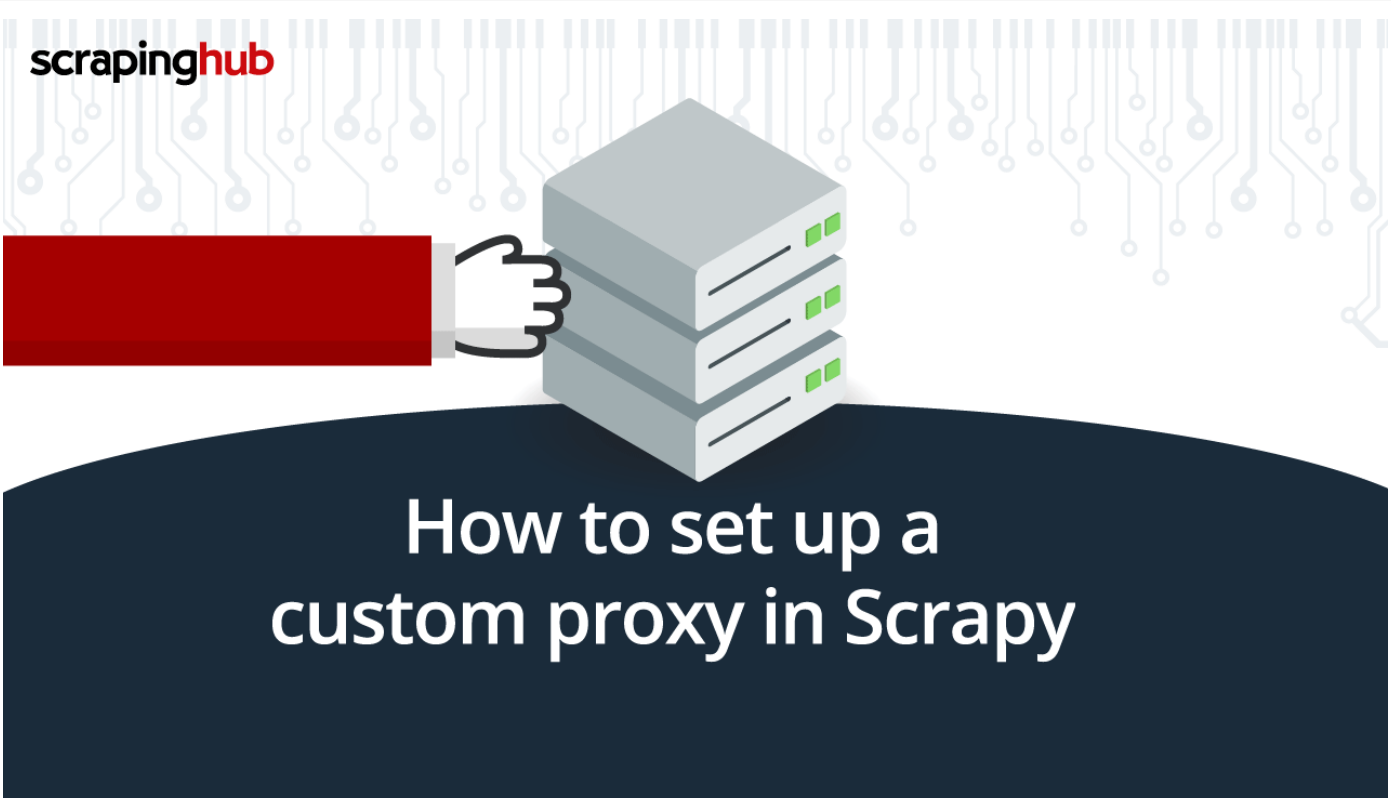
THE SCRAPINGHUB BLOG

Turn Web Content Into Useful Data



HOW TO SET UP A CUSTOM PROXY IN SCRAPY?

August 08, 2019 Attila Tóth 2 Comments

When scraping the web at a reasonable scale, you can come across a series of problems and challenges. You may want to access a website from a specific country/region. Or maybe you want to work around anti-bot solutions. Whatever the case, to overcome these obstacles you need to use and manage proxies. In this article, I'm going to cover how to set up a custom proxy inside your Scrapy spider in an easy and straightforward way. Also, we're going to discuss what are the best

ways to solve your current and future proxy issues. You will learn how to do it yourself but you can also just use Crawlera to take care of your proxies.

Why you need smart proxies for your web scraping projects?

If you are extracting data from the web at scale, you've probably already figured out the answer. **IP banning**. The website you are targeting might not like that you are extracting data even though what you are doing is totally ethical and legal. When your scraper is banned, it can really hurt your business because the incoming data flow that you were so used to is suddenly missing. Also, sometimes websites have different information displayed based on country or region. To solve these problems we use proxies for successful requests to access the public data we need.

Setting up proxies in Scrapy

Setting up a proxy inside Scrapy is easy. There are two easy ways to use proxies with Scrapy - passing proxy info as request parameter or implementing a custom proxy middleware.

Normally when you send a request in Scrapy you just pass the URL you are targeting and maybe a callback function. If you want to use a specific proxy for that URL you can pass it as a meta parameter, like this:

Option 1: Via request parameters

```
def start_requests(self):
    for url in self.start_urls:
        return Request(url=url, callback=self.parse,
                       headers={"User-Agent": "My UserAgent"},
                       meta={"proxy": "http://192.168.1.1:8050"})
```

The way it works is that inside Scrapy, there's a middleware called HttpProxyMiddleware which takes the *proxy* meta parameter from the request object and sets it up correctly as the used proxy. The middleware is enabled by default so there is no need to set it up.

Option 2: Create custom middleware

Another way to utilize proxies while scraping is to actually create your own middleware. This way the solution is more modular and isolated. Essentially, what we need to do is the same thing as when passing the proxy as meta parameter:

```
from w3lib.http import basic_auth_header

class CustomProxyMiddleware(object):
    def process_request(self, request, spider):
```

```
        request.meta[“proxy”] = "http://192.168.1.1:8050"
        request.headers[“Proxy-Authorization”] =
                        basic_auth_header(“<proxy_user>”, “<proxy_pass>”)
```

In the code above, we define the proxy URL and the necessary authentication info. Make sure that you also enable this middleware in the settings and put it before the *HttpProxyMiddleware*:

```
DOWNLOADER_MIDDLEWARES = {
    'myproject.middlewares.CustomProxyMiddleware': 350,
    'scrapy.downloadermiddlewares.httpproxy.HttpProxyMiddleware': 400,
}
```

How to verify if your proxy is working?

To verify that you are indeed scraping using your proxy you can scrape a test site which tells you your IP address and location (like this one). If it shows the proxy address and not your computer's actual IP it is working correctly.

Rotating proxies

Now that you know how to set up Scrapy to use a proxy you might think that you are done. Your IP banning problems are solved forever. Unfortunately not! What if the proxy we just set up gets banned as well? What if you need multiple proxies for multiple pages? Don't worry there is a solution called IP rotation and it is key for successful scraping projects.

When you rotate a pool of IP addresses what you're doing is essentially randomly picking one address to make the request in your scraper. If it succeeds, aka returns the proper HTML page, we can extract data and we're happy. If it fails for some reason (IP ban, timeout error, etc...) we can't extract the data so we need to pick another IP address from the pool and try again. Obviously, this can be a nightmare to manage manually so we recommend using an automated solution for this.

IP rotation in Scrapy

If you want to implement IP rotation for your Scrapy spider you can install the scrapy-rotating-proxies middleware which has been created just for this.

It will take care of the rotating itself, adjusting crawling speed and making sure that we're using proxies that are actually alive.

After installation and enabling the middleware you just have to add your proxies that you want to use as a list to *settings.py*:

```
ROTATING_PROXY_LIST = [
    'proxy1.com:8000',
    'proxy2.com:8031',
    # ...
]
```

Also, you can customize things like, ban detection method, page retries with different proxies, etc…

Conclusion

So now you know how to set up a proxy in your Scrapy project and how to manage simple IP rotation. But if you are scaling up your scraping projects you will quickly find yourself drowned in proxy related issues. Thus, you will lose data quality and ultimately you will waste a lot of time and resources dealing with proxy problems.

For example, you will find yourself dealing with unreliable proxies, you'll get poor success rate and poor data quality, etc... and really just get bogged down in the minor technical details that stop you from focusing on what really matters: making use of the data. How can we end this never-ending struggle? By using an already available solution that handles well all the mentioned headaches and struggles.

This is exactly why we created Crawlera. Crawlera enables you to reliably crawl at scale, managing thousands of proxies internally, so you don't have to. You never need to worry about rotating or swapping proxies again. Here's how you can use Crawlera with Scrapy.

If you're tired of troubleshooting proxy issues and would like to give Crawlera a try then signup today. It has a 14-day FREE trial! Or schedule a call with our crawl consultancy team.

Useful links

Discussion on Github about Socks5 proxies and scrapy

Scrapy proxy rotating middleware

Related

Building spiders made easy: GUI For Your Scrapy Shell
March 03, 2020
In "Open source" , "Scrapy" , "spider"

How To Scrape The Web Without Getting Blocked
February 27, 2020
In "Web Scraping" , "bots" , "Crawlera" , "Proxies"

Introducing Crawlera free trials & new plans
February 18, 2020
In "Crawlera"

Crawlera, Scrapy, Proxies, scrapyproject

<NEXT
Learn how to configure and utilize proxies with Python Requests module

PREVIOUS>
GDPR Update: Scraping Public Personal Data

Aurélien SCHILTZ
9/5/2019, 11:17:49 PM

There is also this middleware about proxies :

https://github.com/aivarsk/scrapy-proxies

To allow rotation and custom proxies.

Reply to *Aurélien SCHILTZ*

---

jagadeesh
11/10/2019, 8:58:06 PM

how to use key and password instead of user:password while webscraping in python

Reply to *jagadeesh*

Leave a Reply

First Name*

Last Name

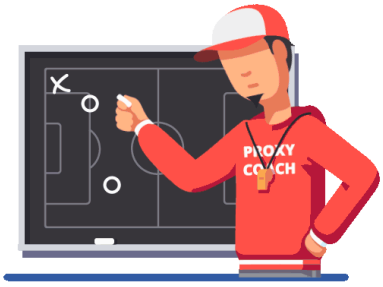
Email*

Website

Comment*

I would like to receive email updates from Scrapinghub on blog posts, products, news, events and offers. You may unsubscribe at any time.

I have read and agree to Scrapinghub's **Privacy Policy** and **Cookie Policy**. *



Story of the Month

HOW TO SCRAPE THE WEB WITHOUT GETTING BLOCKED

Getting data from publicly available websites should not be a problem. In reality though, it's not that easy. We can make it easy.

Search ...

CRAWL WEB DATA AT SCALE WITHOUT BOTTLENECKS OR SLOWDOWNS.

FOLLOW US

POPULAR POSTS

- Learn how to configure and utilize proxies with Python Requests module
- An Introduction to XPath: How to Get Started
- Handling JavaScript in Scrapy with Splash
- How to Build your own Price Monitoring Tool
- How to Crawl the Web Politely with Scrapy

RECENT POSTS

Transitioning to Remote Working as a Company

A Practical Guide to Web Data QA Part I: Validation Techniques

COVID-19: Handling the Situation as a Fully Remote Company

Extracting clean article HTML with News API

Job Postings Beta API: Extract Job Postings at Scale

CATEGORIES

Select Category

ARCHIVES

Select Month