('*');          // all descendants (see note below)
descendants = $('#abc').find('div');        // filtered
siblings = $('#abc').siblings();            // all siblings
siblings = $('#abc').siblings('#def');      // filtered
```

- The parameter to **`find`** is not optional, so if we want all descendants then we need to pass in the universal selector **`'*'`**

DOM Manipulation

- Many methods exist to modify or access attributes
    - This is just a subset. Check the documentation

```
addClass
removeClass
toggleClass
hasClass
css
attr
prop
html
text
val
```

DOM Manipulation

- Many methods will add or remove elements
    - Many of these will take either an element or an HTML string as a parameter

```
before
after
insertBefore
insertAfter
detach
empty
remove
```

`.each` and `$(this)`

- The `.each` method will apply a function to each element in a jQuery collection
- Use `$(this)` in the function to access the jQuery wrapper around the element
    - Just `this` will access the bare DOM element
- Return `false` from your function to exit the each loop early

```
$('.someClass').each(function(index) {
    console.log(index + ': ' + $(this).text());
});
```

On Ready

- We can specify a function to execute on page load after the DOM has been created by passing the function to the jQuery object
    - The old syntax was to use the `ready` method, but that is now deprecated

```
// Recommended syntax
$(function() {
    // The code I need to run on page load
});

// Old style
$().ready(function() {
    // The code I need to run on page load
});
```

Event Listeners

- The **`on`** method in jQuery is the equivalent of **`addEventListener`**
  - **`$(this)`** is the target element in the handler
  - Returning false is the same as calling both **`event.stopPropagation()`** and **`event.preventDefault()`**

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

Event Delegation

- We can use the **`on`** method to attach a single listener to multiple descendants of an element
  - This will even work for descendant elements that are created after we attach the listener

```
$('#someElement').on('click', 'button', function(event) {
    // Will be called when any descendant button of #someElement is clicked
});
```