Fetching Data Using OpenWeatherMap API

Michael Borck

Table of contents

Today

- Emphasize the importance of understanding APIs and data fetching in Python.
- Highlight the skills that will be learned in this session:
  - Fetching data using the OpenWeatherMap API.
  - Parsing JSON data.
  - Handling errors.

Review Homework

- Review the provided notebook on basics of Python

Introduction to OpenWeatherMap API

- The OpenWeatherMap API provides weather data for various locations. You need to sign up and get an API key to access the data.
- Steps to Get API Key:
  1. Go to OpenWeatherMap and sign up for an account.
  2. After signing up, navigate to the API keys section and generate a new API key.
  3. Keep this API key safe, as you will need it to make API requests.

Fetching Data

- Demonstrate how to use the `requests` library to fetch data from the OpenWeatherMap API.
- Introduce the concept of HTTP requests and responses.
- Show how to handle errors and exceptions in Python.

```
import requests

def fetch_weather_data(api_key, location):
    url = f"http://api.openweathermap.org/data/2.5/weather?q={location}&appid={api_key}"
    response = requests.get(url)
    if response.status_code == 200:
        data = response.json()
        return data
    else:
        print(f"Failed to fetch data: {response.status_code}")
        return None
```

```
api_key = 'your_api_key_here'
location = 'London'
data = fetch_weather_data(api_key, location)
data
```

Parsing JSON Data

- Introduce the concept of JSON data and how it is used to represent data.
- Demonstrate how to use the `json` library to parse JSON data in Python.
- Highlight the importance of data parsing and how it is used in the weather dashboard project.

```
def parse_weather_data(data):
    if data:
        weather = {
            "Location": data["name"],
            "Temperature (K)": data["main"]["temp"],
            "Humidity (%)": data["main"]["humidity"],
            "Weather": data["weather"][0]["description"]
        }
        return weather
    else:
        return None

parsed_data = parse_weather_data(data)
parsed_data
```

Saving Data to a CSV File

- We'll save the fetched data to a CSV file using pandas.

```
import pandas as pd

def save_to_csv(data, filename):
    df = pd.DataFrame([data])
    df.to_csv(filename, index=False)

save_to_csv(parsed_data, 'data/raw/weather_data.csv')
```

```
df = pd.read_csv('data/raw/weather_data.csv')
df
```

Error Handling

- Introduce the concept of error handling in Python and its importance.
- Demonstrate how to use try-except blocks to handle errors in Python.
- Highlight the importance of error handling in the weather dashboard project.

```
invalid_location = 'InvalidCity'
invalid_data = fetch_weather_data(api_key, invalid_location)
invalid_data
```

Breakout Room Activity

- Emphasize the importance of teamwork and collaboration in the project.
- Encourage students to work together to fetch and parse data for different cities.
- Encourage students to share their findings and discuss challenges.

Q&A and Wrap-Up

- In this session, we learned how to fetch weather data from the OpenWeatherMap API, parse the JSON response, and save the data to a CSV file. We also covered basic error handling to manage failed API requests.
- Any questions"
- The importance of practicing and experimenting with the concepts learned in the session.
- Next session, we will focus on managing data with CSV and SQL.

Homework

- Experiment with fetching weather data for different cities.
- Explore the OpenWeatherMap API documentation to see what other data you can fetch.