Yes, You Can Easily Scrape Websites with Pandas. Here's How.

Source: https://scribe.rip/geekculture/yes-you-can-easily-scrape-websites-with-pandas-heres-how-f833157781d5

Scraping websites with pandas and Python with only a few lines of code.

✍Image via Shutterstock under license to Frank Andrade

Scraping websites doesn't have to be hard (especially if you know Python).

Dynamic websites can be scraped with libraries such as Selenium and Scrapy. Simple websites can be scraped with BeautifulSoup, and super simple websites can be scraped with only pandas.

And we only need one or two lines of code to scrape websites with pandas!

In this article, we're going to scrape data from Wikipedia.

We'll extract the group tables from the 2022 FIFA World Cup → https://en.wikipedia.org/wiki/2022_FIFA_World_Cup. There are 8 tables from Group A to Group H and we'll get them with a few lines of code using pandas and Python.

Group A

Main article: 2022 FIFA World Cup Group A

Pos	Team [v·t·e]	Pld	W	D	L	GF	GA	GD	Pts	Qualification
1	Qatar (H)	0	0	0	0	0	0	0	**0**	Advance to knockout stage
2	Ecuador	0	0	0	0	0	0	0	**0**	
3	Senegal	0	0	0	0	0	0	0	**0**	
4	Netherlands	0	0	0	0	0	0	0	**0**	

First things first — Installing the libraries and dependencies

The first thing we have to do is install the libraries we'll use for this tutorial — pandas and string.

`pandas` will be used to extract data and the `string` module will help us better organize the data extracted.

```
pip install pandas
pip install strings
```

Note: To do web scraìng with pandas we also need to install some dependencies such as `lxml` and `html5lib` (we can install them with pip)

Once these libraries are installed on our computers, we can start with this tutorial.

Scraping the website (in one line of code)

Simple websites like Wikipedia can be easily scraped in one/two lines of code using pandas.

To do so, first, we have to import pandas. Then, we have to use the method `.read_html` and within parentheses write the website we want to scrape.

```
import pandas as pd
all_tables =
pd.read_html("https://en.wikipedia.org/wiki/2022_FIFA_World_Cup")
```

That's pretty much it! All the tables on the Wikipedia website are now stored in the list `all_tables`.

Now we have to look for the tables that belong to groups A, B, …H (8 tables in total)

If we navigate through the elements of the list, we'll see that the first, second and third tables are in index 11, 18, and 25 respectively.

```
all_tables[11]
all_tables[18]
all_tables[25]
```

Here's how the table of Group C (index 25) looks.

	Pos	Teamvte	Pld	W	D	L	GF	GA	GD	Pts	Qualification
0	1	Argentina	0	0	0	0	0	0	0	0	Advance to knockout stage
1	2	Saudi Arabia	0	0	0	0	0	0	0	0	Advance to knockout stage
2	3	Mexico	0	0	0	0	0	0	0	0	NaN
3	4	Poland	0	0	0	0	0	0	0	0	NaN

Organizing the data

If we navigate through the indexes of the `all_tables` list, we'll find that the first table is index 11 and the following tables are 7 indexes ahead.

We can link these indexes with the name of each group using the `zip` function.

```
for letter, i in zip(alphabet, range(11, 67, 7)):
    print(letter, i)
```

The output will be the following

```
A 11
B 18
C 25
D 32
E 39
F 46
G 53
H 60
```

Great! Now we know that index 11 belongs to Group A and index 60 belong to Group H.

It's time to better organize the tables extracted in a dictionary, so we don't have to deal with these ugly indexes anymore. We'll also clean the dataframes by renaming the name of the second column "Teamvte" and dropping the column "Qualification."

```
dict_tables = {}
for letter, i in zip(alphabet, range(11, 67, 7)):
    df = all_tables[i]
    df.rename(columns={df.columns[1]: 'Team'},
inplace=True)
    df.pop('Qualification')
    dict_tables[f'Group {letter}'] = df
```

That's it! Now we have all the tables stored in the `dict_tables` dictionary. Let's have a look

```
>>> dict_tables.keys()
dict_keys(['Group A', 'Group B', 'Group C', 'Group D',
'Group E', 'Group F', 'Group G', 'Group H'])
```

Now we can get the table of any group by specifying its key. Here's how we'll do it for Group H.

```
dict_tables['Group H']
```

And here's the output.

	Pos	Team	Pld	W	D	L	GF	GA	GD	Pts
0	1	Portugal	0	0	0	0	0	0	0	0
1	2	Ghana	0	0	0	0	0	0	0	0
2	3	Uruguay	0	0	0	0	0	0	0	0
3	4	South Korea	0	0	0	0	0	0	0	0

Congratulations! You've learned how to scrape websites with pandas. Here's all the code we've written in this tutorial.

`pandas_ws.py` → https://gist.github.com/ifrankandrade/f0e02d757068dd9cf4aba24b0fd1ab6c#file-pandas_ws-py

```
import pandas as pd
from string import ascii_uppercase as alphabet

all_tables =
pd.read_html("https://en.wikipedia.org/wiki/2022_FIFA_Worl
d_Cup")

dict_tables = {}
for letter, i in zip(alphabet, range(11, 67, 7)):
    df = all_tables[i]
    df.rename(columns={df.columns[1]: 'Team'},
inplace=True)
    df.pop('Qualification')
    dict_tables[f'Group {letter}'] = df

# show all the keys
print(dict_tables.keys())

# show table of Group H
dict_tables['Group H']
```

Turn websites into datasets! **Get my FREE Web Scraping Cheat Sheet by joining my email list with 10k+ people. → https://frankandrade.ck.page/ca38420833**

If you enjoy reading stories like these and want to support me as a writer, consider signing up to become a Medium member. It's $5 a month, giving you unlimited access to thousands of Python guides and Data science articles. If you sign up using my link → https://frank-andrade.medium.com/membership, I'll earn a small commission with no extra cost to you.

Join Medium with my referral link — Frank Andrade → https://frank-andrade.medium.com/membership *As a Medium member, a portion of your membership fee goes to writers you read, and you get full access to every story…*frank-andrade.medium.com → https://frank-andrade.medium.com/membership