Web Scraping with Python

By Zachary King

What is Web Scraping?

Web Scraping is the process of using a script or computer program to retrieve information from the Internet.

The process is usually automatic but can involve manual input if desired.

Purpose of Web Scraping

- ➢ Web scraping makes it easy to retrieve exactly what you need from a webpage.
- ➢ No tedious searching of long--or even short--pages manually.
- ➢ Statistical programs such as for research, testing, tracking, etc.
- ➢ Automate common visits to the web

Applications

- Scrape product pages from retailer or manufacturer websites to show in their own website or provide specs/price comparison
- Scrape product reviews from retailers to detect fraudulent reviews
- Scrape news websites for analysis, often for providing better targeted news to their audience
- Scrape sports pages for stat tracking on individual teams or players
- Scrape your Facebook news feed for your own Facebook application! (or other social media)

General Process

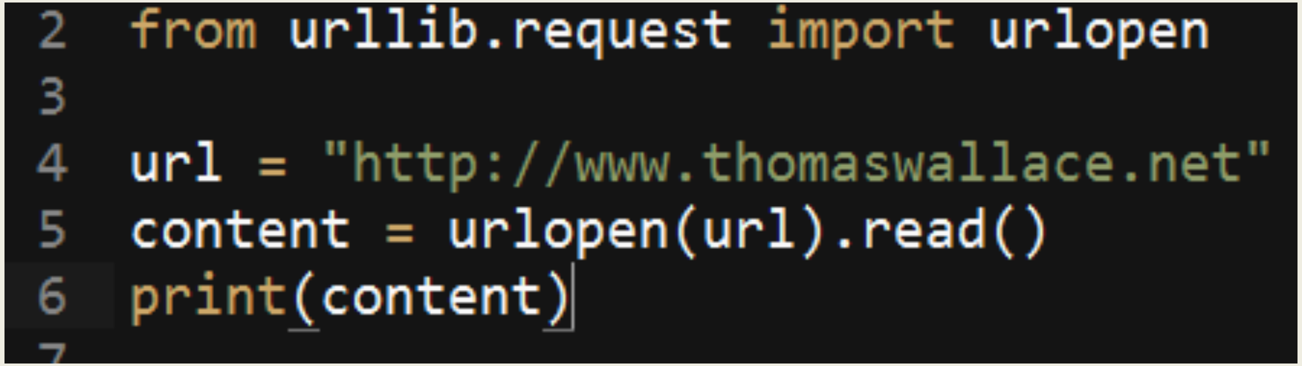
1. Fetch a web page
2. Download web page content (optional)
3. Parse data (HTML)
4. Apply parsed data (your usage)

Using Python

Some packages:
-bs4 (BeautifulSoup4)**
-urllib2 (for Python 2)
-urllib (for Python 3)**
-requests (for Python 3)
-urllib.request (Python 3)**

Go Fetch!

To simply get the HTML content of a web page and output it:

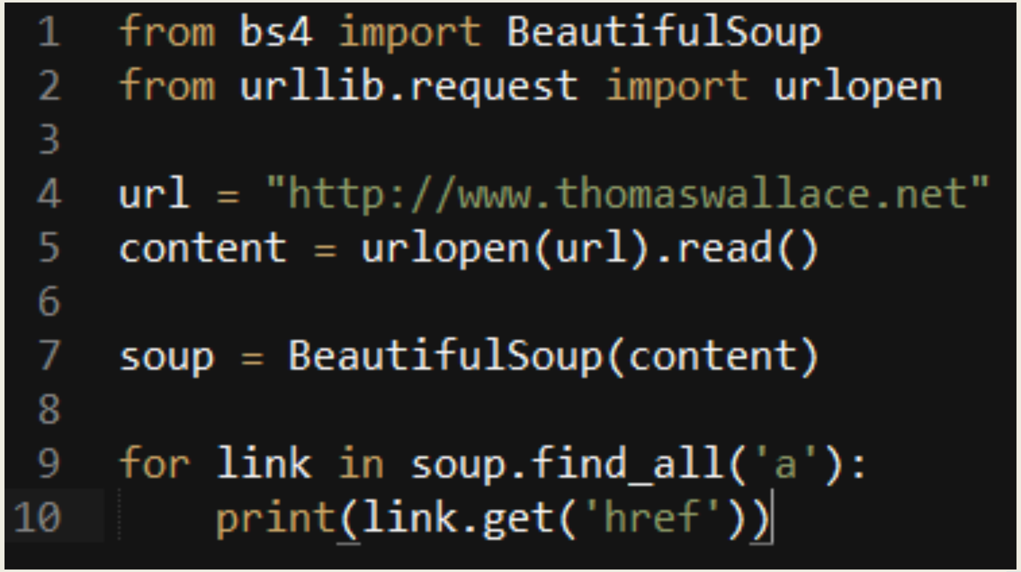


```python
from urllib.request import urlopen

url = "http://www.thomaswallace.net"
content = urlopen(url).read()
print(content)
```

Specific Searches

With BeautifulSoup, create a “soup” object that allows for easy searching within the contents of the web page.



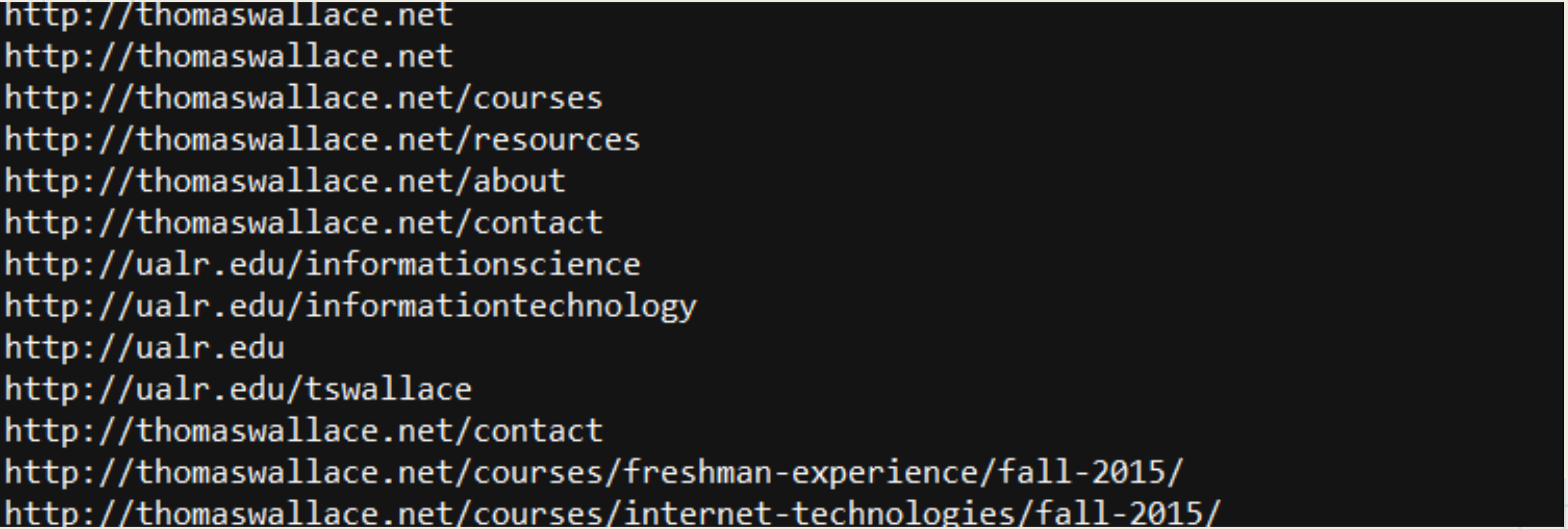
```python
from bs4 import BeautifulSoup
from urllib.request import urlopen

url = "http://www.thomaswallace.net"
content = urlopen(url).read()

soup = BeautifulSoup(content)

for link in soup.find_all('a'):
    print(link.get('href'))
```

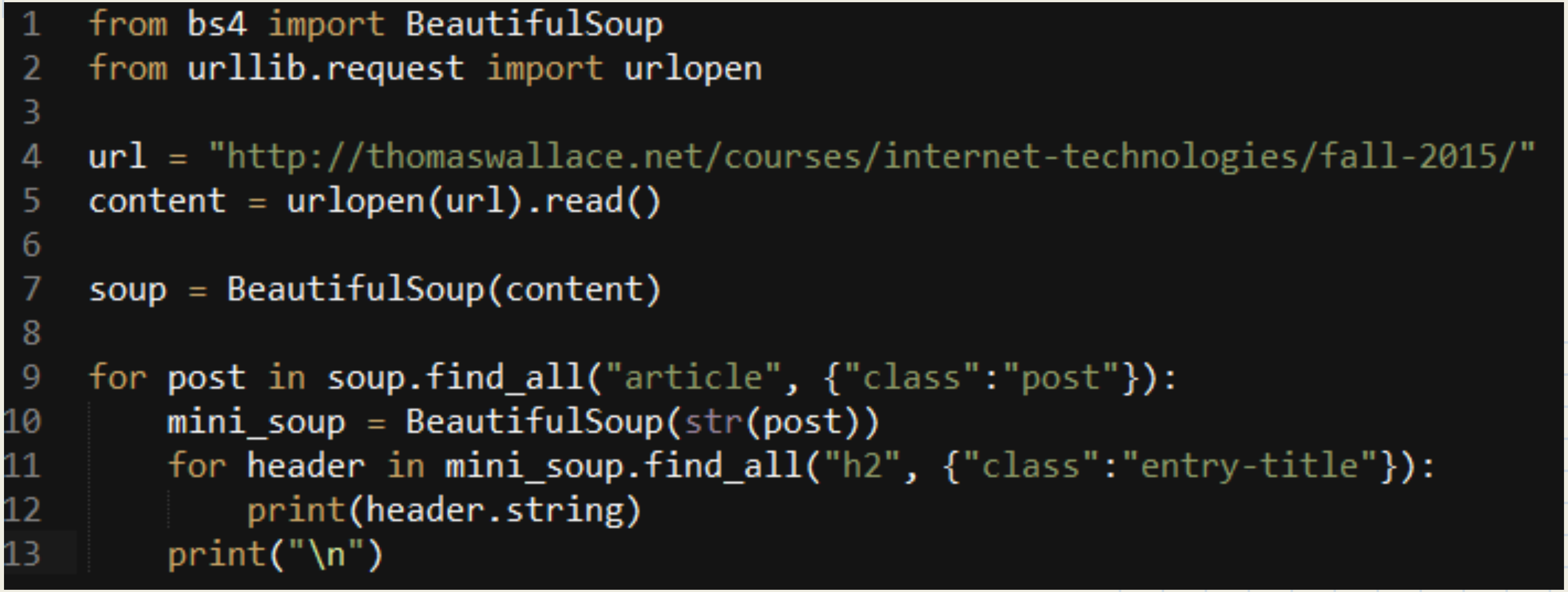
Output



```
http://thomaswallace.net
http://thomaswallace.net
http://thomaswallace.net/courses
http://thomaswallace.net/resources
http://thomaswallace.net/about
http://thomaswallace.net/contact
http://ualr.edu/informationscience
http://ualr.edu/informationtechnology
http://ualr.edu
http://ualr.edu/tswallace
http://thomaswallace.net/contact
http://thomaswallace.net/courses/freshman-experience/fall-2015/
http://thomaswallace.net/courses/internet-technologies/fall-2015/
```

*More Specific Searches

Use multiple “soups” to search specific parts of the web page.



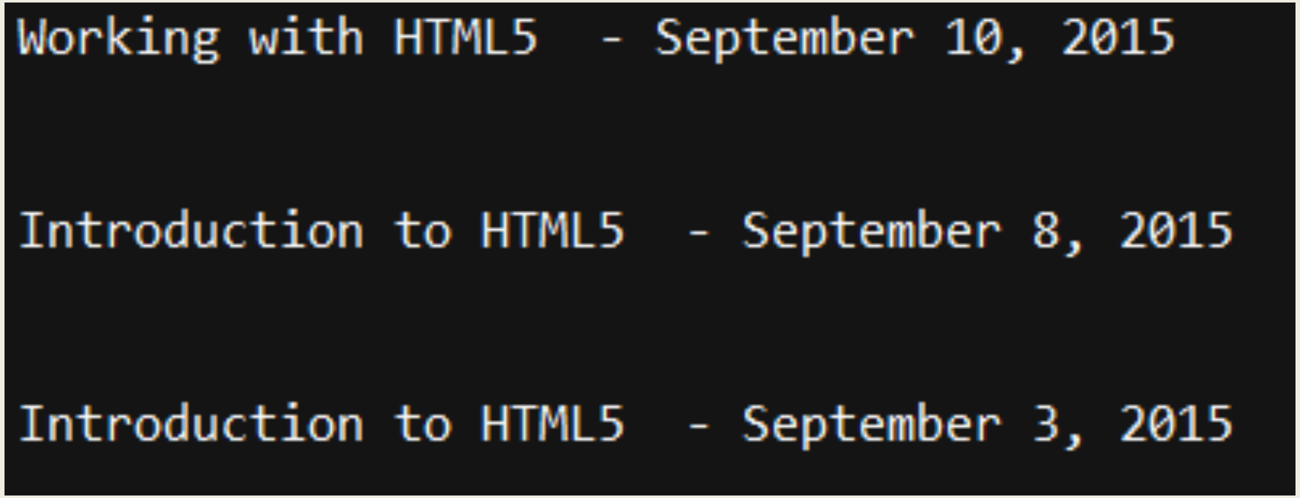
```python
from bs4 import BeautifulSoup
from urllib.request import urlopen

url = "http://thomaswallace.net/courses/internet-technologies/fall-2015/"
content = urlopen(url).read()

soup = BeautifulSoup(content)

for post in soup.find_all("article", {"class":"post"}):
    mini_soup = BeautifulSoup(str(post))
    for header in mini_soup.find_all("h2", {"class":"entry-title"}):
        print(header.string)
    print("\n")
```

Output



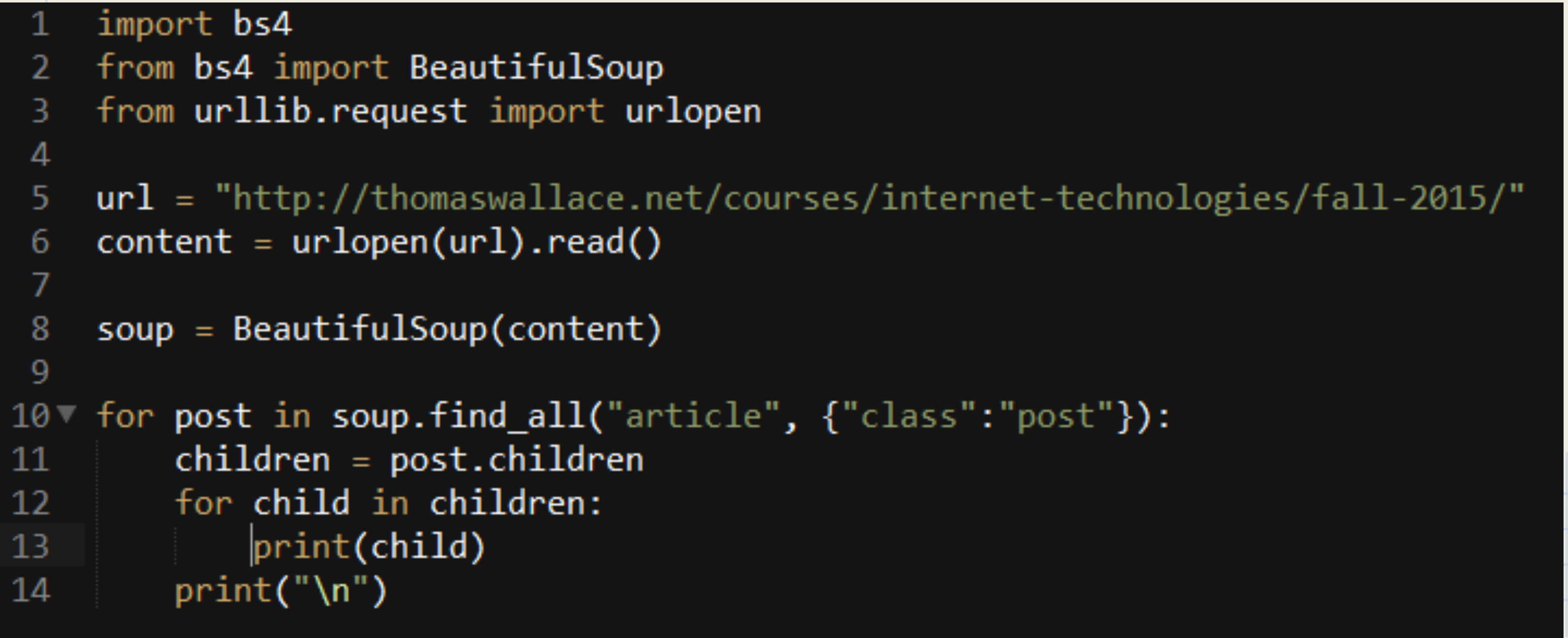
```
Working with HTML5  - September 10, 2015


Introduction to HTML5  - September 8, 2015


Introduction to HTML5  - September 3, 2015
```

Child Elements

An approach to retrieving all the child elements for a given tag are by using the ***.children*** attribute of *BeautifulSoup* objects.



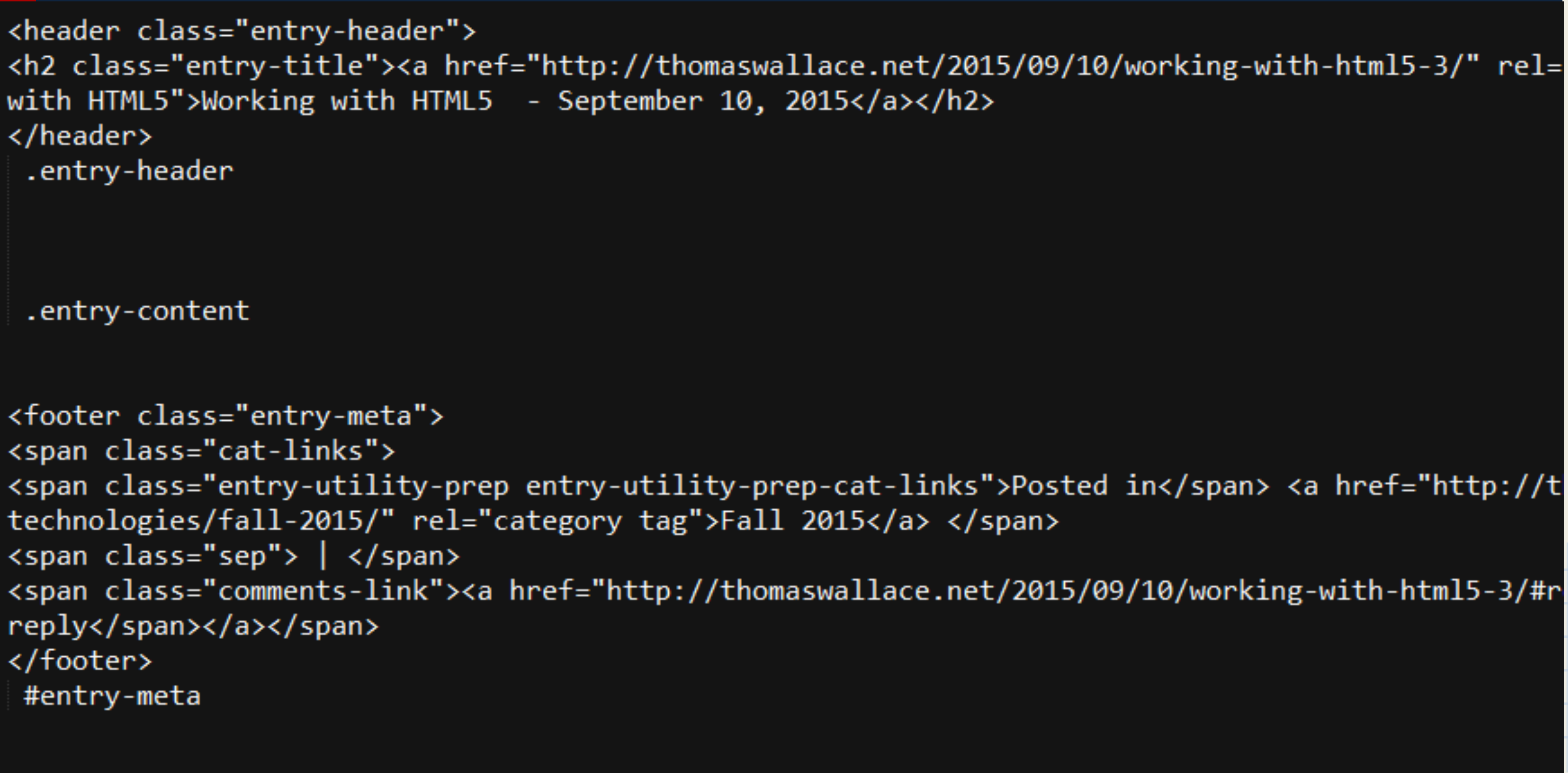
```python
import bs4
from bs4 import BeautifulSoup
from urllib.request import urlopen

url = "http://thomaswallace.net/courses/internet-technologies/fall-2015/"
content = urlopen(url).read()

soup = BeautifulSoup(content)

for post in soup.find_all("article", {"class":"post"}):
    children = post.children
    for child in children:
        print(child)
    print("\n")
```

Output



```
<header class="entry-header">
<h2 class="entry-title"><a href="http://thomaswallace.net/2015/09/10/working-with-html5-3/" rel=
with HTML5">Working with HTML5  - September 10, 2015</a></h2>
</header>
 .entry-header




 .entry-content


<footer class="entry-meta">
<span class="cat-links">
<span class="entry-utility-prep entry-utility-prep-cat-links">Posted in</span> <a href="http://t
technologies/fall-2015/" rel="category tag">Fall 2015</a> </span>
<span class="sep"> | </span>
<span class="comments-link"><a href="http://thomaswallace.net/2015/09/10/working-with-html5-3/#r
reply</span></a></span>
</footer>
 #entry-meta
```

Extending your Scraper

I have my scraped data, now what?

- ➢ Graphs/charts for visual representation
- ➢ Output to a file
- ➢ Store in an organized manner (data structures)
- ➢ Reformat into a new web page

What Now?

- ➢ Bare in mind the legality of web scraping (it’s a blurry line).
- ➢ Always get the green light from the owner of the site (preferably recorded/signed), before scraping their data.
- ➢ Check out the docs for BeautifulSoup at http://www.crummy.com/software/BeautifulSoup/bs4/doc/
- ➢ Take a refresher with the bs4 beginner article at http://www.pythonforbeginners.com/python-on-the-web/beautifulsoup-4-python/

Questions?

You can download all of my example files from this presentation, as well as my more complete Python web scraping files from my GitHub at **https://github.com/zach-king/Python-Web-Scraping**