JSON + AJAX

AJAX Requests

AJAX stands for
A synchronous **J** avaScript **A** nd **X** ML

The meaning: web applications can send or receive data from the server without requesting an entire page - instead, they can just request or send an arbitrary amount of data

Making AJAX Requests with jQuery

The easiest and most cross-browser compatible way to make an AJAX request is with jQuery.

```
$.ajax({
    type: "POST",
    url: "/request",
    data: {q: "131 Humboldt St"}
})
```

Making AJAX Requests with jQuery

If the "success" property contains a function, that function will be called once the AJAX request completes successfully:

```
$.ajax({
    type: "POST",
    url: "/request",
    data: {q: "131 Humboldt St"},
    success: function (d) {
        alert('data returned is: ' + d);
    }
});
```

Making AJAX Requests with jQuery

Try making this request from the console on the Spotify api.

```
$.ajax({
  type: "GET",
  url: 'https://api.spotify.com/v1/search',
  data: {
      q: 'prince',
      type: 'artist'
  },
  success: function (response) {
      console.log(response);
  }
});
```

JSON - JavaScript Object Notation

J ava **S** cript **O** bject **N** otation

“a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate.”
-json.org

```
{
  "string": "Hello!",
  "number": 1.2345,
  "array": [1, 2, 3],
  "object": { 1: true, 2: false }
}
```

JSON - JavaScript Object Notation

- JSON is *far* stricter than javascript in terms of syntax.
  - Keys *must* be quoted
  - There can be *no* dangling commas
  - There is no concept of variables
- JSON can only return one thing, but it doesn't have to be objects. It could also be a different value, such as an array or an integer

JSON - JavaScript Object Notation

JSON is parsed into a JavaScript object by using the `JSON.parse()` method and passing it the JSON as a string.

```
var parsed = JSON.parse('{"jelly": "fish"}')
parsed.jelly
>>"fish"
```

Most JSON is returned as the result of an **AJAX call**.

JSON - JavaScript Object Notation

JSON can also be created by using `JSON.stringify()` and passing it a value

```
var obj = { tiger: "lily" };
JSON.stringify(obj);
>> "{tiger: \"lily\"}"
```

Cross-domain considerations

- You cannot normally make an AJAX request across different domains
- CORS is the policy that defines this, an acronym that stands for "Cross-Origin Resource Sharing"
- Some domains don't allow you to ping them from other domains and some do

art-share API

- We've devised a simple API to allow you to try out JSON requests and responses
- The art-share API is available at art-share.herokuapp.com
- All documentation is also contained on this page

Cross-Domain/AJAX Exercises

- Practice sending some AJAX requests back and forth to the art-share API based
- If necessary parse the JSON objects returned and accessing their properties
- Finally, place some of the data returned onto the page using the following:

```
$("body").html(data.name)
```

Cross-domain/AJAX Exercises

To further understand the fact that you cannot make an AJAX request to another domain name, just attempt to make one inside of this sample app to nycda.com for the homepage (/), log the output to the console, and see what happens

Example of Cross Domain Error Response.

```
$.ajax({
    type:"GET",
    url:"https://nycda.com/",
    success: function (d) {
        alert('data returned is: ' + d);
    }
});
```

Grab your popcorn...

```
$.getJSON("http://www.omdbapi.com/?",
    {
      t: "sharknado"
    },
    function(response) {
      console.log(response);
    }
  );
```

$.getJSON is a jQuery utility function - a shorter version of $.ajax

omdb API

```
$.ajax({
  url: "http://www.omdbapi.com/?",
  data: {
    t: "sharknado"
  },
  dataType: "json",
  success: function(response) {
    console.log(response);
  }
});
```

Rendering the response

Create placeholder elements for the parts of the response you want to show

```
<h2 id="movie-title">Movie Title</h2>
<img id="poster"/>
```

Rendering the response

In your success function, populate those elements using jQuery.

```
function(response) {
  $("#movie-title").html(response.Title);
  $("#poster").attr("src", response.Poster);
}
```

Exercise:

Use the OMDB api to create a quick movie search app.

Using jquery's methods `.onkeyup()` and `.val()` you can create a workable search bar that will execute an api call as someone is typing a movie title into the bar!

1. Have the title and poster update as search result comes in.
2. Use jquery to also display 2-3 more details about the movie (i.e. genre, year released, actor list, etc.)

Bonus:

- make it so that the search only executes after a user has typed in at least 3 characters