APIS

WHAT IS AN API?

API stands for Application Program Interface.

It’s the mechanism that dispenses structured data to webpages but it’s hard to make sense of.

WHAT IS AN API?

It’s really just a website, it has a url (also called an “endpoint”), and sends a response when you make a request to it.

However instead of HTML, an api send raw data.

An api is just a website that sends structured data instead of a rendered interface.

WHAT DOES AN API DO FOR US?

With API's we can make website's that access servers. This can either be an API that we own or an API owned and maintained by third parties.

THIRD PARTY API'S

Third party API's can bring a lot of depth to a site without adding any development side on our part. Imagine what it would take for us to build our own website with up to date weather.

Without API's we could either:

API-LESS WEATHER API? LOL

Without API's we could either:

Option 1: build a weather radar and sensor that also automatically generates HTML for our site and constantly updates it (impossible).

Option 2:

- Step 1: look up the weather, update the html, push the new code.
- Step 2: 5 minutes later, look outside again, update the weather, push the new code.
- Step 3: 5 minutes later, look outside again, update the weather, push the new code.
- Step 4: 5 minutes later, look outside again, update the weather, push the new code.
- Step 5-∞: repeat until ☠️

WEATHER WITH AN API

With API's we can do this much easier. There are a number of weather companies that are already checking the weather and making that data publicly through an API.

All we have to do is make a small script that requests that data and our site will automatically update with the weather.

WHAT DO API'S DO FOR US?

And this applies to a number of data services; everything from live information about weather, to expansive databases about music, events, movies, public health, whatever.

API's let us add data-driven depth to our websites without having to maintain data on our own.

ACCESSING THIRD PARTY API'S

It does cost money for companies to maintain and serve their data. The have a number of ways of making sure they don't lose tons of money on their services.

There are some free ones (often times limited or just-for-fun datasets).

There are some that require payment for requests.

There are others that will require registration and then give a limited amount of free requests.

API SECURITY

In the case of paid or limited API's, somehow API providers need to keep track of the developers that are making the request. This is to ensure that no developer is gaming the system, making too many requests and costing the API providers too much money.

They do this with API "keys".

Think of the Internet as a series of buildings…some big, others small.

Each building has a door


And doors
require
keys…

KEY IDEA

API’s are doorways into different website’s data. For most sites, you’ll need a key to get inside (often just called API keys).

HOW TO ACCESS AN API

1) Super Simple: you just need a url and make an API call
2) Simple: you'll have to get an API key to perform a transaction with an API.
3) Moderate: If the API gets/puts data into an account, you will likely need account information as well.
4) Tough: You may have to enter a username/password using OAuth protocol.

Once you unlock the door, you don’t know what will be inside (or how many rooms there are).

WHICH ROOM?

Continuing this metaphor, API “rooms” are called endpoints.

Each API has different endpoints that return different data once you transact the key/user/OAuth sequence.

READ THE DOCS!

Every API is unique - you will have to explore its documentation to figure out which endpoint you want to use.

THE DOCS

And that does require some skillful reading sometimes.
Documentation comes in all shapes, sizes and quality.
Interpreting the docs can require some skills sometimes.

APIS

This is a huge public catalogue of APIs (but not exhaustive, there are thousands more):

https://github.com/toddmotto/public-apis

API RESPONSES

Lets take a look at what an API gives us back

SWAPI

JSON

API endpoints will almost always return JSON, which looks like javascript and stands for “Javascript Object Notation”

```
{
  “data”: “car”,
  “make”: “ford”,
  “model”: “focus”
  “details”: {
    “color” : “blue”,
    “mileage” : “54019”
  }
}
```

JSON

This looks like it is a javascript object, and fair enough to assume so, but that is technically not true. Technically it is just a non-javascript string that is conveniently in javascript format.

Luckily for us, AJAX will convert that string into a real javascript object automatically.

JSON

So for all intents and purposes, we can treat it like a true javascript object thanks to AJAX implicit formatting.

Bear in mind that with other languages this conversion would not be automatic. JSON is still a popular data type even if it is never converted to a “true” javascript object.

API

JSON

ajax

Asynchronous

http method

Endpoint

Success

Error

AJAX

Asynchronous Javascript and XML

MAKING AN API CALL

We’re going to keep using jQuery and one of it’s most useful parts: .ajax();

AJAX

Today, most endpoints return JSON (JavaScript Object Notation), but we still refer to these types of operations as AJAX

ASYNCHRONOUS

Asynchronous means not existing or occurring at the same time.

ASYNC JS

One of the characteristics of javascript requires an interesting work around.

Javascript is a single-threaded language, which means that it can only process one command at a time.

ASYNC JS

And when we make a request to a server, the server can take time. We are at the will of the server response time! So if the server were to take 2 minutes, our javascript would just… stop for 2 minutes unlessss

ASYNC JS

There was a way to put that response on hold, process some other stuff while it loads, and then pick up the handling for the API when there is a response.


TURNSTONE
بيتزا
Artisan
أرتيزان
Pizzas
تيرن ستون
Think of your request as ordering a pizza
First you place your order
تيرن ستون
THE
TURNSTONE

When waiting for your pizza,
you don’t have to sit by the door


You have other things to do in the meantime!


But once it comes, you know what to do!

THE EVENT LOOP

This is a doozy of a topic. Let's do this on white board!

AJAX

With ajax your request is happening outside of the normal timing of the code. Once it completes, it will run the actions we tell it to

.ajax() - basic

```
$.ajax({
   method: ‘GET’,
   url: ‘http://linktoapi.com’,
   success: function (results) {
      // do something
   }
   error: function (results) {
    // do something else
   }
});
```

WHAT DOES IT MEAN?

.AJAX() - BASIC

```
$.ajax();
```

The $ sign means we are using jQuery and the . after it means we are calling a function. In this case it is called ‘ajax’

.ajax() - basic

```
$.ajax({
   method: ‘GET’
});
```

In the function we put an object. The first key value pair is telling us what the http method is, or what type of request we are making.

.ajax() - Basic

```
$.ajax({
   method: ‘GET’,
   url: ‘http://linktoapi.com’
});
```

The second key - value pair is telling us what endpoint to make the request to, or where the request should go.

.ajax() - basic

```
$.ajax({
   method: ‘GET’,
   url: ‘http://linktoapi.com’,
   success: function(results){
      // do something
   }
});
```

Next we have a function that will run when the results are successfully returned.

.ajax() - basic

```
$.ajax({
   method: ‘GET’,
   url: ‘http://linktoapi.com’,
   success: function(results){
      // do something
   },
   error: function(error){
      // do something else
   }
});
```

The last will run if the request is unsuccessful

API

Object

ajax

Asynchronous

http method

Endpoint

Success

Error

CODEALONG

Assignment 1

YOUR TURN

Assignment 2