Welcome to Week 14, Lecture 01!

APIs & JSON

05/23/22

Agenda

- APIs
- JSON
- Quick Note Re: File Paths
- CodeAlong: Mapping Yelp API Results - Part 1

Assignments

Last Week's assignments:

- Feedback Docs have been updated!
- Make sure your repos are set to Public!
- You should make a separate repo for your Project

This week's assignments:

- Efficient Yelp API Calls (Core)
- Project 3 - Part 2 (Core) - Will have a Post-Class walkthrough of Part 2 on Wednesday.
- Applying Advanced Transformations (Core)

Deadline extended to Sunday at 11:59 PM!

APIs

What is an API?

- API = “Application programming interface”
  - A connection between two computers/computer programs.
  - Structure access/exchange to information.
- You are constantly using APIs via Apps and websites without realizing it.

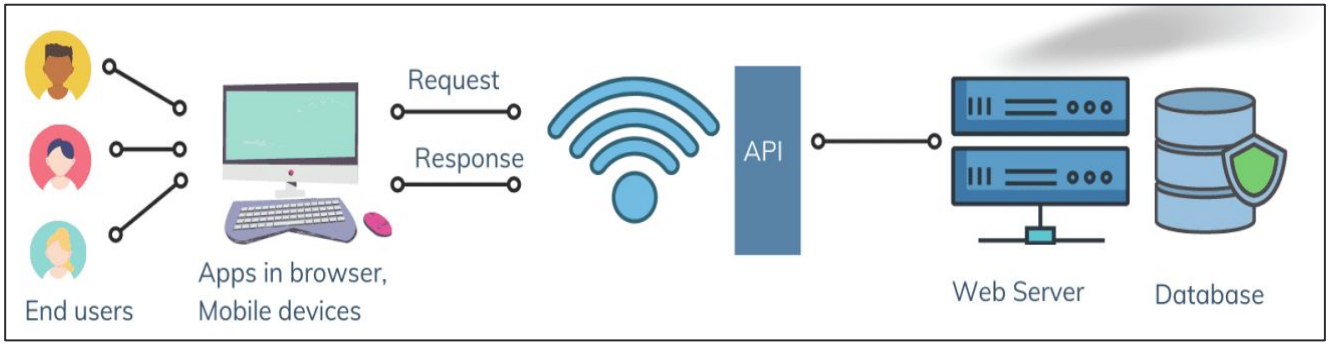

Image Source

- APIs allow data sharing in a structured way that is controlled by the owner of the data.
  - APIs can have different “endpoints” for different information.

What is “Authentication”/”Authorization”?

- **To avoid constantly using/exposing your username and password, APIs allow you to get an alternative (& safer) way to authenticate users.**
  - We create an “app” for an API (like Yelp, Spotify, etc) - and the a**pp will get special credentials.**
- **Instead of a username and password, our app will usually get a “Client-ID” and “API-Key”** (sometimes called a client-secret)
  - We provide our API key every time we perform a query (called an API “call”).
  - Using our API-Key lets Yelp (for example) know which account is requesting data.
  - It will track how many queries you have performed.
- Free vs Paid APIs:
  - Some APIs allow free, but limited access to their data.
  - Usually limited to a specific # of api calls per month (or day)
- We will make a hidden in our user folder called “.secret/” for storing our credentials.

Using APIs

- Using an API requires **getting very familiar with the API's documentation.**
- While there is a general framework for the types of API calls that can be made. There is a LOT of variability between APIs.

- To use an API, we use a web **request.**
- **Requests come in 4 flavors:**
  - GET: Retrieves resources/data.
  - POST: Creates/provides resources/data.
  - PUT: Changes or replaces resources/data.
  - DELETE: Deletes or replaces resources/data.

- **As a Data Scientist, you will mostly be using GET requests.**
  - We can manually construct these requests using Python "requests" package. (See Optional lesson for more).
  - BUT there are usually Python packages for an API that make it easier to use.

JSON

JSON: Javascript Object Notation

- Most popular data format for API's.
- Lightweight "data-interchange" file format - easy to share/access
- Easy for humans to read and write.
- Can be easily read and accessed using packages in other systems and languages.

- Stores text data in a hierarchical structure.
  - Uses Key: Value pairs separated by commas (like dictionaries).
  - Also uses lists/arrays

JSON File Requirements

- Json Objects/Files must start with a list or a dictionary.
  - Must start with a [or {
- Must use double quotes instead of a single quotes.
- The keys must be strings.
- The values can be strings, numbers, lists,boolean, or None.

Example JSON: Three Employees

```
{"employees":[
  { "firstName":"James", "lastName":"Irving" },
  { "firstName":"Sherlin ", "lastName":"Whaley" },
  { "firstName":"Purvi", "lastName":"Kansara" }
]}
```

Reading/Writing JSON with Python

- Use the JSON module that comes with Python.
  - json.load: for loading data from a json file
  - json.dump: for saving data to a json file

- To read in data: combine with the Open function using read mode ('r')

```
# Load API Credentials
with open('/Users/codingdojo/.secret/yelp_api.json','r') as f:
    login = json.load(f)
login.keys()
```

- To save data: combine with the Open function using write mode ('w').
  - Provide the variable to save to json followed by the opened file.

```
with open(JSON_FILE,'w') as f:
    json.dump(previous_results,f)
```

File Path Notation

- Key File Path string info:
  - “~/” indicates the “Root Directory” (usually your User folder)
  - “/” indicates the very top-level directory of your hard drive
    - E.g. “/” == “C:/”
  - “./” indicates the CURRENT folder.
  - “../” indicates the folder ABOVE the current directory.

- Windows File Paths:
  - Use 2 backslashes (\\) instead of a forward-slash (/)
  - Backslash is a special character so must use two backslashes “\\”
  - Must also use a backslash before spaces.

CodeAlong/Activity

Activity Repository

✅ CodeAlong: Mapping Yelp API Results - Part 1:

- https://github.com/coding-dojo-data-science/data-enrichment-wk14-activity-mapping-yelp-api-results
  - Post Class Branch/Notebook: https://github.com/coding-dojo-data-science/data-enrichment-wk14-activity-mapping-yelp-api-results/tree/05-23-22-class

❌ JSON Activity (Breakout Groups):

- https://github.com/coding-dojo-data-science/data-enrichment-wk14-l01-json-activity
- Activity has a solution branch if you'd like more practice sifting through JSON.

APPENDIX

Absolute vs Relative File Paths

- File Paths that start with a slash (or \\ on Windows) are **absolute file paths**
  - Includes every folder from the top-level of the hard drive to the specified file.
  - Example: “/Users/codingdojo/Documents/GitHub/json-apis-activity/Data”

- **Relative file paths** do not start with a slash and are relative to the location of your file/terminal.
  - Example: if you are already in the GitHub folder, it would be: “json-apis-activity/Data”

- For your Jupyter Notebooks in a repository, ALWAYS USE RELATIVE FILE PATHS FOR YOUR DATA!!
- Example absolute file path that will NOT work for others.
  df = pd.read_csv(“/Users/codingdojo/Documents/GitHub/json-apis-activity/Data/example.csv”)
- Instead, use this relative file path:
  df = pd.read_csv(“Data/example.csv”)

Example Absolute vs Relative File Path

- Let's say you've cloned a repo called "json-apis-activity" into the GitHub folder inside your Documents folder (the default folder used by GitHub Desktop).
  - You click on the Repository menu > Open in Terminal/GitBash.
  - The repository included a "Data" folder with a "example.csv" file.

- From our terminal, if we wanted to change directories to the Data folder, we use the "cd" command and specify the folder name. We can use either an absolute OR a relative file path:
  - Example with an absolute file path:
    - "cd /Users/codingdojo/Documents/GitHub/json-apis-activity/Data"
  - Example with a relative file path:
    - "cd Data"