305: jQuery

Dan Goldsmith

May 2018

Introduction

JQuery

- A Library for writing JavaScript applications
- Designed to simplify common and complex tasks
- Has components for UI and data

JQuery

- Rather than write all code from scratch
  - JQuery provides a *Library* or *functions* where this code is already written.
- Also takes care of many *cross-browser* problems.

JQuery: Features

- HTML/DOM manipulation
- CSS manipulation
- HTML event methods
- Effects and animations
- AJAX
- Utilities

JQuery: Who uses it.

- Used by some big names
  - Twitter
  - Microsoft
  - Google

Getting Started

Including JQuery in your site

- Two ways
  - Download the source and host it locally
  - Use an online Library (CDN)

Including JQuery via CDN

- There are several to choose from
- May use already cached version. (speed)
- Will use a server local to the user (speed)
  - Google is good

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">
</script>
</head>
```

Our first JQuery document

```
<html>
  <body>
    <script>
    $(document).ready(function(){
        alert("Hello World!")
        })
    </script>
  </body>
</html>
```

Breaking it down

- Use JQuery to select the current document
  - Remember *selectElementById* yesterday
- Setup a callback to another function when the document is loaded.

Breaking that down

- **$** Signifies JQuery Code
- **(document)** JQuery Selector
- **.ready(. . . .)** JQuery action *function* to run

The Selector function

- Tells jQuery which DOM element we attach the code to (more on this later).
  - $(document)
  - $(“p”) //All Paragraphs
  - $(“.className”) //All Items with class “className”
  - $(“#id”) //All Items with specified Id

Callbacks?

- Another way we can use functions
  - Use them as a parameter in another function
- This can be useful with Asynchronous events
- Network stuff, processing etc.

Callbacks in English

- Imagine we have a chunk Work that takes some time to run
- Other tasks depend on this being completed
  - We create a function that tells us what to do **AFTER** the network stuff happens
  - Pass that callback to the long running function
  - The rest of the code continues to execute

So back to our ready() function

- What is happening is our main code is loaded as a callback
- It takes some time for the page to load
- When this has occurred the main code is called.

JQuery Selectors

Selectors

- So jQuery needs to know *what* element we are going to attach to
- Will talk about common ones here
- Full Reference
  - https://www.w3schools.com/Jquery/jquery_ref_selectors.asp

Selectors

- Give us a much more powerful way of doing:
  - document.getElementById
- Cover a wide range of options

Basics Selector syntax

- **$** jQuery
- **()** Selector

```
$()
```

Selectors: Elements

- Elements our the basic HTML building blocks
  - Tags like <p> etc

```
$("p") //Selects all <p> tags
$("li") //Selects all <li> tags
```

Selectors: #Id

- I find this the most useful: Only selects items with a given Id
  - All Id's in the document should be unique

```
<div id="theDiv"></div>
```

```
$("#theDiv") //Select items with the id theDiv
```

Selectors: .Class

- Selects all elements of a given class

```
<div class="testing"></div>
```

```
$(".testing") //Select all items with class testing
$("div.testing") //Select all DIV's with class testing
```

Your Turn

Your Turn

Grab the SelectorPlayground code from GitHub

- When the button is clicked A callback is triggered
  - Read and understand what the code is doing
  - Change the existing elements that are selected
  - Add new elements and try to select them

jQuery Events

Events

- As well as selection jQuery also has event triggers
  - We have been using these before
  - $(“button”).click()

Events

- Are given a callback as a parameter
- When the event occurs the callback is triggered
  - NOTE: Callback can be specified in a separate function

Common Events: click

- Called when a button / item is clicked

```
$("button").click(function(){
    alert("Button Pressed")
})
```

Other Common Events

- See https://www.w3schools.com/Jquery/jquery_events.asp
- Lab Task: Complete the Events excesses at the bottom of the W3 Schools page

Getting Data from other areas: AJAX

AJAX

- Asynchronous JavaScript and XML
  - Browser requests data using a HTTP Request
  - Data is processed and rendered via the DOM

Why is this useful

- Allows a website to display data without reloading the whole page.
  - Functionality like Chat, “Live” data etc.
  - Takes the web from static to dynamic

Our first Ajax Request

- We need two things for this,
  - The page making the request
  - The data to load

First Ajax Request

```
<!DOCTYPE html>
<html>
<head>
  <!-- Load Jquery -->
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js
</head>

<body>
  <h1>First Ajax Program</h1>

  <div id="div1"><p>Data to be loaded<p></div>

  <button>Get External Content</button>

  <script>
    // Load code when all the Dom has happened
    $(document).ready(function(){
      //Add a callback for the button
      $("button").click(function(){
            $("#div1").load("ajax1_data.txt");
    });
});
```

HTML Part

```
<div id="div1"><p>Data to be loaded<p></div>
<button>Get External Content</button>
```

JS Part

```
    // Load code when all the Dom has happened
    $(document).ready(function(){
      //Add a callback for the button
      $("button").click(function(){
            $("#div1").load("ajax1_data.txt");
    });
});
```

JS Part Description

- jQuery select the document
  - Attach a Ready action to it (so code is available after the DOM loads
- Attach a *click* action to the button
  - Selects the element with id *div1* and loads the text in the file specified.

A Quick improvement

- This example has a minor flaw
  - What would happen if there were two buttons?
  - It would be better if the button was accessed by an ID

The Load() Method

```
$(selector).load(URL,data,callback);
```

- Can be used to load an external resource (such as a file)
  - **URL** to get the data from
  - **Data** Optional: Arguments to the resource
  - **Callback** Optional: What to do with the data when it returns

The Load Method: Callback

```
function(responseTxt, statusTxt, xhr)
```

- **responseTxt:** Data returned by the request
- **statusTxt** HTTP status (success, failiure)
- **xhr** XML HTTP Request object

Lets improve the Code:

- We didn't have the optional callback function in our original code
- Lets add one so we can deal with errors, (and it is good practice for Async later)

Code Improvement

```
$(document).ready(function(){
    //Callback for Button
    $("#loadButton").click(function(){
      $("#div1").load("ajax_data.txt", function(responseTxt, statusTxt, xhr){
        //If the data had an error loading correctly
        if (statusTxt == "error"){
          alert("Error Loading Text")
        }
      }) //Load Callback
    }) //Button
}) //Document
```

Other Request Methods:

- Will Differ based on the API.
- POST Requests
- JSON Requests
- Information at
  https://www.w3schools.com/Jquery/jquery_ref_ajax.asp

Getting Data from a Public Service

Getting Data from a Public Service

- Loading stuff from a file is pretty cool
  - But its hardly the dynamic web we were after
- We can use AJAX to get data from other services

Where to get information from

- HK has a nice number of datasets
  - https://data.gov.hk/en/
- We will use the address lookup set
  - https://data.gov.hk/en-data/dataset/hk-ogcio-st_div_02-als

Examining the API

- Its a really simple API here
- Take a look at what happens in a browser

`https://www.als.ogcio.gov.hk/lookup?q=<input address in fre`

Making the request in Ajax

- We need to use a different type of request here
  - GET request (we could also use post)

Ajax GET Requests

```
$.get(URL,callback);
```

And the callback

```
function(data, status
```

GET Request for this:

```
var URL = "https://www.als.ogcio.gov.hk/lookup?q=VTC"
$("button").click(function(){
    $.get(URL, function(data, status, xhr){
        console.log(data)
    });
});
```

Huston we have a problem!

- Data has been returned
  - But its XML
  - That's fine but its so 2000, we have to parse it manually and everything
- Solution: Request the data in JSON format

Making a JSON request

- If we look at the documentation for the API we can see that changing the HTTP headers will change the request
  - This is a little unusual, normally they are parameters in the body etc
- documentType : xml / json
- language

Modifying the Headers

- Taking a look though the jQuery AJAX documentation we see that there is a function to change settings for Ajax requests

```
$.ajaxSetup({dataType: "json"})
```

Parsing Json

- As we are dealing with JSON data, we may as well tell jQuery to expect it
- Fortunately jQuery has a getJSON function
- And almost its the same as our GET request
  - Optional *data* parameter which we can use to send arguments to the server

getJSON

- Swap the function over and lets take a look at the result

```
$(selector).getJSON(url,data,success(data,status,xhr))
```

JSON

JSON

- JavaScript Object Notation
- Alternative for XML
  - IMHO Much nicer way of doing things

JSON: What is it

- Remember the base OBJECT type
  - Collection of *Key*:*Value* Pairs
- Represents Data returned by the API in this format

Deciphering the JSON from our request

- Two Methods
  1. Read the Docs (gets you started)
  2. Play with the JSON in the Console Window (useful for debugging)

Deciphering our Requests JSON

- From the Docs https://www.als.ogcio.gov.hk/docs/Data_Dictionary_for_ALS_EN.pdf
  - Or http://www.xml.gov.hk/en/approved/structured_address_v1_0.htm
- Returns a object containing two more Objects
  - {AddressRequest, [SuggestedAddresses]}
- Its the Suggested addresses we are interested in

Suggested Address Format

- Lots of Nested Data here
  - Its a bit of a nightmare but we can soon work it out

Suggested Address Format

```
{Address : {
   PremisisAddress : {
      EngPremisisAddress : {
         BuildingName : DATA
         EngStreet : {
            BuildingNoFrom : DATA
            StreetName : DATA
            }
      }
```

Lets print the Building Name

```
//Store the Address Object
var address = item["Address"]["PremisesAddress"]["EngPremis
var buildingName = address["BuildingName"]
console.log(buildingName)
```

What about the Street Address

```
var streetAddressNumber = address["EngStreet"]["BuildingNo
var streetAddress = address["EngStreet"]["StreetName"]
console.log(streetAddressNumber)
console.log(streetAddress)
```

Your Turn

- There are a couple more thing we need to do to fix the code
  - Example3a
- Get the GeoLocation Coordinates for each address
  - NOTE: Several are returned, just take the first item
- Change the Code so we loop through all returned results

Modifying the HTML to deal with the data

Getting out of the Console

- We have data being shown on the console.
  - No use to anyone using the App
- Lets get it to display on the web page

jQuery DOM Manipulation

- Our Plan will be:
  - Create a table to show the results
  - Select the Table
  - Append the results to it

Simple Table Code

```
<table border=1 id="theTable">
  <thead>
    <tr><th>Name</th><th>Address</th><th>Lat</th><th>Long</th></tr>
  </thead>
  <tbody></tbody>
</table>
```

Table Parsing Code

```
var newLine = "<tr><td>"+buildingName+"</td><td>"+streetAddressNumber + " " + s
var theTable = $("#theTable")
.find("tbody")
.append(newLine)
```

Your Turn

- There is some code to get you started in Example3b
- You will need to merge the Lat and Long codes from before

Making it more interesting

Adding a Map

- Final stage is to add a Map to the Page
- We can follow the tutorial at
  - https://leafletjs.com/examples/quick-start/
- You will also need a mapbox API key from
  - https://www.mapbox.com/studio/account/tokens/

Notes

- Decent starting Coorindates

```
var mymap = L.map('theMap').setView([22.2588, 114.1911], 12);
```

Other Resources

Other Resources

- Selectors
  - https://www.w3schools.com/Jquery/jquery_ref_selectors.asp
- Tutorial
- https://www.w3schools.com/Jquery