Programming in Python:

Web Scraping Project

Bo J. Taylor

Western Governors University

Programming in Python: Web Scraping Project

The objective of this assessment is to scrap all hyperlinks embedded in the hypertext markup language (HTML) of the United States (US) Census Bureau Population and Housing Unit Estimates page. The final product is a comma separated value (CSV) file of all unique Uniform Resource Locators (URL) with absolute paths.

My first step was to retrieve the HTML content to be parsed (Figure 1). To do that, I used both the OS and Requests libraries. The using the OS library, my script determines if the HTML file already exists in the current working directory via the os.path.isfile() function. If the file does not, then the content is retrieved using functions provided by the Requests library. The HTML file is then constructed through the .open(), .iter\_content(), and .write() functions to open a new file, iterate over a specified number of bytes of the page’s data, and write that data to the local HTML file, respectively.

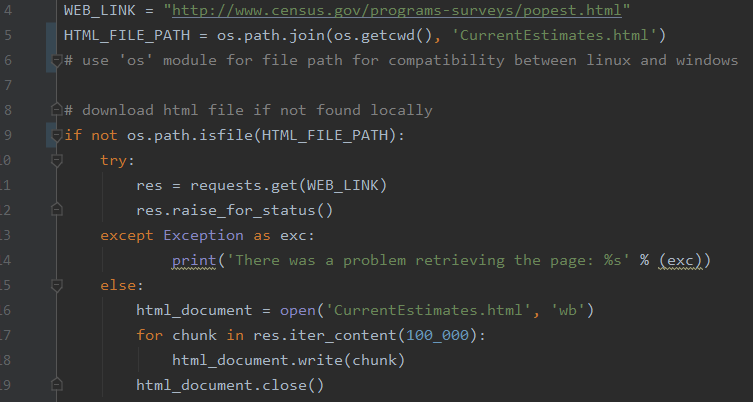


Figure 1. HTML retrieval. This figure contains a snapshot of the code responsible for fetching the website’s HTML content and storing it locally.

After downloading the HTML content to a local file, I then created a BeautifulSoup object. The BeautifulSoup (BS) library provides tools to simplify parsing and working with HTML, in contrast to writing regular expressions and manually manipulating the content from scratch. Creating a BS object was as simple as passing the file to the BS constructor and indicating the content to be parsed was HTML.

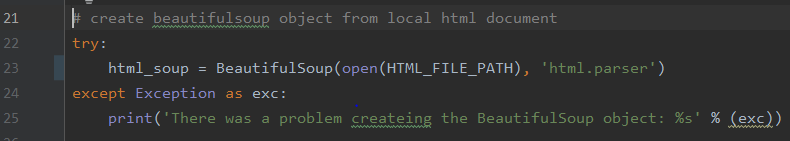


Figure 2. BeautifulSoup object creation. This figure contains code handling the initialization.

Once the BS object was created, I then generated a list of all links in the page. Using the .find\_all() function, I iterated through all the “a” HTML tags, retrieving the URL value with the .get() function. The Urllib library provides a simple function that handles absolute and relative URL’s. By passing the source HTML URL and the URL of each site in this iteration loop, I used this function to append only absolute URL’s to the list of links.

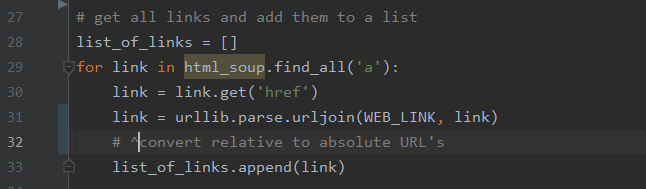


Figure 3. Link parsing. This figure illustrates handling relative URL’s while all links are parsed.

C.    Explain how the program ensures that relative links are saved as absolute URIs in the output file. Identify the code segment that executes this action as part of your explanation.

D.    Explain how the program ensures that there are no duplicated links in the output file. Identify the code that executes this action as part of your explanation.

References

Sweigart, A. (2015). Automate the boring stuff with Python: Practical programming for total beginners. San Francisco, CA: No Starch Press.

Anonymous. (n.d.). C 740. *Fundamentals/Statistics for Data Analytics V5.* Retrieved from https://learn.zybooks.com/zybook/WGUC740V52018/chapter/1/section/1?modal\_name=about-zybook

Kumashiro, Suzanne S. (2019). AFI 33-332. *Air Force Privacy and Civil Liberties Program*. Retrieved from <https://static.e-publishing.af.mil/production/1/saf_cio_a6/publication/afi33-332/afi33-332.pdf>

McCarron, Daniel [Daniel McCarron]. (2011, December 8). Residuals and Residual plots on Excel [Video file]. Retrieved from <https://www.youtube.com/watch?v=6HcyewrPNEw>