Introduction to Python for Social Science

Lecture 8 - APIs and Selenium

Musashi Harukawa, DPIR

8th Week Hilary 2021

Lecture Roadmap

Last Week

- HTTP requests and Internet fundamentals
- Regular Expressions

This Week

- APIs
  - Twitter’s Academic Track
- Browser Automation

APIs

What is an API?

- *Application Programming Interface*
- *Interface*: Specialized endpoint
  - Specific query syntax
  - Returns defined data packets
- We are interested in *Web APIs*

Web API Examples

- Twitter
- Reddit
- NYTimes
- The Guardian
- Spotify
- Netflix

API Mechanics

- REST vs SOAP
- RESTful APIs loosely based on HTTP methods
  - Accept HTTP-like requests to access server-side assets
  - Return the payload usually as JSON or XML
  - *Stateless*: no server-side session information

Twitter's API

- Many different Twitter APIs and endpoints (Standard, Premium, Enterprise, and **Academic**)
- **Academic Research product track** has following endpoints:
  - *Full-archive search*: (Almost) everything back to 2006!
  - *Recent search*: Last 7 days, higher volumes
  - *Filtered stream*: Real-time filtered stream, capped at 1% of total volume
  - *Sampled stream*: 1% of all new Tweets in real-time
  - *Tweet and User Lookup*: Look up user/tweet by id
  - and more

Applying for Access

- The Academic Research track has the following criteria:
  - Master's student or above (doctoral candidate, post-doc, faculty, researcher, etc.)
  - Clearly defined research objective and specific plans for how you will use the Twitter data
  - Non-commercial use
- You can apply here

Using the API (with Python)

- We can use Python to generate requests to interact with Twitter's API
- Twitter provides a "wrapper" package: `searchtweets-v2`
- Documentation provided here and here

Managing Credentials

- Once you are granted access, you will be given a set of credentials for your project/application.
- Store these securely, i.e. do not post them somewhere public.
- Place them in a credentials `yaml` file that looks like the following:

```
search_tweets_v2:
  endpoint:  https://api.twitter.com/2/tweets/search/all
  consumer_key: <CONSUMER_KEY>
  consumer_secret: <CONSUMER_SECRET>
  bearer_token: <BEARER_TOKEN>
```

Writing and Sending Requests

- To be discussed in the coding tutorial

Browser Automation

When does static scraping fail?

- Sometimes the information you need is not contained in the `html` returned by a request.
- Obtaining that information may require interaction with the web app.
  - Log in
  - Dynamic elements
- Some web servers block suspicious activity

Static vs Dynamic Webpages

- Interactive $\not\to$ Dynamic
- Dynamic page source generation
  - Server-side: `php`
  - Client-side: javascript

Browser Automation

- Selenium Browser Automation Framework
- Designed for testing, but useful for scraping!
- Any and all browser actions can be emulated/automated.

Using Selenium

- Actions are methods of a “WebDriver” object.
- Many similar methods to `BeautifulSoup` for navigating DOM.
  - Search for elements by id, regex, xpath, etc.
- Selenium IDE allows you to record your own usage and codify it afterwards.

Considerations

- Race conditions—“wait”s are your friend!
- Overhead/overkill
- Human-like automation