Scraping dynamic content using python-Scrapy

Asked 9 years, 10 months ago Modified 5 years ago Viewed 60k times

48

Disclaimer: I've seen numerous other similar posts on StackOverflow and tried to do it the same way but was they don't seem to work on this website.

I'm using Python-Scrapy for getting data from koovs.com.

However, I'm not able to get the product size, which is dynamically generated. Specifically, if someone could guide me a little on getting the 'Not available' size tag from the drop-down menu on this link, I'd be grateful.

I am able to get the size list statically, but doing that I only get the list of sizes but not which of them are available.

python web-scraping scrapy

Share Improve this question Follow

edited May 24, 2015 at 17:15

asked May 20, 2015 at 9:27
Pravesh Jain
4,288 6 32 49

Correct me if I'm wrong, you are able to get the list of sizes, but having difficulties filtering only available sizes? – alecxe May 20, 2015 at 9:36

Exactly! I am able to get them statically and doing that I only get the list of sizes and not which of them are available. I'll add this to the question. – Pravesh Jain May 20, 2015 at 9:58

Would you be okay involving selenium? – alecxe May 21, 2015 at 9:04

I've never really used selenium but if it's required only to get some data and not required during the actual scraping then it's good. Could you guide me a little on how it would be used? – Pravesh Jain May 21, 2015 at 10:03

1 This question and this one helped me a lot – Henadzi Rabkin Dec 22, 2019 at 22:31

4 Answers

Sorted by: Highest score (default)

57

You can also solve it with `ScrapyJS` (no need for `selenium` and a real browser):

> This library provides Scrapy+JavaScript integration using Splash.

Follow the installation instructions for `Splash` and `ScrapyJS`, start the splash docker container:

```
$ docker run -p 8050:8050 scrapinghub/splash
```

Put the following settings into `settings.py`:

```
SPLASH_URL = 'http://192.168.59.103:8050'

DOWNLOADER_MIDDLEWARES = {
    'scrapyjs.SplashMiddleware': 725,
}

DUPEFILTER_CLASS = 'scrapyjs.SplashAwareDupeFilter'
```

And here is your sample spider that is able to see the size availability information:

```
# -*- coding: utf-8 -*-
import scrapy


class ExampleSpider(scrapy.Spider):
    name = "example"
    allowed_domains = ["koovs.com"]
    start_urls = (
        'http://www.koovs.com/only-onlall-stripe-ls-shirt-59554.html?
from=category-651&skuid=236376',
    )

    def start_requests(self):
        for url in self.start_urls:
            yield scrapy.Request(url, self.parse, meta={
                'splash': {
                    'endpoint': 'render.html',
                    'args': {'wait': 0.5}
                }
            })
    def parse(self, response):
        for option in response.css("div.select-size select.sizeOptions option")
[1:]:
            print option.xpath("text()").extract()
```

Here is what is printed on the console:

```
[u'S / 34 -- Not Available']
[u'L / 40 -- Not Available']
[u'L / 42']
```

Share Improve this answer Follow

answered May 21, 2015 at 15:56
alecxe
474k 127 1.1k 1.2k

Both great answers. Both the approaches work. I wonder if there is an advantage using one of them over the other? – Pravesh Jain May 25, 2015 at 10:18

3 @PraveshJain from what I understand, if you are okay with both the approaches, I would stick to splash - in theory, this should be faster since it doesn't involve a real browser at all. Besides, you can use this option in a non-real-screen headless environment. It is also easy to set up and there are almost no changes to the scrapy code - the key part is the middleware that scrapyjs provides. Hope that helps. – alecxe May 25, 2015 at 13:20

$ docker run -p 8050:8050 scrapinghub/splash - this command..how can i automate this command along with scrapy to scrape data using a cron job scheduler.. it obviously is not a great idea to keep docker process running at all time..may be some sh script before i make call to reactor at scheduled time ? – MrPandav Jul 3, 2015 at 9:40

1 @Chelsea the settings.py should be stored in ur project directory. ProjectName > projectName > settings.py – Genfood Jan 25, 2018 at 19:29

1 @Plasmatiger run `docker-machine ip default` in docker then change your SPLASH_URL to that, worked for me to resolve your issue. – Winters Jul 21, 2020 at 2:16

10

From what I understand, the size availability is determined dynamically in javascript being executed in the browser. Scrapy is not a browser and cannot execute javascript.

If you are okay with switching to `selenium` browser automation tool, here is a sample code:

```
from selenium import webdriver
from selenium.webdriver.support.select import Select
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.common.by import By
from selenium.webdriver.support import expected_conditions as EC

browser = webdriver.Firefox()  # can be webdriver.PhantomJS()
browser.get('http://www.koovs.com/only-onlall-stripe-ls-shirt-59554.html?
from=category-651&skuid=236376')

# wait for the select element to become visible
select_element = WebDriverWait(browser,
10).until(EC.visibility_of_element_located((By.CSS_SELECTOR, "div.select-size
select.sizeOptions")))

select = Select(select_element)
for option in select.options[1:]:
    print option.text

browser.quit()
```

It prints:

```
S / 34 -- Not Available
L / 40 -- Not Available
L / 42
```

Note that in place of `Firefox` you can use other webdrivers like Chrome or Safari. There is also an option to use a headless `PhantomJS` browser.

You can also combine Scrapy with Selenium if needed, see:

- selenium with scrapy for dynamic page
- `scrapy-webdriver`
- `seleniumcrawler`

Share Improve this answer Follow

edited May 23, 2017 at 12:18
Community Bot
1 1

answered May 21, 2015 at 13:16
alecxe
474k 127 1.1k 1.2k

4

I faced that problem and solved easily by following these steps

pip install splash
pip install scrapy-splash
pip install scrapyjs

download and install docker-toolbox

open docker-quickterminal and enter

```
$ docker run -p 8050:8050 scrapinghub/splash
```

To set the SPLASH_URL check the default ip configured in the docker machine by entering `$ docker-machine ip default` (My IP was 192.168.99.100)

```
SPLASH_URL = 'http://192.168.99.100:8050'
DOWNLOADER_MIDDLEWARES = {
    'scrapyjs.SplashMiddleware': 725,
}

DUPEFILTER_CLASS = 'scrapyjs.SplashAwareDupeFilter'
```

That's it!

Share Improve this answer Follow

answered Jun 5, 2017 at 14:11
Srivardhan Cholkar
130 2 9

docker toolbox link is broken... :.(– oldboy Jun 13, 2018 at 6:59

@Anthony - You can get docker from here: docker.com/get-docker The Docker Toolbox is for older Mac and Windows systems that do not meet the requirements of Docker for Mac and Docker for Windows. – Tony Jun 22, 2018 at 18:25

0

You have to interpret the json of the website, examples scrapy.readthedocs and testingcan.github.io

```
import scrapy
import json
class QuoteSpider(scrapy.Spider):
   name = 'quote'
   allowed_domains = ['quotes.toscrape.com']
   page = 1
   start_urls = ['http://quotes.toscrape.com/api/quotes?page=1']

   def parse(self, response):
       data = json.loads(response.text)
       for quote in data["quotes"]:
         yield {"quote": quote["text"]}
       if data["has_next"]:
           self.page += 1
           url = "http://quotes.toscrape.com/api/quotes?page=
{}".format(self.page)
           yield scrapy.Request(url=url, callback=self.parse)
```

Share Improve this answer Follow

edited Mar 10, 2020 at 2:55
kaya3
51.2k 7 85 115

answered May 25, 2019 at 18:38
Alexis Mejía
51 1 3

Start asking to get answers

Find the answer to your question by asking.

Ask question

Explore related questions

python web-scraping scrapy

See similar questions with these tags.