Web Scraping

@June 2, 2022

Documentation for Project:

In this demonstration, I plan to start with an introduction to what web scrapping is and its practical uses. Then, we look into what regex is and why it is not as efficient and hence made the module BeautifulSoup increasingly popular. I have made a basic project that collects data from the website https://pythonjobs.github.io/, arranges the data accordingly and then converts it into a csv file. Additionally, I show other functions of BeautifulSoup module and other functionality like counting how many jobs have the word python or in a specific country.

Script:

15 minute video

Outline -

1. What is Web scraping? What is it used for?
2. What is regex? How can you use regex to scrape the web?
3. What is BeautifulSoup module?
4. Program to scrape the https://pythonjobs.github.io/ website to convert data into a dictionary and then converting it into a csv file
5. Other functions in the module, and some other examples of its usage.

Information for Script -

It is safe to say that **web scraping has become an essential skill to acquire in today's digital world**, not only for tech companies and tech positions, but also for non-

tech jobs. The ability to compile large datasets is fundamental to Big Data analytics, Machine Learning, and Artificial Intelligence.

Script -

Web Scraping is exactly what it sounds like. It is the process of extracting data from websites. The extracted information is collected and then exported into a format that is more useful for the user. It has now become an essential skill to acquire due to its ability to compile large datasets which is fundamental to Big Data analytics, Machine Learning, and Artificial Intelligence.

-

Web Scraping in its most simple sense is implemented through regular expressions. Regex is a sequence of characters that specifies a search pattern in text, commonly known as pattern matching.

explain web scraping using regex code by first opening url and viewing source code

-

BeautifulSoup is a Python library for pulling data out of HTML and XML files, and other markup languages.

explain web scraping using beautiful soup module by first opening url and viewing source code

-

saving the dictionary into a csv file and opening it

alternate method to extract using the beautiful soup module

-

Resources Used -

- https://realpython.com/python-web-scraping-practical-introduction/
- https://realpython.com/beautiful-soup-web-scraper-python/
- https://www.geeksforgeeks.org/implementing-web-scraping-python-beautiful-soup/

Here's a step-by-step outline of this project:

1. Download the webpage using `requests`
2. Parse the HTML source code using beautiful Soup
3. Extract topic names, descriptions, and URLs from page
4. Compile the data and create a CSV file using Pandas

Things to keep in mind:

- `class` **(no underscore)** is one of the reserved keywords in Python:

  > The following identifiers are used as reserved words, or ***keywords*** of the language, and cannot be used as ordinary identifiers:
  >
  > ```
  > False      await      else       import     pass
  > None       break      except     in         raise
  > True       class      finally    is         return
  > and        continue   for        lambda     try
  > as         def        from       nonlocal   while
  > assert     del        global     not        with
  > async      elif       if         or         yield
  > ```

- `class_` **(with underscore)** has no special meaning in general. `bs4` just wanted a CSS "class" param, but `class` is reserved so they chose `class_` with an underscore:

  > Using `class` as a keyword argument will give you a syntax error. As of Beautiful Soup 4.1.2, you can search by CSS class using the keyword argument `class_`:
  >
  > ```
  > soup.find_all("a", class_="sister")
  > ```

Personal Notes -

1. Extract text from HTML using string methods (find(), len(), slicing)
   You can have the issue of spacing in HTML that will definitely mess up with the operations as .find() searches for specific strings and not regex at the moment.
2. import re, to use regex
   the asterisk character (`*`) stands for zero or more of whatever comes just before

the asterisk
re.sub() - used to substitute
non-greedy matching pattern `*?`, which works the same way as `*` except that it matches the shortest possible string of text

3. Browser Object - page = browser.get() function, and then you can do page.soup can use to log into browser subpages using username and password

- the BeautifulSoup library is that it is built on the top of the HTML parsing libraries like html5lib, lxml, html.parser, etc. So BeautifulSoup object and specify the parser library can be created at the same time.
- Source HTML code → Option+Cmmd+U / Option+Cmmd+I
