Introduction to APIs

Tim Book & Jeff Hale

GENERAL ASSEMBLY

What is an API?

Application **P**rogramming **I**nterface

Set of protocols for using and building application software.

Contract between an information provider and an information user.

Specifies the content required by the consumer (the client call) and the content required from the producer (the server response).

-adapted from Red Hat https://www.redhat.com/en/topics/api/what-is-a-rest-api

API example

The API for a weather service could specify that a user supply a 5 digit zip code and the producer reply with the high temperature predicted today.

Adapted from RedHat: https://www.redhat.com/en/topics/api/what-is-a-rest-api

Web APIs

Acronym Soup

- API
- REST
- CRUD
- HTML
- URL
- JSON

REST APIs

REST: REpresentational **S**tate **T**ransfer

- An architectural style for the web
- A RESTful API allows a user to make HTTP requests to manipulate data through **CRUD** operations

CRUD operations

Create

Read

Update

Delete

CRUD operations

Each CRUD operation has an associated request verb.

Use the **get** request to read data ⭐

CRUD operations

CRUD Operation	HTTP Verb
Create	post
Read	**get**
Update	put/patch
Delete	delete

Read more here.

What is a Web API?

We use a web API to *get* the **HTML** or JSON content of a webpage at a **URL** endpoint.

HTTP = **H**yper**T**ext **T**ransfer **P**rotocol (the rules)

URL = Uniform Resource Locator (the website address)

What is HTTP?

HTTP is a set of rules for moving data from the web

HyperText

As in, HyperText Markup Language (HTML)

Transfer

Movin' stuff around

Protocol

A set of rules

The Example Example

Let’s check out this website:

www.example.com

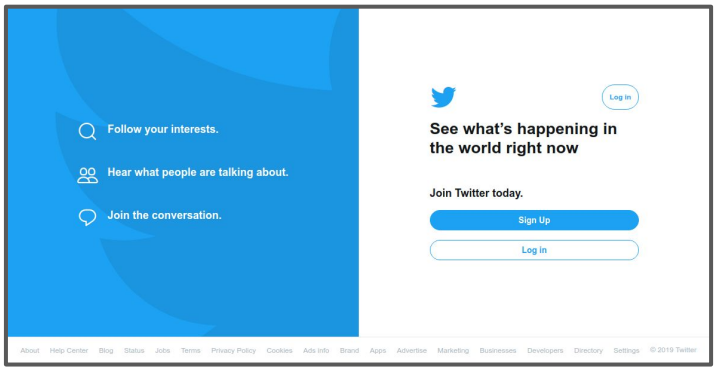
Right click and select *Inspect* if using Chrome.

What we do... literally every time we go to a website

`GET www.twitter.com`

```
<!doctype html>
<html>
<head>
   <title>Welcome to Twitter or Whatever!</title>
</head>

<body>
    ...
</body>
</html>
```



The Example Example

Let's use our command line to make our first HTTP request!

curl is a command line client for making requests.

In the terminal:

```
curl www.example.com
```

Data-focused APIs

HTML vs JSON

An HTML endpoint can return **HTML** or **JSON**

HTML - intended for viewing in a browser

JSON - data to use for data things 🙂

(Or XML if you are unlucky ☹)

Let's get JSON data

Use the PokeAPI: https://pokeapi.co/

From the cmd line run:

```
curl https://pokeapi.co/api/v2/pokemon/bulbasaur
```

Meet JSON!

Java**S**cript **O**bject **N**otation

- Standard format for data transfer on the web
- Text encoded as a raw string.
- What Python object does it resemble?

```
{
  "abilities": [
    {
      "ability": {
        "name": "chlorophyll",
        "url":
"https://pokeapi.co/api/v2/ability/34/"
      },
      "is_hidden": true,
      "slot": 3
    },
    {
      "ability": {
        "name": "overgrow",
        "url":
"https://pokeapi.co/api/v2/ability/65/"
      },
      "is_hidden": false,
      "slot": 1
    }
  ],...
```


Getting Data

API Issues

- API Keys
- Rate limits
- $

DO NOT store unencrypted API keys in on the internet (e.g. GitHub)

Read more on secrets management here:

https://blog.gitguardian.com/secrets-api-management/

Fine, fine, time for Python

Let’s go to the notebook!

Data APIs

- Use Python *requests* library
- If you get back JSON, cool! 🎉
- If you get back HTML or XML use BeautifulSoup to parse it.

Python API Wrappers

Great if a Python wrapper exists for the API you want

Install and skip using the requests library

See Jeff's list of Python API Wrappers here

How to get data (roughly from easiest to most work)

1. **From a file**
2. **From a database**
3. **From a data API**
    a. Directly from Python if there's a wrapper library
    b. Bare bones with requests - usually returns JSON
4. **From a website's HTML**
    a. pd.read_html() returns DataFrames
    b. Requests to fetch, BS4 to parse, then put into a DataFrame
    c. Use selenium for headless browsing (something that can handle JS, waiting, form filling)

Acronym Soup practice

Explain to a friend what each one stands for and does:

- API
- HTML
- REST
- CRUD
- JSON
- URL

