

Scraping News Articles using Python

In Every direction(North East West South), Coming from the days where News used to arrive to us through one hour program in the evening, the news data that arrives to us now is quite overwhelming. Trying to be a rational decision maker, it is wise to be up to date with the current world scenario.



Keeping that in mind, in the current web scrapping project, I would like to gather the data on Economics from one website, namely `indianexpress.com`. Later on the project will be accentuated to look into other websites as well and accumulate as much information on some particular topics as well and work on the data for further processing.

What is scrapping you ask? Since the websites have so much data coming in from all around the places. The main frame of the website is the same but the data it shows change over time. Web scrapping is a way to collect the required data for some personal and professional use.

In this project, python programming language is used. There are several inbuilt and community built libraries which can be used to download a web page in the form of text and use that to extract the required data. We will be using Requests and BeautifulSoup libraries for the above.

About `indianexpress.com`



The Indian Express is an English-language Indian daily newspaper founded in 1932. It is published in Mumbai by the Indian Express Group. In 1999, eight years after the group's founder Ramnath Goenka's death in 1991, the group was split between the family members. The southern editions took the name The New Indian Express, while the northern editions, based in Mumbai, retained the original Indian Express name.

Project Idea



In this project, we will parse through the `indianexpress` news of economics page to get details about the news from around the world. Here in this project, `indianexpress.com` has been used for downloading the data for first 50 pages on the topic of Economics. We will retrieve information from the page "Economics" using web scraping: a process of extracting information from a website programmatically. Web scraping isn't magic, and yet some readers may grab information on a daily basis. For example, a recent graduate may copy and paste information about companies they applied for into a spreadsheet for job application management.

Project Steps

Steps taken:

1. Find the potential news websites to scrape.
2. Download the desired webpage using Requests library.
3. Find all the tags containing the necessary information from the webpage.
4. Create list of dictionaries of the information.
5. Write the information into the csv file.
6. Repeat the steps for 50 pages

Project Goal

The project goal is to build a web scraper that withdraws all desirable information and assemble them into a single CSV. By the end of this project a csv file will be created with Date, Title, Brief and news link followed be the corresponding data for the next 734 enteries.

	Date	Title	Brief	Link
730	November 14, 2020 2:06:49 pm	Money sent abroad under LRS touches pre-Covid ...	Remittances for the purposes of maintaining re...	https://indianexpress.com/article/business/eco...
731	November 14, 2020 12:51:33 am	After a positive September, exports slip in Oc...	Exports in October were recorded at \$24.89 bil...	https://indianexpress.com/article/business/eco...
732	November 13, 2020 6:30:10 pm	Industry bodies, experts to submit ideas on Bu...	Owing to the pandemic, the ministry decided to...	https://indianexpress.com/article/business/eco...
733	November 13, 2020 11:02:36 am	Diwali: Festive season shopping shows Indian e...	Data compiled by Bloomberg showed an increase ...	https://indianexpress.com/article/business/eco...
734	November 13, 2020 4:20:24 am	Sitharaman unveils Atmanirbhar 3.0: Incentives...	Fresh stimulus worth Rs 1.2 lakh cr; Rs 10,000...	https://indianexpress.com/article/business/eco...

How to run the code.

The current project is hosted on [Jovian]. You can execute the notebook by going to the THIS link and click on [run] button and select [run on binder] option there. Make sure to clear all the output and run all the cells one after another.

Installing and importing the Libraries required for the current project.

Before starting the project importing some of the dependencies is a good idea. The current project uses the following libraries:

1. Requests - to download the the web page in the text format.
2. BeautifulSoup - to be able to use the text downloaded from the Requests library to be used for further processing.
3. Pandas - Pandas is used to analyze data.
4. Jovian - [Jovian.ai](https://jovian.ai) provides a neat way to save the versions of the project. It is useful if one is working either online or offline given a case where machine crashes.

Lets Begin



Note : We will use the 'Jovian library' and its 'commit()' function throughout the code to save our progress as we move along.

Use the "Run" button to execute the code.

```
# Install the library
!pip install jovian --upgrade --quiet
# Import the library
import jovian
```

```
!pip install requests --upgrade --quiet
import requests
```

```
!pip install beautifulsoup4 --upgrade --quiet
from bs4 import BeautifulSoup
```

```
!pip install pandas --upgrade --quiet
import pandas as pd
```

```
# Execute this to save new versions of the notebook
jovian.commit(project="project-economy")
```

```
[jovian] Updating notebook "abhineets17/project-economy" on https://jovian.ai
[jovian] Committed successfully! https://jovian.ai/abhineets17/project-economy
'https://jovian.ai/abhineets17/project-economy'
```

Making a list of URLs for the pages to be scrapped off.

We use the base url for the page on that url we run a `for` loop to iterate over and create a list of 50 pages.

```
base_url = 'https://indianexpress.com/section/business/economy/page/'

def url_building(url):
    urls = []
    for i in range(50):
        urls.append(base_url + str(i + 1))
    return urls

url_building(base_url)[:5]
```

```
['https://indianexpress.com/section/business/economy/page/1',
 'https://indianexpress.com/section/business/economy/page/2',
 'https://indianexpress.com/section/business/economy/page/3',
 'https://indianexpress.com/section/business/economy/page/4',
 'https://indianexpress.com/section/business/economy/page/5']
```

First page image



Services activity hits 11-year high in June despite mounting cost pressures

JULY 5, 2022 2:48:15 PM

The seasonally adjusted S&P Global India Services PMI Business Activity Index rose from 58.9 in May to 59.2 in June – its highest mark since April 2011.



Government's windfall tax seen netting \$12.7 billion, cut budget gap

JULY 4, 2022 1:24:52 PM

The taxes on production of crude, and exports of petrol, diesel and aviation fuel could garner about 1 trillion rupees (\$12.7 billion) for the government if the levies continue for the rest of the fiscal year ending in March, Mumbai based Kotak Mahindra Bank Ltd. economists

Suvodeep Rakshit, Upasna Bhardwaj and Anurag Balajee wrote in their report.



See pvt capex in 2022-27 growing at twice the rate of 2016-2022 due to PLI: MD & CEO, CRISIL Ltd

JULY 4, 2022 10:44:30 AM

The persistence of US inflation has been a surprise. In just six months since December, they have halved the US GDP growth projections for 2022. The Fed has signalled it will

Getting the response from the particular URL using Requests.

`requests.get` function from the requests library is used for downloading the webpage which is in the response object.

In order to check if we can receive the webpage as required for further processing we use `status_code` property of response object.

```
response = requests.get(url_building(base_url)[2])
print (response.status_code)
```

200

In the above code, the `requests.get` downloads the webpage. The `status_code` of 200 shows that we have been able to get to the webpage without any problems.

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

In the above section we have passed the downloaded page in the `BeautifulSoup` object with 'html.parser' argument.

```
# printing the first 1000 lines of the downloaded webpage.
print(response.text[:1000])
```

<!DOCTYPE html>

```

<html lang="en" xmlns:og="http://ogp.me/ns#">
<head>
<meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,minimum-scale=1,initial-
scale=1">
    <!-- HTML5 shim and Respond.js IE8 support of HTML5 elements and media queries
-->
<!--[if lt IE 9]>
    <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>
    <script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js">
</script>
<![endif]-->
    <title>Economy News - Latest Indian Economy, GDP, Growth Rate, GST News,
Govt Policy News, Infra News, Budget 2022-23 News Updates, International, Trade News |
Page 3, Economy | The Indian Express</title><meta name="description" content="Economy
News, Page 3 - Find detailed coverage on Economy at IndianExpress.com." /><meta
name="keywords" content="Economy News, India GDP rate, Global Economy News, Latest
Economy News, Indian economy growth, Manufacturing news, Tax News, Budget 2022 News,
Economic News, Latest Economic

```

```
len(response_text.text)
```

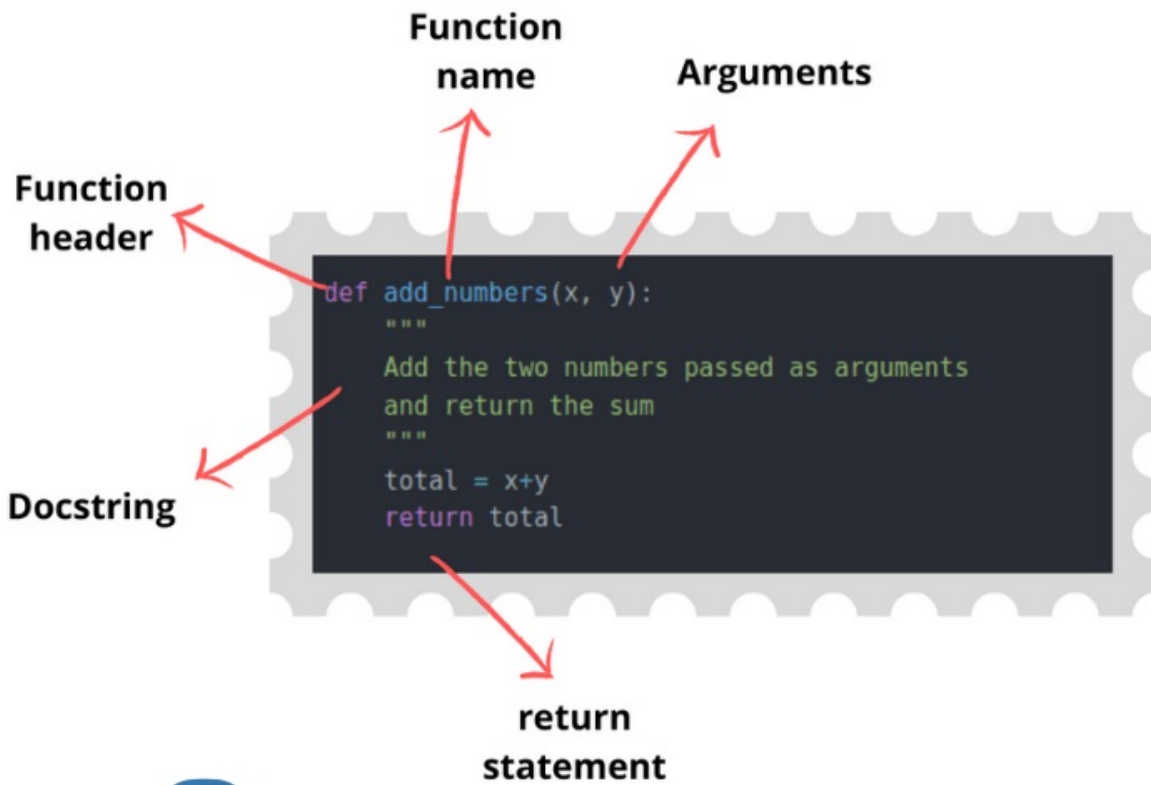
14156

As we can see there are 14943 characters of code in the above downloaded page content.

Creating the reusable funtion for the requests.get and returning a BeautifulSoup object.

Python def

Python `def` keyword is used to define a function, it is placed before a function name that is provided by the user to create a user-defined function.



Python Functions

```
def getting_response(url):  
    '''Getting the response from the url,  
       raise exception if required response is not recieved and passing the response to E  
  
    # requesting the links and returning the response  
    response = requests.get(url)  
  
    # raising an exception if the page does not respond with the status code - 200.  
    if response.status_code != 200:  
        print('Status code:', response.status_code)  
        raise Exception('Failed to fetch web page ' + topic_repos_url)  
  
    # passing the response.text through BeautifulSoup to retuen it.  
    return BeautifulSoup(response.text, 'html.parser')  
  
Response = getting_response(url_building(base_url)[2])
```

Lets take a look at what a webpage looks like when we inspect the element and how are we going to extract the information that we required.



Services activity hits 11-year high in June despite mounting cost pressures

JULY 5, 2022 2:48:15 PM

The seasonally adjusted S&P Global India Services PMI Business Activity Index rose from 58.9 in May to 59.2 in June – its highest mark since April 2011.



Government's windfall tax seen netting \$12.7 billion, cut budget gap

JULY 4, 2022 1:34:52 PM

The taxes on production of crude and exports of petrol

DevTools - indianexpress.com/section/business/economy/

```

Elements Console Sources Network Performance >> 4
▼ <body class="archive tax-ie_section term-economy term-442186100" style="overflow: initial;">
  <div style="display: block !important; height: 0px;"></div>
  <div id="div-gpt-ad-1527257683510-0"></div>
  ▼ <div id="wrapper">
    ▶ <header id="common">...</header>
    ▶ <div class="add-left skinning-fixed" style="display: block;">...</div>
    ▶ <div class="add-right skinning-fixed" style="display: block;">...</div>
    ▶ <script>...</script>
    ▼ <div id="section">
      ▼ <div class="container">
        ▶ <div class="breaking-news">...</div>
        ▼ <div class="row">

```

As we can see the articles are structured in a list of unstructured list which is represented by the `li` tags . Taking a look into the `li` tags can reveal more info.


BUSINESS

ECONOMY

MARKET

BANKING & FINANCE


COMPANIES



Services activity hits 11-year high in June despite mounting cost pressures

JULY 5, 2022 2:48:15 PM

The seasonally adjusted S&P Global India Services PMI Business Activity Index rose from 58.9 in May to 59.2 in June – its highest mark since April 2011.



Government's windfall tax seen netting \$12.7 billion, cut budget gap

JULY 4, 2022 1:34:52 PM

The taxes on production of crude and exports of petrol

DevTools - indianexpress.com/section/business/economy/

Elements

Console

Sources

Network

Performance

>>

4

<div class="articles">

>

<div class="snaps">...</div>

>

<h2 class="title"> == \$0

>

<a href="https://indianexpress.com/article/business/economy/india-eu-fta-talks-next-round-at-brussels-in-sept-says-commin-8005779/"

India-EU FTA talks: Next round at Brussels in Sept, says ComMin

</h2>

<div class="date"> July 3, 2022 7:42:23 am</div>

>

<p>...</p>

</div>

>

<div class="articles">...</div>

>

<div class="articles">...</div>

All the information that we will be requiring to be scrapped for a particular news articles.

Compile the extracted information in a dictionary.

Creating a function to parse the data from just one `div` and returning dictionary containing all the information.

```
def parse_repo(doc):
    req_tag = doc.find_all('div', class_ = 'articles')[0]    #first news of this page

    Title = req_tag.find('h2').text.strip()
    Brief = req_tag.find('p').text
    Date = req_tag.find('div', class_ = 'date').text
    Link = req_tag.find('a')['href']

    Dict = {'Date': Date, 'Title': Title, 'Brief': Brief, 'Link': Link }
    return Dict

parse_repo(getting_response(url_building(base_url)[2]))
```

```
{'Date': ' June 20, 2022  7:24:48 am',
 'Title': 'Mixed signals in capacity utilisation: Cement, steel up, FMCG, auto lag',
 'Brief': 'Analysts said capacity utilisation of 75-80 per cent needs to be sustained
over 3-4 quarters for it to translate into an expansionary drive by the industry.'
```

Besides, there are mixed signals within sectors too.',
'Link': '<https://indianexpress.com/article/business/economy/mixed-signals-in-capacity-utilisation-cement-steel-up-fmcg-auto-lag-7979084/>'}

As we can see the returned dictionary containing the information about the Date, Time, Brief and Link of a particular news article.

Bringing it all together in one block.

Putting it all together in one place. Along with it a function have been defined which will loop over the article tags on a particular page and then scrape the required data and this will be done over and over again for next 50 pages. Later all the scrapped data will be added to list of dictionaries. The data will then be written in a csv file which can be viewed later.

```
def url_building():  
    '''Creating the list of urls for first 50 pages.'''  
  
    # Initiating a list to store all the urls  
    urls = []  
  
    #  
    base_url = 'https://indianexpress.com/section/business/economy/page/'  
  
    # using a for loops to get first 50 pages  
    for i in range(50):  
        urls.append(base_url + str(i + 1))  
    return urls
```

```
def getting_response(url):  
    '''Getting the response from the url,  
        raise exception if required response is not received and passing the response  
  
    # requesting the links and returning the response  
    response = requests.get(url)  
  
    # raising an exception if the page does not respond with the status code - 200.  
    if response.status_code != 200:  
        print('Status code:', response.status_code)  
        raise Exception('Failed to fetch web page ' + topic_repos_url)  
  
    # passing the response.text through BeautifulSoup to return it.  
    return BeautifulSoup(response.text, 'html.parser')
```

```
def parse_repo(tag):  
    '''Getting the info required and parsing in the dictionary.'''  
  
    # defining the tags and passing them to the dictionaries  
    title = tag.find('h2').text.strip()  
    brief = tag.find('p').text  
    date = tag.find('div', class_ = 'date').text
```

```

link = tag.find('a')['href']
return {'Date': str(date), 'Title': str(title), 'Brief' : str(brief), 'Link' : str

```

```

def scrapping_info(doc):
    '''Scrapping the required info.'''

    # finding the required tags
    matching_tags = doc.find_all('div', class_ = 'articles')[10:]

    # parsing all the tags into dictionaries one by one and saving them into a list
    repos = [parse_repo(tag) for tag in matching_tags]

    # print(repos)
    return repos

```

```

def write_csv(items, path):

    '''Defining the funtion to write down everything in the .csv file format'''

    dataframe = pd.DataFrame(items)
    dataframe.to_csv(path, index = None)

```

```

def Final(path = 'Economy_News_headlines.csv' ):

    '''Bringing it all together by using all the funcions into one place.'''

    # initiating a list to store all the data
    main = []

    # using the first function get the urls of first 50 number of pages
    #url_building()

    # looping over the list from url_building() function,
    # request the pages to be downloaded,
    # passing it through BeautifulSoup and getting the tags needed for further process
    for i in range(1,50):

        # extending the list

        main.extend(scrapping_info(getting_response(url_building()[i])))

    # witing the gained list of dictionaries into a csv file
    write_csv(main, path)
    print(f'News articles of 50 pages from Indian Express on the topic Economics have b
    return path

```

Pushing off the final button!!!

```
Final()
```

News articles of 50 pages from Indian Express on the topic Economics have been written to the file `Economy_News_headlines.csv`

```
'Economy_News_headlines.csv'
```

Lets showcase the downloaded csv file in the dataframe.

```
df = pd.read_csv('Economy_News_headlines.csv')
df.head()
```

	Date	Title	Brief	Link
0	June 28, 2022 5:24:19 pm	47th GST Council meet underway: To discuss rat...	With high inflation rate, any major rejig of t...	https://indianexpress.com/article/business/eco...
1	June 28, 2022 7:05:55 am	NITI Aayog pitches for incentives to draw more...	Based on a survey conducted across urban centr...	https://indianexpress.com/article/business/eco...
2	June 28, 2022 8:18:48 am	Rationalisation, reforms may test Centre-state...	With high inflation rate, any major rejig of t...	https://indianexpress.com/article/business/eco...
3	June 26, 2022 8:31:49 am	Citing fiscal strains, Expenditure Department ...	The government has spent approximately Rs 2.60...	https://indianexpress.com/article/business/eco...
4	June 25, 2022 3:05:52 am	'FY22 MSME loan growth 36% above pre-Covid level'	According to CRIF High Mark, a credit informat...	https://indianexpress.com/article/business/eco...

```
df.tail()
```

	Date	Title	Brief	Link
730	November 23, 2020 5:21:57 pm	Momentum of economic reforms will continue, FM...	"The momentum for reform shall continue. Sever...	https://indianexpress.com/article/business/eco...
731	November 23, 2020 3:41:25 pm	India likely to have current account surplus t...	Noting that COVID crisis is a crisis to demand...	https://indianexpress.com/article/business/eco...
732	November 23, 2020 4:54:59 am	LVB merger: Investors may seek legal recourse,...	There are some who also raised questions on th...	https://indianexpress.com/article/business/eco...
733	November 23, 2020 4:49:23 am	After slight fall, MSME dues of CPSEs & minist...	As per government data on MSME Samadhaan porta...	https://indianexpress.com/article/business/eco...
734	November 22, 2020 12:25:24 am	In eight months, forex reserves rise by more t...	This jump of 22 per cent has come following a ...	https://indianexpress.com/article/business/eco...

```
jovian.commit()
```

[jovian] Updating notebook "abhineets17/project-economy" on <https://jovian.ai>

[jovian] Committed successfully! <https://jovian.ai/abhineets17/project-economy>

'<https://jovian.ai/abhineets17/project-economy>'

Summary

In the current project the News website namely `indianexpress.com` has been used for scrapping the articles on the 50 pages. Python as programming language and `requests` and `BeautifulSoup` libraries have been used to download the pages and then exploring and getting the relevant data from the website. Further the work has been in a `csv` file to be used for further processing.

Future works

The current web scrapping project is a starting point for a project. The updates of which will be posted afterwards. In this project I started to look into some of the websites where I could get the data for news articles around the specific topic of 'Economics' .

Later the project will take a flight towards getting the information for user defined topics and scrape of the data accordingly from selective websites - at first the topics in Economics and later in other areas as well.

Another area to look into is how a particular topic is covered in the media and what is the general sense of the authors and which area of the spectrum of academia is the topic is tipping towards.

References

- [1] Python official documentation. <https://docs.python.org/3/>
- [2] Requests library. <https://pypi.org/project/requests/>
- [3] BeautifulSoup documentation. <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>
- [4] Aakash N S, Introduction to Web Scrapping. <https://jovian.ai/aakashns/python-web-scrapping-and-rest-api>
- [5] Pandas library documentation. <https://pandas.pydata.org/docs/>
- [6] Web Scrapping Article. <https://www.toptal.com/python/web-scrapping-with-python>
- [7] Working with Jupyter Notebook <https://towardsdatascience.com/write-markdown-latex-in-the-jupyter-notebook-10985edb91fd>

```
jovian.commit(files = ['Economy_News_headlines.csv'])
```

[jovian] Updating notebook "abhineets17/project-economy" on <https://jovian.ai>

[jovian] Uploading additional files...

[jovian] Committed successfully! <https://jovian.ai/abhineets17/project-economy>

'<https://jovian.ai/abhineets17/project-economy>'

```
jovian.commit(outputs = ["Economy_News_headlines.csv"])
```