

Forum:[Hack Forums](#)

Thread: [Scraping doesn't work the same with multithreading](#)

The Pulp Fiction - July 29, 2023, 7:58 am

So, I have developed a web scraping tool to scrape data from the website, "[Mercateo](#)". The tool has many other websites integrated into it.

Since I need to scrape like 1000 products from each website, I was forced to use multithreading/Multiprocessing.

For somereason, the website wont scrape the data completely when I use multithreading. and works fine with a for loop.

I assume, It has something something to do with website making them unable to process completely while scraping the data.

example :

```
List3 = ['09140006153', '09140009952', '09 14 000 6152', '09 14 003 4501', '09 14 000 6252',  
'09140034501', '09 14 000 6253', '09 14 003 4501', '09140006258', '345346', '61 SBK4 PA',  
'VAS17608R', '14610C C24', 'VAT11713L', 'K0737.2101X20', 'SP123231', '2746', '2591', '2748',  
'2592', '2747', '2590', '7878', 'Z17L-UK27', ]  
Code List5 = []  
for i in list3:  
    Lis = Main_file_Mercateo().search_in_website(str(i))  
    List5.append(Lis)  
print(List5)
```

And when I do the same thing with Multiprocessing, It works for the first few and then randomly returns empty values :

```
Code # import multiprocessing  
# if __name__ == '__main__':  
#     with multiprocessing.Pool() as pool:  
#         multiprocessing.freeze_support()  
#         results = pool.map(Main_file_Mercateo().search_in_website, list3)  
#     print(results)
```

Any idea why this is happening? It was working fine with the same code until few months back, so it certainly has something to do with the website, trying to handle all the traffic from one IP.

If you have any idea how to solve this issue, it would be really helpful, and you don't have to do it for free. :)

BabyKitten - July 29, 2023, 10:47 am

The website is probably rate limiting you depending on how many threads you have going. Do you have access to a vm in the cloud that you could run multiple instances from different ips? Like running multiple single threaded versions on different machines.

tamim - July 30, 2023, 5:28 am

```
List5 = []
```

```
for i in list3:  
    Lis = Main_file_Mercateo().search_in_website(str(i))  
    List5.append(Lis)  
print(List5)
```

Hackers - July 30, 2023, 7:58 am

For times like this I setup an inner loop that if it receives an error or unexpected format, it'll wait a second and then re-run. It increases overall run time especially if your timeout delay is greater than the time needed to actually wait, but you can fine tune that and have much more reliable results.

I'm assuming the server is either giving you errors or timing out. Try the retry method I mentioned, or look into exact responses to come up with a more specific, better plan for when this happens.

The other possibility is that you messed up your code and your data isn't being passed back properly or there's a simple edge case where the data your targeting has become an undefined or otherwise null / empty node/element on the web page.

We'd need to see more of your code to truly debug.

Is using proxy list with user agent match ups a reliable way for you to grab a few pages per IP or do you need to stick to one IP?

I'm assuming you're following all local and federal laws, TOS of website use, etc, so I am happy to help!

Panda™ - July 30, 2023, 10:24 am

Could be a rate limit, I would probably run it with rotating proxies if you aren't already doing that.

msmithham2346 - July 31, 2023, 3:57 am

agree with Panda™

did you Try with Rotating Quality Proxies or Rotating VPN?

Fresku - July 31, 2023, 11:41 am

Besides rotating proxies, try to randomize your footprint on the target server as much as possible.

Try using random user agents, host, origin and referer headers, this will obfuscate your scraping attempts even more.

The Pulp Fiction - August 3, 2023, 1:09 pm

Got a lot of different ideas, Thanks, folks. Closing this thread as I don't want this discussion to go further.