

Python Web Scraper

Want to pull the latest working version (if current version is still under development)?

✓ Get Commit: **18d45a5**

Run `landing_page.py` to initialize the flask application and the web server.

```
* Serving Flask app "landing_page" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 221-025-799
* Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [21/Nov/2019 12:10:07] "GET /api/products HTTP/1.1" 200 -
127.0.0.1 - - [21/Nov/2019 12:10:08] "GET /favicon.ico HTTP/1.1" 404 -
```

To get the part numbers: The CSV file with the product number will be read by `read_csv()` to have an `iter_set` of product numbers, `i_s_part_nums`.

To search the website: (in this case newegg.com) the `iter_set` of part numbers, `i_s_part_nums`, is stepped through per each individual part number to create a custom url for that product search, `part_num`. This is passed into `get_custom_url()` to create a searchable page to scrape from.

To get data off the site and store it into a Database:

`get_product_details()` - The custom url must be opened and essentially parsed using **BeautifulSoup** a Python library for pulling data out of HTML and XML files. This data is basically looped through to find the right div/section for the product list results. Then the appropriate Title, Price, Image, etc.. are pulled into the sqlite3 database.

Troubleshooting steps:

VIRTUALENV

How to run virtualenv (env/scripts/activate.ps1) on windows powershell:

1) Navigate to dir

2) RUN: PS C:\Users\Preston\web_scraping_with_python\scraper\python_web_scraper>

set-executionpolicy remotesigned

3) [Y] Yes ... : y [enter]

4) RUN: PS C:\Users\Preston\web_scraping_with_python\scraper\python_web_scraper>

.\env\scripts\activate.ps1

5) (env) PS C:\Users\Preston\web_scraping_with_python\scraper\python_web_scraper>

^^^ env should appear before your path.

python_web_scraper

1. Take in .csv files as searchable product data
2. Create custom url handler to search newegg.com/PRODUCT_PART_NUMBER
3. Searching newegg's results and separating out whether products are found or not found
4. Selecting the best result-> pulling the Title, Price, and Product Image
5. Returning the found data for each .csv entry into a separate .csv style DB- sqlite3
6. Returning data to program and printing to screen products information