Building and evaluating web-scraping solutions – how to test the efficiency and the effectiveness of a custom-built web-scraping solution.

Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Silvia Settle, School of Applied Computing, University of Wales Trinity St David

School of
Applied Computing

Introduction

Web scraping offers a way of obtaining these large amounts of information faster and from various sources [1]. The benefit of being able to access a huge amount of publicly available data automatically and at scale [7] is the drive behind a lot of scraping projects [8]. These can be applied to a variety of contexts: market research, goods and service-price comparisons, search-engine optimisation strategy/monitoring, web-application monitoring, social media posts, research and behaviour patterns, etc. [9] Any solution, however, should be carefully considered alongside some of the challenges faced by most web-scraping projects, and especially those running at scale – the challenge of data quality and provisioning with well-suited and finely-tuned infrastructure [12]. Ultimately, any data scraping endeavour requires a well-rounded evaluation of the investment made in the project terms of time, effort and resources versus the return from it – how useful is the data obtained.

The aim of this project is therefore to evaluate how to implement an efficient and effective web-scraper application. The objective is to investigate different approaches to web-scraping and examine how the efficiency and effectiveness of custom-built web-scraping solutions can be measured against a set of performance testing metrics. To achieve this, a closer look will be taken into the **approaches and challenges of building a web-scraper**, and as a result, look into **constructing a framework for testing and evaluating the efficiency and effectiveness of a web-scraper**.

Materials and methods

Python's BeautifulSoup4, requests, lxml, selectolax, re libraries were used to develop several prototype solutions.

Pytest was used to develop unit-style tests to test the functionality of the code (effectiveness).

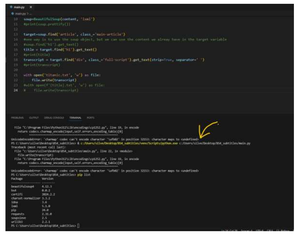
A mock website 'Accessible Eats' was used as a target for the scraping code.

Results

```
In [3]: #import modules
        from bs4 import BeautifulSoup
        import requests
        import re

In [5]: url = "https://www.imdb.com/chart/top-english-movies/?ref_=chtmvm_ql_4"

In [6]: page = requests.get(url)

In [7]: page.content

Out[7]: b'<html>\r\n<head><title>403 Forbidden</title></head>\r\n<body>\r\n<center><h1>403 Forbidden</h1></center>\r\n</body>\r\n</html
        >\r\n'

In [8]: #parse
        soup=BeautifulSoup(page.content, 'html.parser')

In [9]: print(soup.prettify)

        <bound method Tag.prettify of <html>
        <head><title>403 Forbidden</title></head>
        <body>
        <center><h1>403 Forbidden</h1></center>
        </body>
        </html>
        >
```

```
In [40]: #finding text in the content
         re.findall(r'[aA]ccessib[a-z ]{1,}', page.text)

Out[40]: ['Accessible ',
          'Accessible ',
          'accessibility in mind',
          'accessible entrance',
          'Accessible restroom',
          'Accessible ']
```

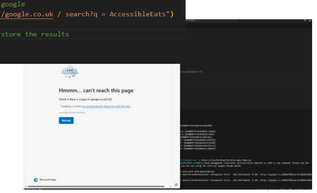
Several prototypes were tested using different modules or frameworks in Python and different parsers. Also different methods of obtaining the target data were tested. Within the context of extracting accessibility data from unknown number of websites, using regular expressions (Regex, re) was considered the most optimal solution.



```
import requests
from bs4 import BeautifulSoup

#making a request
resp=requests.get('https://silviasettle.github.io/AccessibleEats/Pages/about.html')
#check for status code
print(resp)
#parse the HTML
soup=BeautifulSoup(resp.content, 'html.parser')
print(soup.prettify)
```

```
</section>
<section>
<div id="image-carousel">
<!-- Image carousel content goes here -->
<img alt="Wheelchair accessible entrance" src="../Images/outside2.jpg"/>
<img alt="Spacious interior" src="../Images/hall.jpg"/>
<img alt="Accessible restroom" src="../Images/outdoor_space.jpg"/>
<img alt="Sensory space" src="../Images/garden_lights.jpg"/>
<img alt="Sensory space 2" src="../Images/garden_Swan.jpg"/>
<img <="" alt="Aquarium" div="" src="../Images/aquarium.jpg"/>
</div></section>
</main>
<footer>
<p>© 2024 The Accessible Eats. All rights reserved.</p>
</footer>
</body>
</html>>
PS C:\Users\silve\Desktop\ParselScraper>
```

```
##SELENIUM
#import the webdriver
from selenium import webdriver

#create the webdriver object
driver = webdriver.Edge()

#get a result from google
driver.get("https://google.co.uk / search?q = AccessibleEats")

#structure here to store the results
driver.close()
```



Conclusions

- The **aim** of investigating the possible implementations a custom-built web-scraping solution and evaluate its efficiency and effectiveness
- O1: to investigate different approaches to building a web-scraper
- O2: to summarise the challenges of building a web-scraper
- O3: to construct and implement a framework for testing the efficiency and effectiveness of a custom-built web-scraper

Going forwards, instead of building rule-based scrapers, investigating solutions that use machine learning and artificial intelligence approaches and algorithms (bag-of-words for example) may be investigated .

Literature cited

U. Vignesh, R. Parvathi, and R. Goncalves, "Covid Live Multi-Threaded Live COVID 19 Data Scraper," in Structural and Functional Aspects of Biocomputing Systems for Data Processing, IGI Global, 2023, pp. 28–56. doi: 10.4018/978-1-6684-6523-3.ch002.

[2] Ur Rahman, Rizwan, et al. "The Emerging Threats of Web Scrapping to Web Applications Security and Their Defense Mechanism." Encyclopedia of Criminal Activities and the Deep Web, edited by Mehdi Khosrow-Pour D.B.A., IGI Global, 2020, pp. 788-809. https://doi-org.ezproxy.uwtsd.ac.uk/10.4018/978-1-5225-9715-5.ch053

[3] Mitchell, Ryan. Web Scraping with Python. Data Extraction from the Modern Web. O'Reilly, 2024

[4] Zyte, Enterprise web scraping: A guide to scraping at a scale. Whitepaper, 2021

Acknowledgments

I would like to thank my supervisor Mike Dacey for his guidance and support, as well as my family for putting up with me for 6 very long years.

Further information

Please **see https://www.linkedin.com/in/silvia-settle**