Team: Campbell Boulanger, Lydia Boatright, Emma Calhoun, James Lee, and Taylor Whitefield
Strategic Management - MGMT 4513
Professor Jason Kiley
26th March 2020

Learning Project Resources - Web Scraping with Python

Topic:
Our project covered what web scraping is, the ethical implications of it, our program we created to showcase web scraping in action, the alternatives available (what a API is), and why you should care to know about web scraping moving forward in your career.

Link to video: https://vimeo.com/401033968

Resources explained:

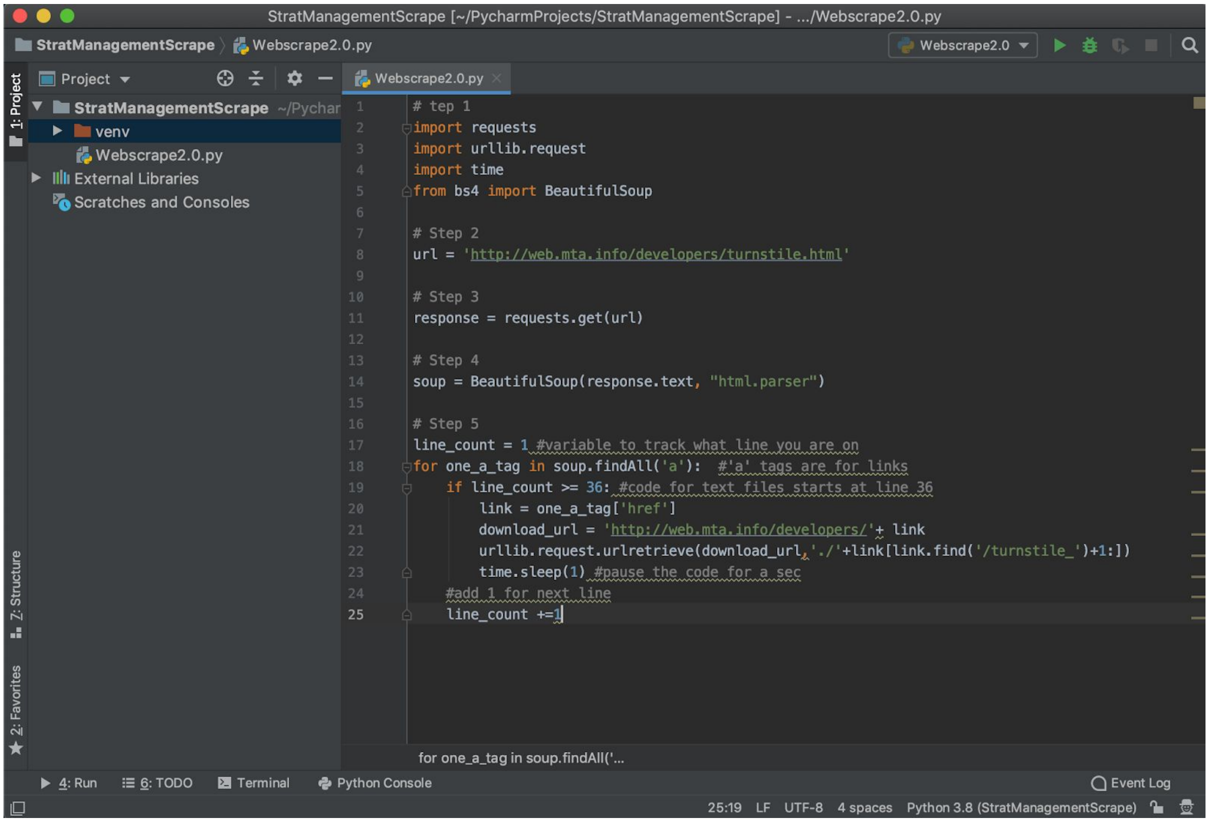
- https://scrapinghub.com/what-is-web-scraping/
  - The general definition was paraphrased from this useful website and some of the information it provided was quoted in the notes of the slides. Such as how we perform the function of web scraping everyday by copying and pasting. This website explained how to use web scraping in a way that anyone can comprehend.
- https://www.octoparse.com/blog/what-is-web-scraping
  - The benefits of knowing how and why to web scrape were inspired by or paraphrased from this website which also clearly defines what web scraping is. A graphic was used from this website and a caption with the hyperlink was provided under the image.
- https://www.promptcloud.com/blog/web-scraping-better-alternative-to-api/
  - The statistics as to why web scraping is better than an API scraper and why people still used the API alternative were pulled from this site. A paraphrase was used when describing the situation when talking about why the alternatives were used.
- https://www.saashub.com/scraper-api-alternatives
  - SAAS hub was used when finding the API alternatives. We used their rankings and their lists when looking for applications we wanted to talk about. Along with using their rankings, we paraphrased what they had said about what the applications were used for and what made them effective
- https://github.com/jtkiley/text_seminar/blob/master/notebooks/3a_retrieval1.ipynb
  - GitHub is a great resource for any person learning to code. Our professor provided this resource to us to help us get a basic understanding of how to web scrape and the logic behind it.

- https://www.youtube.com/watch?v=E5cSNSeBhjw
    - This simple 30 minute video showcased the basics of creating a web scraping program using PyCharm ,importing libraries (BeautifulSoup and Requests), and some of the syntax logic behind creating your first program. It was the first video that we found that made sense and was easy to follow along.
- https://towardsdatascience.com/how-to-web-scrape-with-python-in-4-minutes-bc49186a8460
    - Using the information taught in the above video, we combined using PyCharm while following this guide to scrape the new york MTA website. Everything is also explained very easily like the previous sources. It is essential to understand how to install python and the libraries necessary beforehand because the article does not show you how to do that. It is assumed you have a basic understanding of python. We recommend you follow the video and use the github resource first before jumping into this tutorial.

Programs used:

PyCharm

Code:


StratManagementScrape [~/PycharmProjects/StratManagementScrape] - .../Webscrape2.0.py
StratManagementScrape
Webscrape2.0.py
Webscrape2.0
1: Project
Project
StratManagementScrape ~/Pychar
venv
Webscrape2.0.py
External Libraries
Scratches and Consoles
Z: Structure
2: Favorites
for one_a_tag in soup.findAll('...
4: Run
6: TODO
Terminal
Python Console
Event Log
25:19 LF UTF-8 4 spaces Python 3.8 (StratManagementScrape)

```python
# tep 1
import requests
import urllib.request
import time
from bs4 import BeautifulSoup

# Step 2
url = 'http://web.mta.info/developers/turnstile.html'

# Step 3
response = requests.get(url)

# Step 4
soup = BeautifulSoup(response.text, "html.parser")

# Step 5
line_count = 1 #variable to track what line you are on
for one_a_tag in soup.findAll('a'):  #'a' tags are for links
    if line_count >= 36: #code for text files starts at line 36
        link = one_a_tag['href']
        download_url = 'http://web.mta.info/developers/'+ link
        urllib.request.urlretrieve(download_url,'./'+link[link.find('/turnstile_')+1:])
        time.sleep(1) #pause the code for a sec
    #add 1 for next line
    line_count +=1
```