Published in Towards Data Science

Vincent Tatan Follow

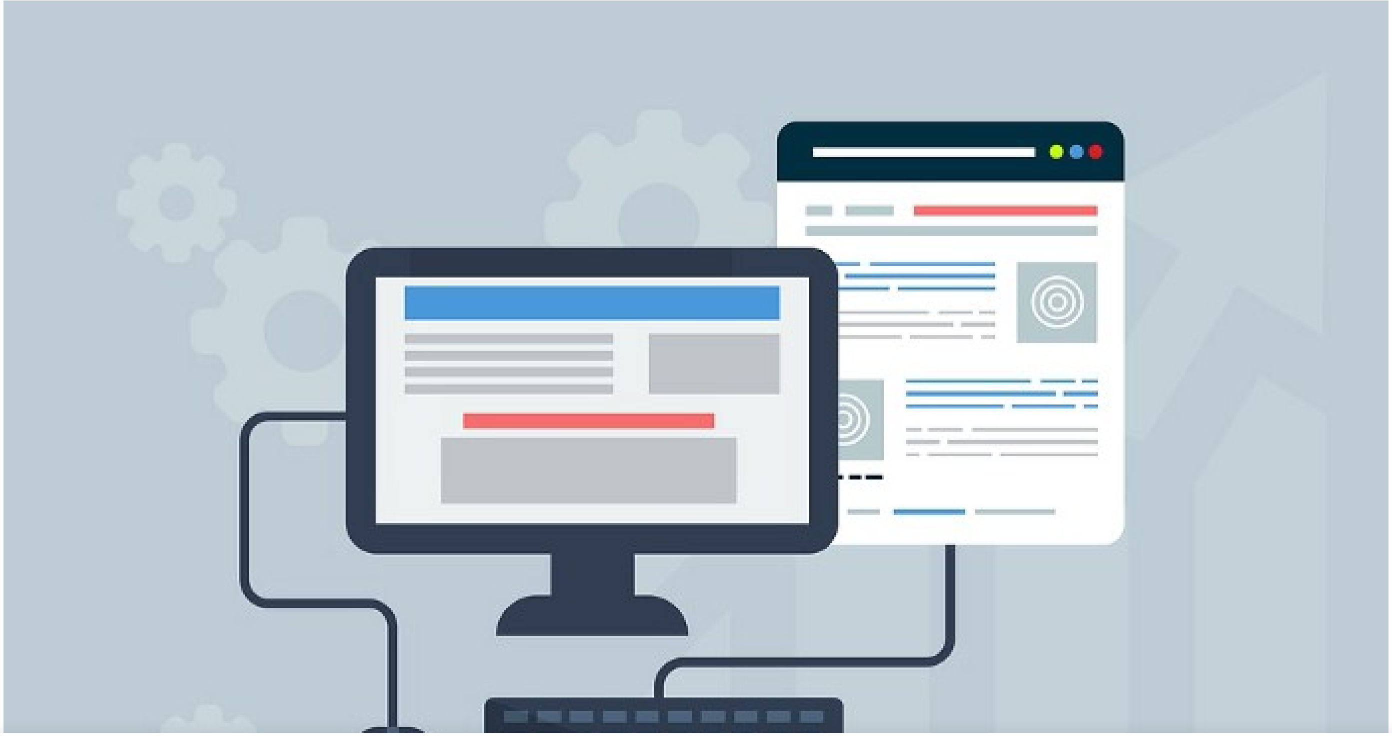
Jun 17, 2019 · 10 min read ★ · Listen

Save

DEFINITIVE GUIDE TO ANALYTICS

In 10 minutes: Web Scraping with Beautiful Soup and Selenium for Data Professionals

Extract Critical Information from Wikipedia and eCommerce Quickly with BS4 and Selenium



WebScraping — Free Image

Introduction

Web Scraping is a process to extract valuable information from websites and online contents. It is a free method to extract information and receive datasets for further analysis. In this era where information is practically highly related to each other, I believe that the need for Web Scraping to extract alternative data is enormous especially for me as a data professional.

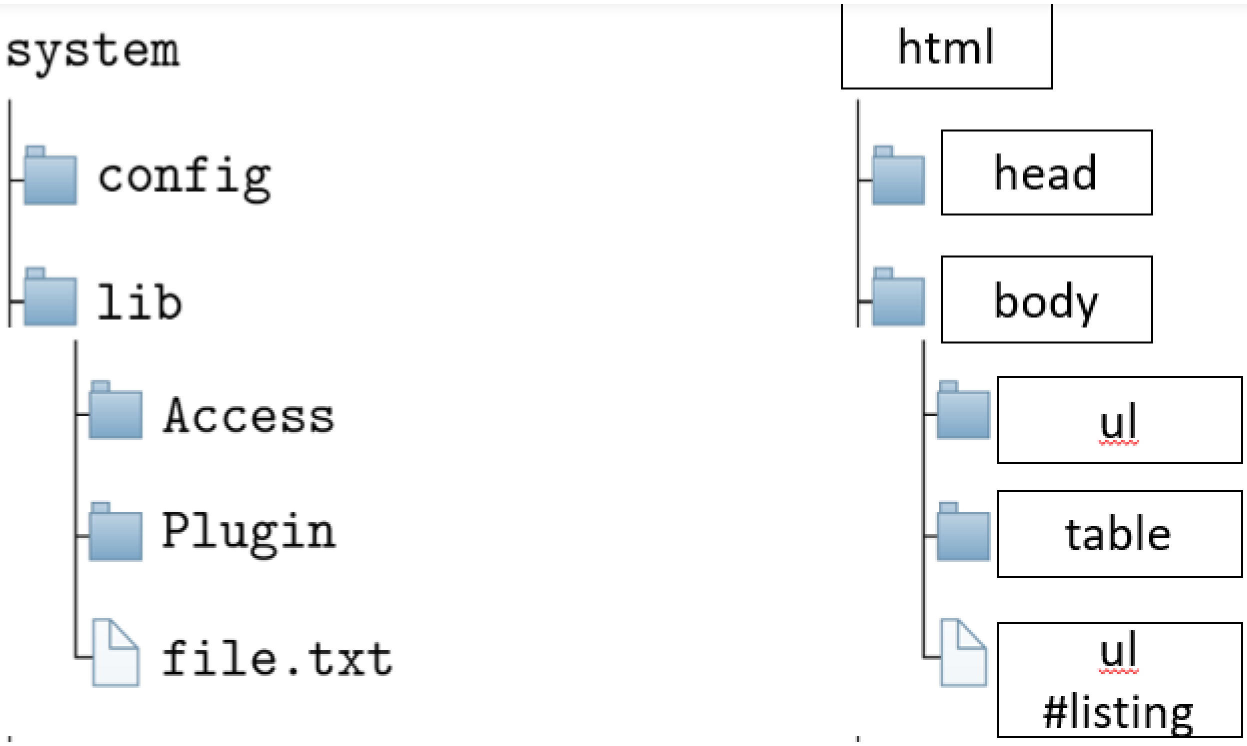
The objective for this publication is for you to understand several ways on scraping any publicly available information using quick and dirty Python Code. Just spend 10 minutes to read this article — or even better, contribute. Then you could get a quick glimpse to code your first Web Scraping tool.

In this article, we are going to learn how to scrape data from Wikipedia and e-commerce (Lazada). We will clean up, process, and save the data into *.csv* file. We will use Beautiful Soup and Selenium as our main Web Scraping Libraries.

What are Beautiful Soup and Selenium

Beautiful Soup

Beautiful Soup parses HTML into an easy machine readable tree format to extract DOM Elements quickly. It allows extraction of a certain paragraph and table elements with certain HTML ID/Class/XPATH.



Parsing of DOM elements compared to Tree Dir Folder

Whenever I need a quick and dirty way approach to extract information online. I will always use BS as my first approach. Usually it would take me in less than 10 minutes within 15 lines of codes to extract.

Beautiful Soup Documentation - Beautiful Soup 4.4.0 documentation

Beautiful Soup 4 is published through PyPi, so if you can't install it with the system packager, you can install it...

www.crummy.com

Selenium

Selenium is a tool designed to automate Web Browser. It is commonly used by Quality Assurance (QA) engineers to automate their testings Selenium Browser application.

Additionally, it is very useful to web scrape because of these automation capabilities:

3. Extracting the DOM elements for browser HTML code

Selenium - Web Browser Automation

Selenium has the support of some of the largest browser vendors who have taken (or are taking) steps to make Selenium a…

www.seleniumhq.org

Coding your first Web Scraping Tool

(Github is available at the end of this article)

Beautiful Soup

Problem Statement

Imagine you were UN ambassadors, aiming to make visits on cities all around the world to discuss about the Kyoto Protocol status on Climate Changes. You need to plan your travel, but you do not know the capital city for each of the country. Therefore, you googled and found this link on Wikipedia.

List of national capitals - Wikipedia

This is a list of national capitals, including capitals of territories and dependencies, non-sovereign states including…

en.wikipedia.org

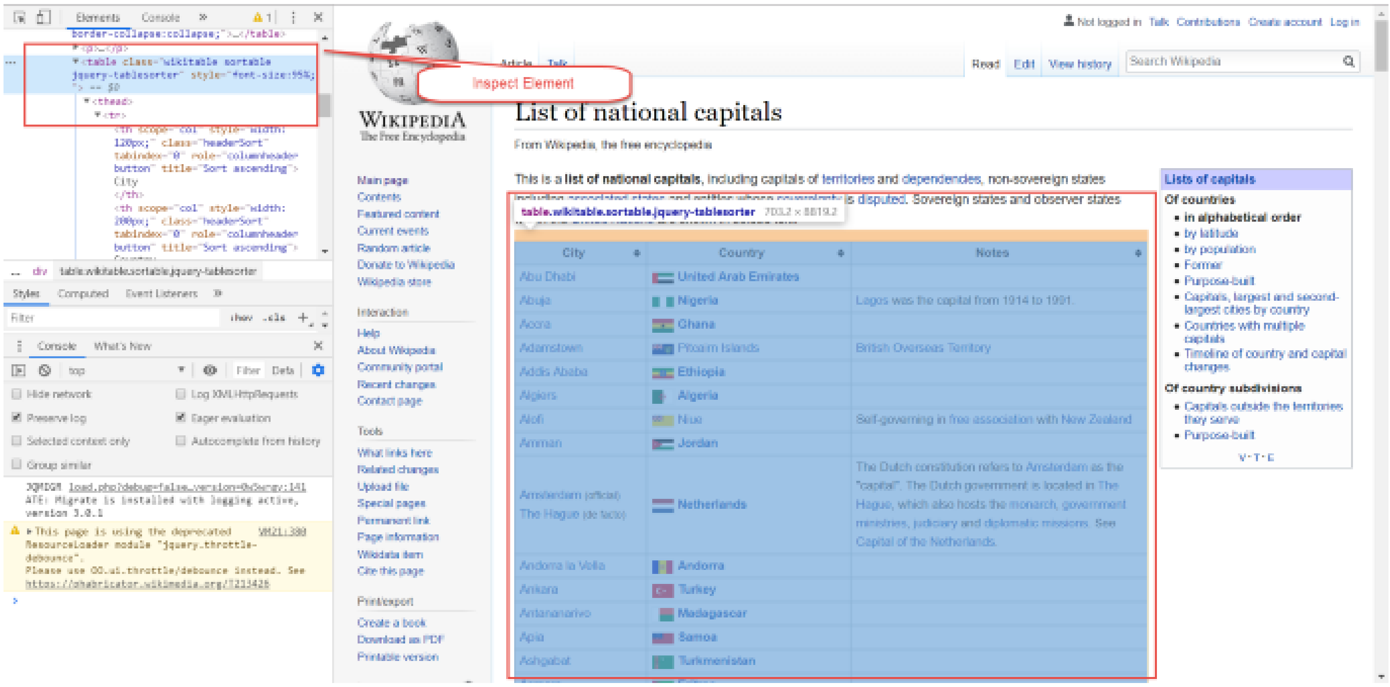
Inside this link, there is a table which maps each country to the capital city. You find this is good, but you do not stop there. As a data scientist and UN ambassador, you want to extract the table from Wikipedia and dump it into your data application. You took up the challenge to write some scripts with Python and BeautifulSoup.

Steps

We will leverage on the following steps:

2. **Check which DOM element the table is referring to**. Right click on your mouse and click on *inspect element*. Shortcut is CTRL+I (inspect) for Chrome Browser.

3. **Click on the inspect button** at the top left corner to highlight the elements you want to extract. Now you know that the element is a table element in the HTML document.



```
border-collapse:collapse;">…</table>
▶<p>…</p>
▼<table class="wikitable sortable
jquery-tablesorter" style="font-size:95%;
"> == $0
  ▼<thead>
    ▼<tr>
      <th scope="col" style="width:
      120px;" class="headerSort"
      tabindex="0" role="columnheader
      button" title="Sort ascending">
      City
      </th>
      <th scope="col" style="width:
      200px;" class="headerSort"
      tabindex="0" role="columnheader
      button" title="Sort ascending">
```

div table.wikitable.sortable.jquery-tablesorter

```
JQMIGR load.php?debug=false…version=0v5wngv:141
ATE: Migrate is installed with logging active,
version 3.0.1
⚠ ▶This page is using the deprecated     VM21:388
ResourceLoader module "jquery.throttle-
debounce".
Please use OO.ui.throttle/debounce instead. See
https://phabricator.wikimedia.org/T213426
```

List of national capitals

From Wikipedia, the free encyclopedia

This is a **list of national capitals**, including capitals of territories and dependencies, non-sovereign states including associated states and entities whose sovereignty is disputed. Sovereign states and observer states

table.wikitable.sortable.jquery-tablesorter 703.2 × 8819.2

City	Country	Notes
Abu Dhabi	United Arab Emirates	
Abuja	Nigeria	Lagos was the capital from 1914 to 1991.
Accra	Ghana	
Adamstown	Pitcairn Islands	British Overseas Territory
Addis Ababa	Ethiopia	
Algiers	Algeria	
Alofi	Niue	Self-governing in free association with New Zealand
Amman	Jordan	
Amsterdam (official) The Hague (de facto)	Netherlands	The Dutch constitution refers to Amsterdam as the "capital". The Dutch government is located in The Hague, which also hosts the monarch, government ministries, judiciary and diplomatic missions. See Capital of the Netherlands.
Andorra la Vella	Andorra	
Ankara	Turkey	
Antananarivo	Madagascar	
Apia	Samoa	
Ashgabat	Turkmenistan	

National Capitals Elements Wikipedia

4. **Add header and url into your requests**. This will create a request into the wikipedia link. The header would be useful to spoof your request so that it looks like it comes from a legitimate browser.

For Wikipedia, it might not matter as all the information is open sourced and publicly available. But for some other sites such as Financial Trading Site (SGX), it might block the requests which do not have legitimate headers.

```
headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 6.3; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/54.0.2840.71
Safari/537.36'}
url = "https://en.wikipedia.org/wiki/List_of_national_capitals"
```

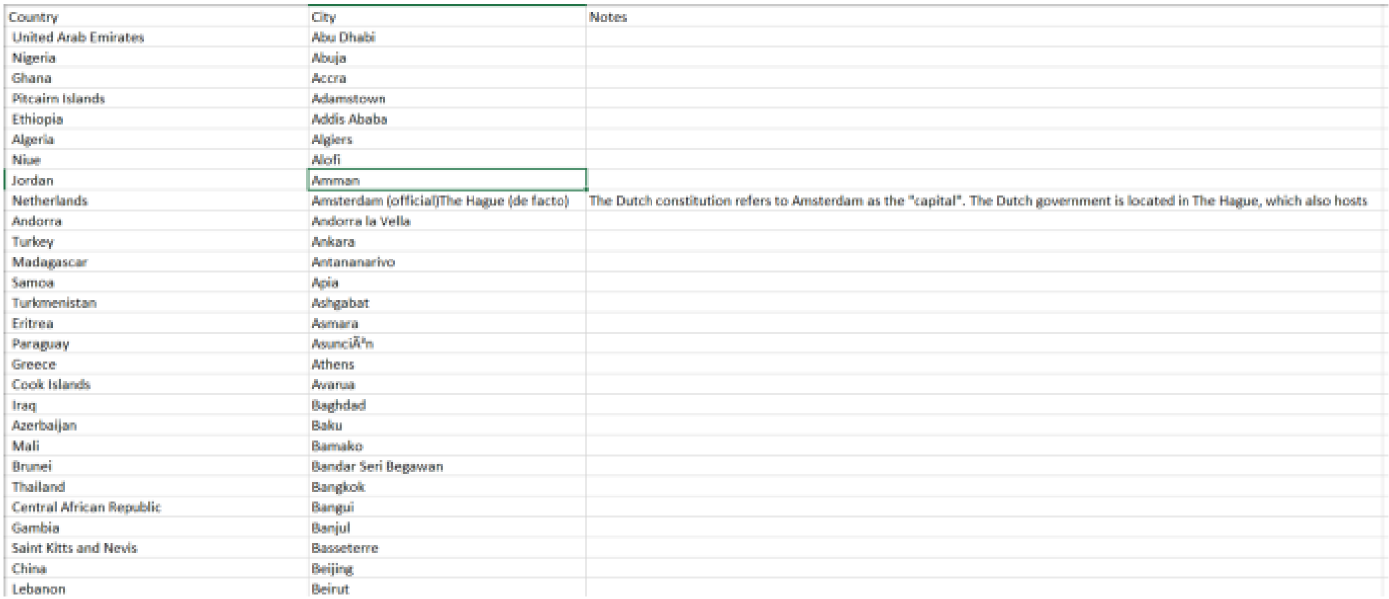
```
soup = BeautifulSoup(r.content, "html.parser")
table = soup.find_all('table')[1]
rows = table.find_all('tr')
row_list = list()
```

6. **Iterate through all of the rows in table** and get through each of the cell to append it into rows and row_list

```
for tr in rows:
    td = tr.find_all('td')
    row = [i.text for i in td]
    row_list.append(row)
```

7. **Create Pandas Dataframe and export data into csv.**

```
df_bs = pd.DataFrame(row_list,columns=['City','Country','Notes'])
df_bs.set_index('Country',inplace=True)
df_bs.to_csv('beautifulsoup.csv')
```



Country	City	Notes
United Arab Emirates	Abu Dhabi	
Nigeria	Abuja	
Ghana	Accra	
Pitcairn Islands	Adamstown	
Ethiopia	Addis Ababa	
Algeria	Algiers	
Niue	Alofi	
Jordan	Amman	
Netherlands	Amsterdam (official)The Hague (de facto)	The Dutch constitution refers to Amsterdam as the "capital". The Dutch government is located in The Hague, which also hosts
Andorra	Andorra la Vella	
Turkey	Ankara	
Madagascar	Antananarivo	
Samoa	Apia	
Turkmenistan	Ashgabat	
Eritrea	Asmara	
Paraguay	AsunciÃ³n	
Greece	Athens	
Cook Islands	Avarua	
Iraq	Baghdad	
Azerbaijan	Baku	
Mali	Bamako	
Brunei	Bandar Seri Begawan	
Thailand	Bangkok	
Central African Republic	Bangui	
Gambia	Banjul	
Saint Kitts and Nevis	Basseterre	
China	Beijing	
Lebanon	Beirut	

Congratulations! You have become a web scraper professional in only 7 steps and within 15 lines of code

The Limitations of Beautiful Soup

So far BS has been really successful to web scrape for us. But I discovered there are some limitations depending on the problems:

1. The requests takes the html response prematurely without waiting for async calls from Javascript to render the browser. This means it does not get the most recent DOM elements that is generated by Javascript async calls (AJAX, etc).
2. Online retailers, such as Amazon or Lazada put anti-bot software throughout the websites which might stop your crawler. These retailers will shut down any requests from Beautiful Soup as it knows that it does not come from legitimate browsers.

> ***Note***
>
> *If we run Beautiful Soup in e commerce websites such as Lazada and Amazon, we will run to this Connection Error which is caused by their anti scraping software to deter bots from making http requests.*
>
> *HTTPSConnectionPool(host='www.amazon.com', port=443): Max retries exceeded with url: / (Caused by SSLError(SSLError(1, '[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed (_ssl.c:833)'),))*

One way to fix it is to use client browsers and automate our browsing behavior. We can achieve this by using Selenium.

All hail Selenium!!

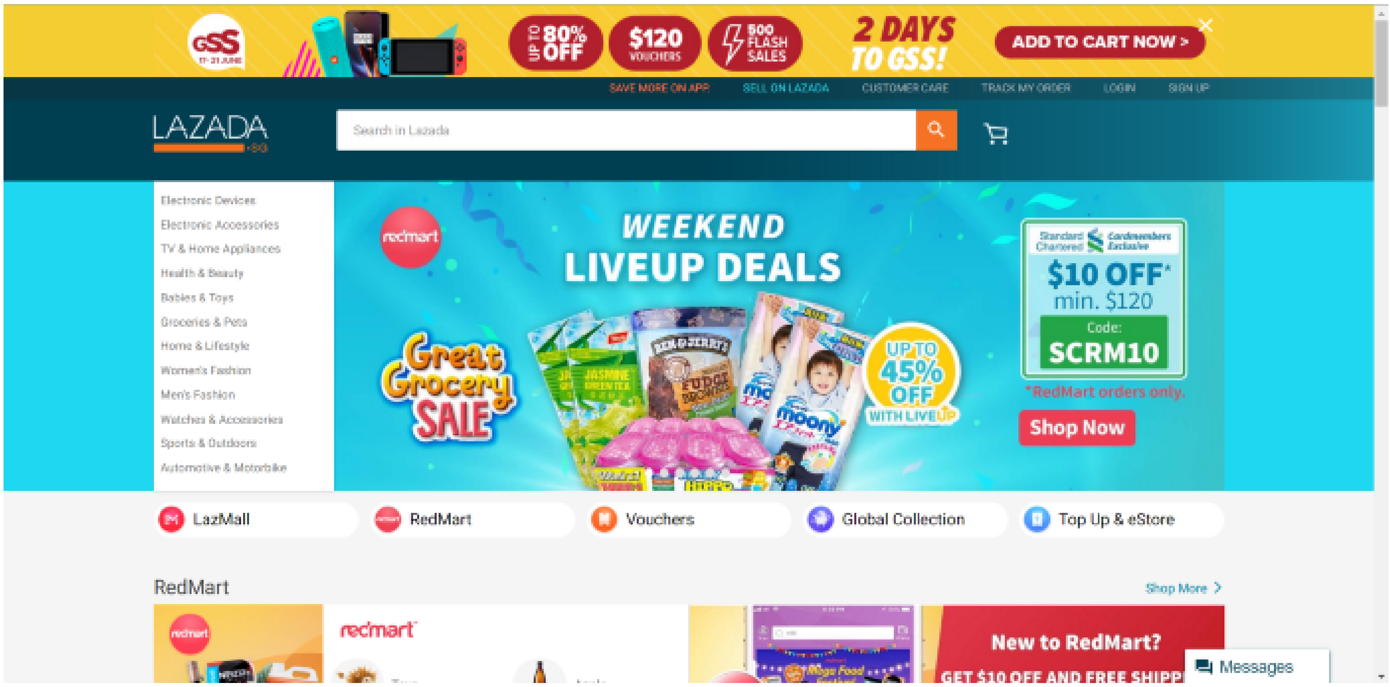
Selenium

Problem Statement

into Excelsheet. This process would be very repetitive, especially if you'd like to collect the data point every day/every hour. This would also be a very time consuming process as it involves many manual clicks and browses to duplicate the information.

> *What if I tell you, you can automate this process:*
>
> *By having Selenium doing the exploration of products and clicking for you.*
>
> *By having Selenium opening your Google Chrome Browser to mimic legitimate user browsing behaviors.*
>
> *By having Selenium pump all of the information into lists and csv files for you.*

Well you're in luck, because all you need to do is write a simple Selenium script and you can now run the web scraping program while having a good night sleep.



Extracting Lazada Information and Products are time consuming and repetitive

Setting Up

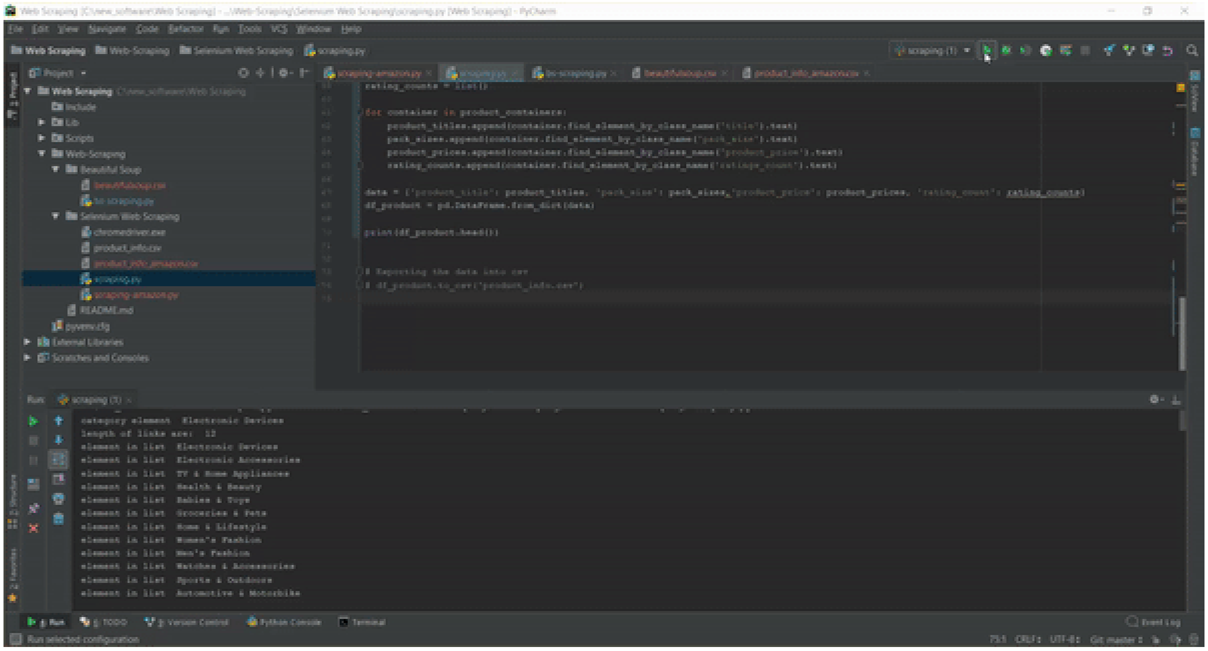
1. Pip install selenium.

3. **Include these import**

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.common.exceptions import TimeoutException
```

4. **Drive Selenium Chrome Browser** by inserting the executable path and url. In my case, I used the relative path to find the chromedriver.exe located in the same directory as my script.

```
driver = webdriver.Chrome(executable_path='chromedriver')
driver.get('https://www.lazada.sg/#')
```



```
    rating_counts = list()

    for container in product_containers:
        product_titles.append(container.find_element_by_class_name('title').text)
        pack_sizes.append(container.find_element_by_class_name('pack_size').text)
        product_prices.append(container.find_element_by_class_name('product_price').text)
        rating_counts.append(container.find_element_by_class_name('ratings_count').text)

    data = {'product_title': product_titles, 'pack_size': pack_sizes,'product_price': product_prices, 'rating_count': rating_counts}
    df_product = pd.DataFrame.from_dict(data)

    print(df_product.head())


    # Exporting the data into csv
    # df_product.to_csv('product_info.csv')
```

```
category element  Electronic Devices
length of links are:  12
element in list  Electronic Devices
element in list  Electronic Accessories
element in list  TV & Home Appliances
element in list  Health & Beauty
element in list  Babies & Toys
element in list  Groceries & Pets
element in list  Home & Lifestyle
element in list  Women's Fashion
element in list  Men's Fashion
element in list  Watches & Accessories
element in list  Sports & Outdoors
element in list  Automotive & Motorbike
```

Selenium Running Chrome and Extract Lazada and Redmart Data

5.**Wait page to load and find the element**. This is how Selenium could be different from

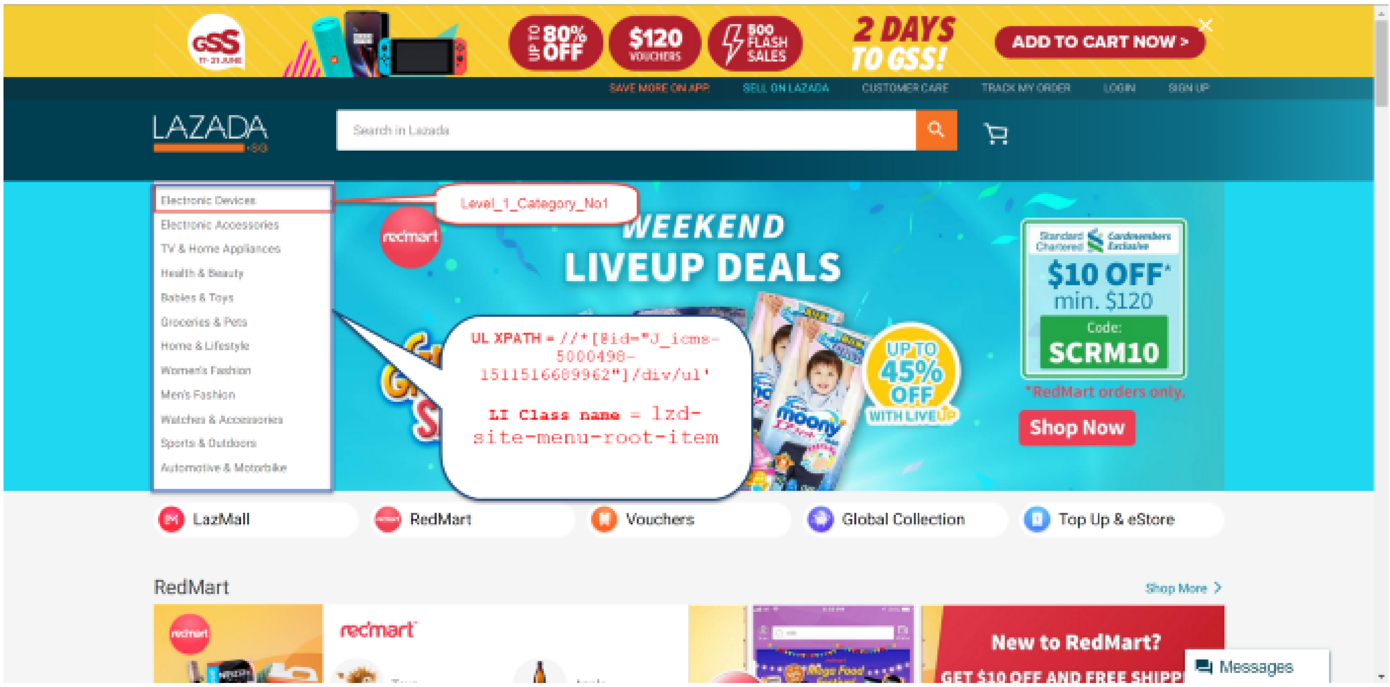
You can stop the wait until Expected Conditions (EC) is met to find by ID *“Level_1_Category_No1”*. If 30 seconds already passed without finding such element, then pass *TimeoutException* to shut the browser.

```
timeout = 30
try:
    WebDriverWait(driver,
timeout).until(EC.visibility_of_element_located((By.ID,
"Level_1_Category_No1")))
except TimeoutException:
    driver.quit()
```

Congrats. We have setup Selenium to use our Chrome Browser. Now we are ready to automate the Information Extraction.

Information Extraction

Let us identify several attributes from our Lazada Websites and extract their DOM Elements.



Level_1_Category_No1

UL XPATH = //*[@id="J_icms-5000498-1511516609962"]/div/ul'

LI Class name = lzd-site-menu-root-item

Extracting the DOM Elements via ID, Class, and XPATH Attributes

```
category_element =
driver.find_element(By.ID,'Level_1_Category_No1').text;
#result -- Electronic Devices as the first category listing
```

2. **Get the unordered list xpath** (ul) and extract the values for each list item (li). You could inspect the element, right click, and select copy>XPATH to easily generate the relevant XPATH. Feel free to open the following link for further detail.

How to Locate Elements in Chrome and IE Browsers for Building Selenium Scripts - Selenium Tutorial...

This is tutorial #7 in our Selenium Online Training Series. If you want to check all Selenium tutorials in this series...

www.softwaretestinghelp.com

```
list_category_elements = driver.find_element(By.XPATH,'//*
[@id="J_icms-5000498-1511516689962"]/div/ul')
links = list_category_elements.find_elements(By.CLASS_NAME,"lzd-site-
menu-root-item")
for i in range(len(links)):
    print("element in list ",links[i].text)
#result {Electronic Devices, Electronic Accessories, etc}
```

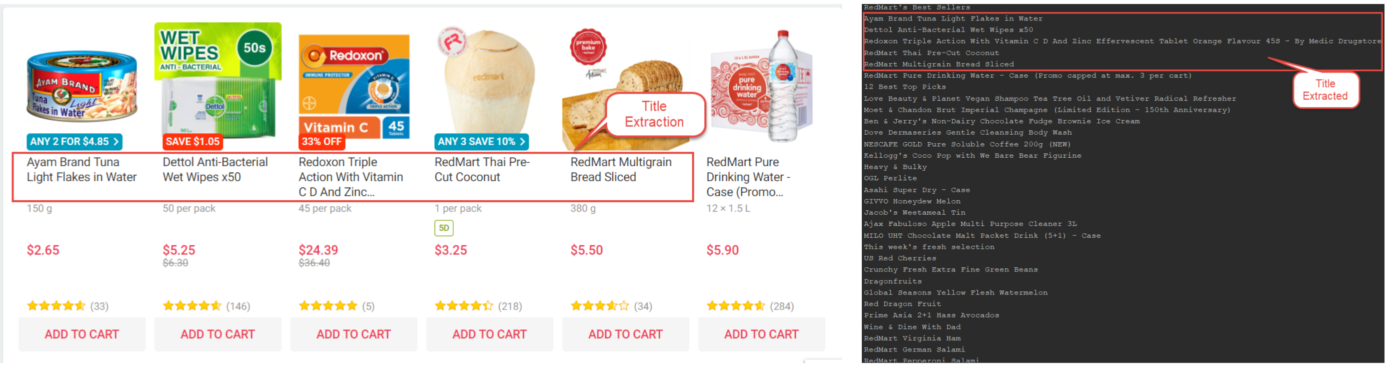
Clicks and Actions

1. **Automate Actions**. Supposedly you want to browse to Redmart from Lazada Homepage, you can mimic the click in the *ActionChains Object*.

```
element = driver.find_elements_by_class_name('J_ChannelsLink')[1]
webdriver.ActionChains(driver).move_to_element(element).click(element)
.perform()
```

Extracting all product listings from Redmart

```
product_titles = driver.find_elements_by_class_name('title')
for title in product_titles:
    print(title.text)
```



Title Extraction

```
RedMart's Best Sellers
Ayam Brand Tuna Light Flakes in Water
Dettol Anti-Bacterial Wet Wipes x50
Redoxon Triple Action With Vitamin C D And Zinc Effervescent Tablet Orange Flavour 45S - By Medic Drugstore
RedMart Thai Pre-Cut Coconut
RedMart Multigrain Bread Sliced
RedMart Pure Drinking Water - Case (Promo capped at max. 3 per cart)
12 Best Top Picks
Love Beauty & Planet Vegan Shampoo Tea Tree Oil and Vetiver Radical Refresher
Moet & Chandon Brut Imperial Champagne (Limited Edition - 150th Anniversary)
Ben & Jerry's Non-Dairy Chocolate Fudge Brownie Ice Cream
Dove Dermaseries Gentle Cleansing Body Wash
NESCAFE GOLD Pure Soluble Coffee 200g (NEW)
Kellogg's Coco Pop with We Bare Bear Figurine
Heavy & Bulky
OGL Perlite
Asahi Super Dry - Case
GIVVO Honeydew Melon
Jacob's Weetameal Tin
Ajax Fabuloso Apple Multi Purpose Cleaner 3L
MILO UHT Chocolate Malt Packet Drink (5+1) - Case
This week's fresh selection
US Red Cherries
Crunchy Fresh Extra Fine Green Beans
Dragonfruits
Global Seasons Yellow Flesh Watermelon
Red Dragon Fruit
Prime Asia 2+1 Hass Avocados
Wine & Dine With Dad
RedMart Virginia Ham
RedMart German Salami
RedMart Pepperoni Salami
```

Title Extracted

Redmart Best Seller Title Extractions

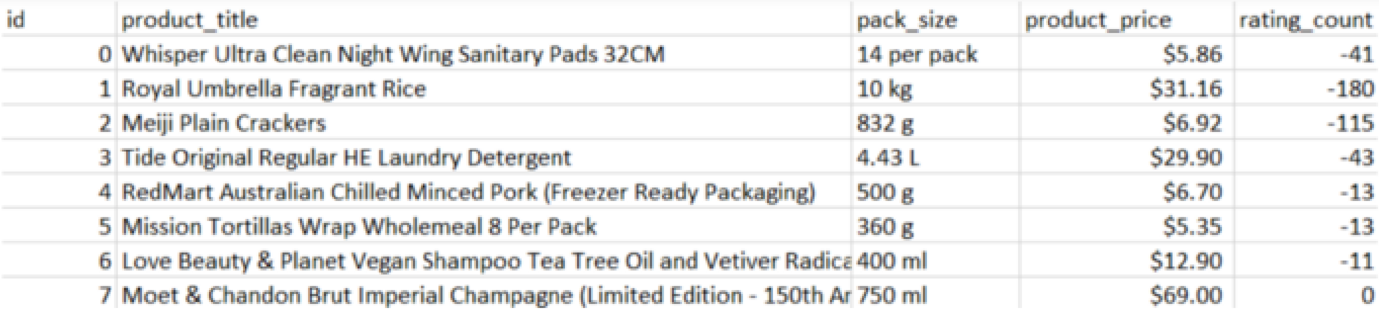
2. **Extract the product title, pack size, price, and rating**. We will open several lists to contain every item and dump them into a Dataframe.

```
product_containers = 
driver.find_elements_by_class_name('product_container')

for container in product_containers:
product_titles.append(container.find_element_by_class_name('title').te
xt)
pack_sizes.append(container.find_element_by_class_name('pack_size').te
xt)
product_prices.append(container.find_element_by_class_name('product_pr
ice').text)
rating_counts.append(container.find_element_by_class_name('ratings_cou
nt').text)

data = {'product_title': product_titles, 'pack_size': 
pack_sizes,'product_price': product_prices, 'rating_count': 
rating_counts}
```

3. **Dump the information** into a Pandas Dataframe and csv



id	product_title	pack_size	product_price	rating_count
0	Whisper Ultra Clean Night Wing Sanitary Pads 32CM	14 per pack	$5.86	-41
1	Royal Umbrella Fragrant Rice	10 kg	$31.16	-180
2	Meiji Plain Crackers	832 g	$6.92	-115
3	Tide Original Regular HE Laundry Detergent	4.43 L	$29.90	-43
4	RedMart Australian Chilled Minced Pork (Freezer Ready Packaging)	500 g	$6.70	-13
5	Mission Tortillas Wrap Wholemeal 8 Per Pack	360 g	$5.35	-13
6	Love Beauty & Planet Vegan Shampoo Tea Tree Oil and Vetiver Radica	400 ml	$12.90	-11
7	Moet & Chandon Brut Imperial Champagne (Limited Edition - 150th Ar	750 ml	$69.00	0

CSV Dump for each of the product in Best Seller Redmart

> Congrats! You have effectively expanded your skills to extract any information found online!

Purpose, Github Code and Your Contributions

The purpose for this Proof Of Concepts (POC) was created as a part of my own side project. The goal of this application is to use web scraping tool to extract any publicly available information without much cost and manpower.

In this POC, I used Python as the scripting language, *Beautiful Soup and Selenium library* to extract the necessary information.

The Github Python Code is located below.

VincentTatan/Web-Scraping

Web Scraping with Beautiful Soup and Selenium. Contribute to VincentTatan/Web-Scraping development by creating an...

github.com

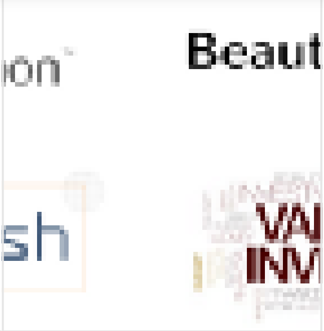
Feel free to clone the repository and contribute whenever you have time.

Beautiful Soup and Stocks Investing

Value Investing Dashboard with Python Beautiful Soup and Dash Python

An Overview of Web Scraping with a Quick Dash Visualization for Value Investing

towardsdatascience.com



Hopefully from this relevant publication, you could learn how to scrape critical information and develop an useful application. Please read and reach out to me if you like it.

Finally…

Whew… That's it, about my idea which I formulated into writings. I really hope this has been a great read for you guys. With that, I hope my idea could be a source of inspiration for you to develop and innovate.

Please **Comment** out below to suggest and feedback.

Happy coding :)

About the Author

Vincent Tatan is a Data and Technology enthusiast with relevant working experiences from Visa Inc. and Lazada to implement microservice architectures, data engineering, and analytics pipeline projects.

Vincent is a native Indonesian with a record of accomplishments in problem solving with strengths in Full Stack Development, Data Analytics, and Strategic Planning.

He has been actively consulting SMU BI & Analytics Club, guiding aspiring data scientists and engineers from various backgrounds, and opening up his expertise for businesses to develop their products .

Please reach out to Vincent via **LinkedIn , Medium or Youtube Channel**

This disclaimer informs readers that the views, thoughts, and opinions expressed in the text belong solely to the author, and not necessarily to the author's employer, organization, committee or other group or individual. References are picked up from the list and any similarities with other works are purely coincidental

This article was made purely as the author's side project and in no way driven by any other hidden agenda.