Scraping Top Repositories for Topics on GitHub

Project Outline:

- We are going to scrape https://github.com/topics (https://github.com/topics)
- We will get a list of topics and for each topic we will get topic title,topic page URL and topic description
- For each topic we will get the top repositories
- And for each repository we will grab the repo name, username, stars and repo url
- And for each topic separate csv would be created.

Tools used - Python, requests, BeautifulSoup, Pandas

1. Scraping list of topics on GitHub

- Used Requests to download the page
- Used BS4 to parse and extract information
- Convert it to a Pandas Dataframe

In [1]:

```
import requests
import pandas as pd
from bs4 import BeautifulSoup
```

1.1 Using Requests to download Web Page

In [3]:

```
topic_url = "https://github.com/topics"
response = requests.get(topic_url)

# To Check if Web page downloaded succesfully. If output is 200 then we got it right. Reminder: Please keep your Internet On while running
response.status_code
```

Out[3]:

```
200
```

1.2 Use BS4 to parse and extract information

In [4]:

```
doc = BeautifulSoup(response.text,'html.parser')

# To check if web page got successfully parsed. if output bs4.BeautifulSoup than we got it right
type(doc)
```

Out[4]:

```
bs4.BeautifulSoup
```

In [5]:

```
# Get List of topic TITLES tags
sel_class = "f3 lh-condensed mb-0 mt-1 Link--primary"
topic_title_tags=doc.find_all('p',class_ =sel_class)
len(topic_title_tags)

# Get List of topic TITLES
topic_titles = []
for tag in topic_title_tags:
    topic_titles.append(tag.text)
print(topic_titles)

# Get list of Topic DESCRIPTIONS tags
topic_des = 'f5 color-fg-muted mb-0 mt-1'
topic_desc_tags = doc.find_all('p',class_ = topic_des)
topic_desc_tags

# Get list of Topic DESCRIPTIONS
topic_desc= []
for tag in topic_desc_tags:
    topic_desc.append(tag.text.strip())
print(topic_desc)

# Get list of topic URL tags
link_class = 'no-underline flex-1 d-flex flex-column'
topic_link_tag = doc.find_all('a',class_=link_class)
topic_link_tag[0]['href']

# Get list of topic URL
topic_link = []
base_url = 'https://github.com'
for tag in topic_link_tag:
    topic_link.append(base_url+tag['href'])
print(topic_link)
```

```
['3D', 'Ajax', 'Algorithm', 'Amp', 'Android', 'Angular', 'Ansible', 'API', 'Arduino', 'ASP.NET', 'Atom', 'Awesome Lists',
'Amazon Web Services', 'Azure', 'Babel', 'Bash', 'Bitcoin', 'Bootstrap', 'Bot', 'C', 'Chrome', 'Chrome extension', 'Command
line interface', 'Clojure', 'Code quality', 'Code review', 'Compiler', 'Continuous integration', 'COVID-19', 'C++']
['3D refers to the use of three-dimensional graphics, modeling, and animation in various industries.', 'Ajax is a technique
for creating interactive web applications.', 'Algorithms are self-contained sequences that carry out a variety of tasks.',
'Amp is a non-blocking concurrency library for PHP.', 'Android is an operating system built by Google designed for mobile d
evices.', 'Angular is an open source web application platform.', 'Ansible is a simple and powerful automation engine.', 'An
API (Application Programming Interface) is a collection of protocols and subroutines for building software.', 'Arduino is a
n open source platform for building electronic devices.', 'ASP.NET is a web framework for building modern web apps and serv
ices.', 'Atom is a open source text editor built with web technologies.', 'An awesome list is a list of awesome things cura
ted by the community.', 'Amazon Web Services provides on-demand cloud computing platforms on a subscription basis.', 'Azure
is a cloud computing service created by Microsoft.', 'Babel is a compiler for writing next generation JavaScript, today.',
'Bash is a shell and command language interpreter for the GNU operating system.', 'Bitcoin is a cryptocurrency developed by
Satoshi Nakamoto.', 'Bootstrap is an HTML, CSS, and JavaScript framework.', 'A bot is an application that runs automated ta
sks over the Internet.', 'C is a general purpose programming language that first appeared in 1972.', 'Chrome is a web brows
er from the tech company Google.', 'Chrome extensions enable users to customize the Chrome browsing experience.', 'A CLI, o
r command-line interface, is a console that helps users issue commands to a program.', 'Clojure is a dynamic, general-purpo
se programming language.', 'Automate your code review with style, quality, security, and test-coverage checks when you need
them.', 'Ensure your code meets quality standards and ship with confidence.', 'Compilers are software that translate higher
-level programming languages to lower-level languages (e.g. machine code).', 'Automatically build and test your code as you
push it upstream, preventing bugs from being deployed to production.', 'The coronavirus disease 2019 (COVID-19) is an infec
tious disease caused by SARS-CoV-2.', 'C++ is a general purpose and object-oriented programming language.']
['https://github.com/topics/3d', 'https://github.com/topics/ajax', 'https://github.com/topics/algorithm', 'https://github.c
om/topics/amphp', 'https://github.com/topics/android', 'https://github.com/topics/angular', 'https://github.com/topics/ansi
ble', 'https://github.com/topics/api', 'https://github.com/topics/arduino', 'https://github.com/topics/aspnet', 'https://gi
thub.com/topics/atom', 'https://github.com/topics/awesome', 'https://github.com/topics/aws', 'https://github.com/topics/azu
re', 'https://github.com/topics/babel', 'https://github.com/topics/bash', 'https://github.com/topics/bitcoin', 'https://git
hub.com/topics/bootstrap', 'https://github.com/topics/bot', 'https://github.com/topics/c', 'https://github.com/topics/chrom
e', 'https://github.com/topics/chrome-extension', 'https://github.com/topics/cli', 'https://github.com/topics/clojure', 'ht
tps://github.com/topics/code-quality', 'https://github.com/topics/code-review', 'https://github.com/topics/compiler', 'http
s://github.com/topics/continuous-integration', 'https://github.com/topics/covid-19', 'https://github.com/topics/cpp']
```

1.3 Create a DataFrame and convert it into CSV

In [6]:

```
# Create a dictionary with already estalished list and give them titles
topics_dict={
    'title': topic_titles,
    'description': topic_desc,
    'url':topic_link
}

# Convert dictionary into DataFrame
topics_df = pd.DataFrame(topics_dict)
topics_df.to_csv('topics.csv',index=None)
```

2. Get top Repos from each topic

In [8]:

```
# Define function to Parse Star Counts

def parse_star_count(star_str):
    star_str = star_str.strip()
    if star_str[-1] == "k":
        return int(float(star_str[:-1])*1000)
    return int(star_str)
```

In [9]:

```
# Define info to parse all repo information

def get_repo_info(h1_tag,star_tag):
    a_tags = h1_tag.find_all('a')
    username = a_tags[0].text.strip()
    repo_name = a_tags[1].text.strip()
    rep_url = base_url+a_tags[1]['href']
    star_count = parse_star_count(star_tag.text)
    return username,repo_name,star_count,rep_url
```

In [11]:

```
def get_topic_repos(topic_url):
    # to download web page
    response = requests.get(topic_url)
    if response.status_code != 200:
        raise Exception("Failed to load page".format(topic_url))

    # To parse each topic url
    topic_doc = BeautifulSoup(response.text,'html.parser')

    # To parse repo name and username
    repo_tags = topic_doc.find_all('h3',class_ = 'f3 color-fg-muted text-normal lh-condensed')

    # parsing no of stars for the repo
    star_tags = topic_doc.find_all('span',class_='Counter js-social-count')
    topic_repos_dict = {
        "username" : [],
        "repo_name" : [],
        "stars" : [],
        "repo_url" : []
    }

    # Extracting username, repo name, no. of stars and repo url of top repositories for each topic
    for i in range(len(repo_tags)):
        repo_info = get_repo_info(repo_tags[i],star_tags[i])
        topic_repos_dict['username'].append(repo_info[0])
        topic_repos_dict['repo_name'].append(repo_info[1])
        topic_repos_dict['stars'].append(repo_info[2])
        topic_repos_dict['repo_url'].append(repo_info[3])
    return pd.DataFrame(topic_repos_dict)
```

In [14]:

```
# Example - Data Frame for 3D topic

get_topic_repos(topic_link[0]).head()
```

Out[14]:

	username	repo_name	stars	repo_url
0	mrdoob	three.js	92300	https://github.com/mrdoob/three.js
1	pmndrs	react-three-fiber	22700	https://github.com/pmndrs/react-three-fiber
2	libgdx	libgdx	21600	https://github.com/libgdx/libgdx
3	BabylonJS	Babylon.js	20800	https://github.com/BabylonJS/Babylon.js
4	ssloy	tinyrenderer	17100	https://github.com/ssloy/tinyrenderer

3. To Create separate CSV files for each Topic

In [ ]:

```
for i in topic_link:
    # as topic_link variable has list of all topic urls

    df=get_topic_repos(i)

    # 'https://github.com/topics/algorithm' after 26 characters we get the topic title and adding .csv to recognize which topic's repos ar
    top = i[26:] + ".csv"

    # adding all csv file to particular folder and copying its path
    df.to_csv(path_or_buf = "C:/myproject/lms/Scripts/GitHub Data/"+top,index=None)
```