Kyle Hammerschmidt Ahmed Soliman Daniel Lee

CS 302 FDP-Brakes documentation

Contents

[How to Use 2](#_Toc513501421)

[How to start the spiders 2](#_Toc513501422)

[How to view the data 3](#_Toc513501423)

[Code 4](#_Toc513501424)

[Overview 4](#_Toc513501425)

[Spiders 4](#_Toc513501426)

[NAPA 4](#_Toc513501427)

[Advanced Auto 4](#_Toc513501428)

[AutoZone 5](#_Toc513501429)

[O’reilly 5](#_Toc513501430)

[Settings File 5](#_Toc513501431)

[Bot Name 5](#_Toc513501432)

[Spider Modules 5](#_Toc513501433)

[New Spider Modules 5](#_Toc513501434)

[Spider Middlewares 5](#_Toc513501435)

[Dupefilter, HTTPCache, Splash URL 5](#_Toc513501436)

[User Agent 6](#_Toc513501437)

[Robots.txt 6](#_Toc513501438)

[Download Delay 6](#_Toc513501439)

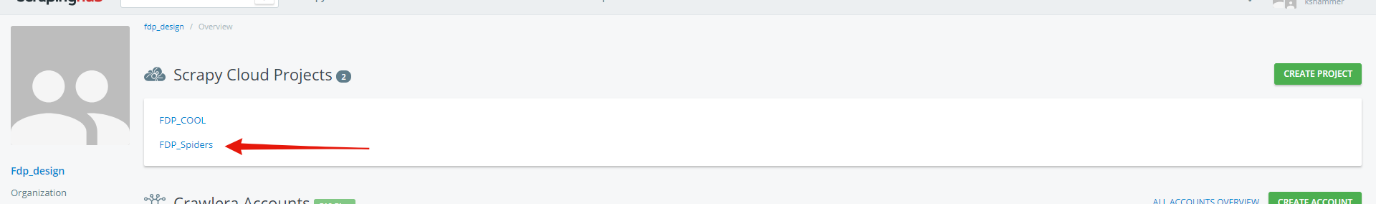
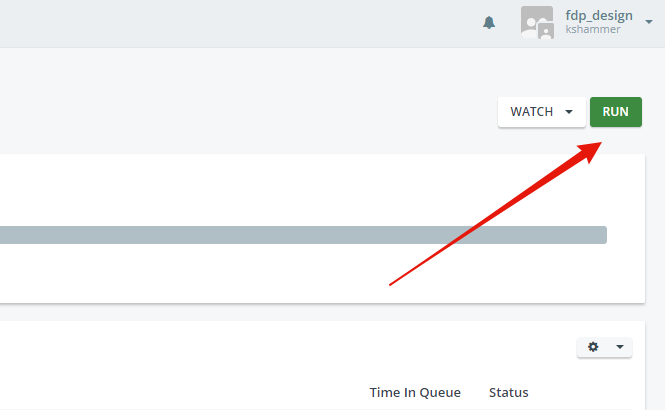
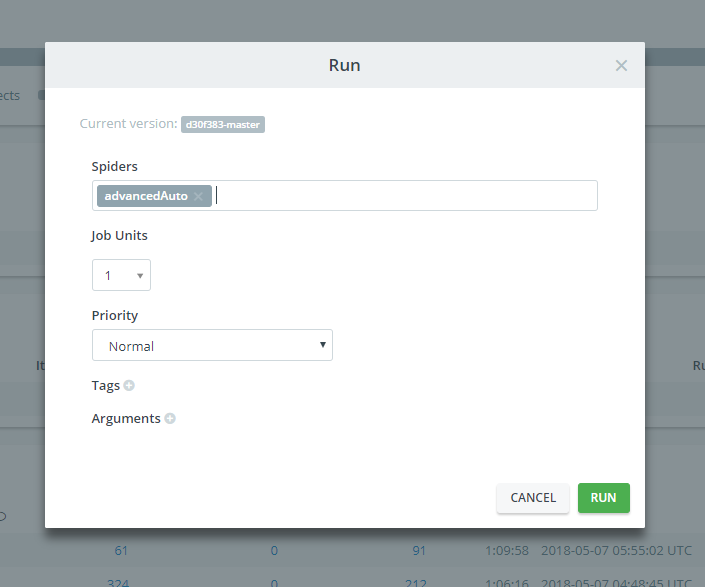
[Download Middlewares 6](#_Toc513501440)

[Crawlera 6](#_Toc513501441)

[Final Notes 6](#_Toc513501442)

# How to Use

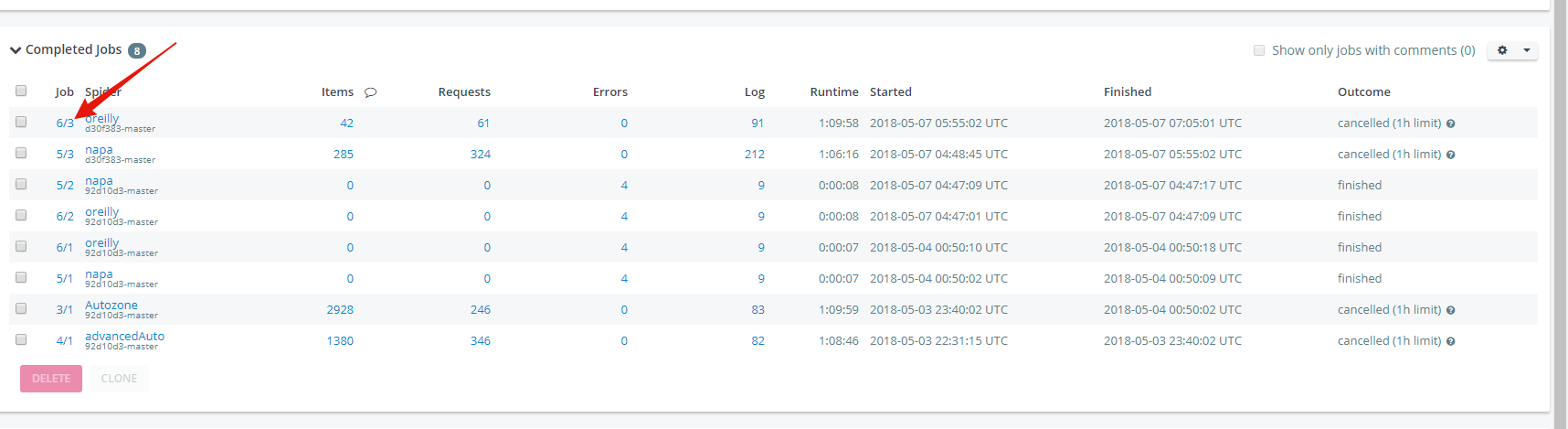
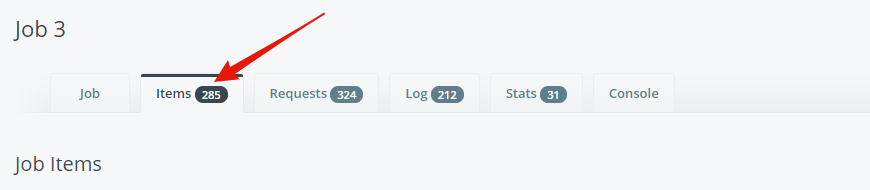
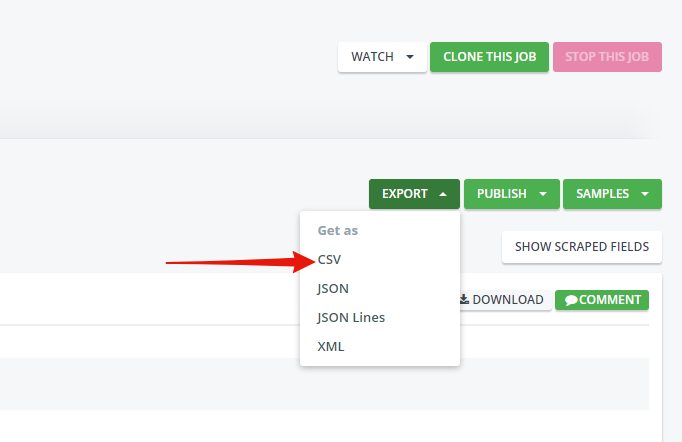
## How to start the spiders

1. Log into <https://scrapinghub.com/> (Sponsor has the admin account)
2. Click on FDP-Spiders in the Scrapy Cloud block 
3. Click on the green run button on the top right 
4. Type in the name of the spider you wish to run and click the green run button

The spider will then begin running. At the time of writing this documentation (5/7/2018) the spiders will be on the free tier of scrapy cloud. What that means is that each spider can only run for one hour and only one spider can run at a time. This can be adjusted by subscribing to a scrapy cloud package. If you start multiple spiders at once, each will wait till resources are available to run.

## How to view the data

Even though you can view the data as the spider is running it is recommended you wait for the spider to finish before viewing the data. The data retention is about a week for the scrapy cloud, so it is recommended you download the data as soon as possible.

1. In the completed jobs block of the website click on the job number you wish to view. 
2. Click on the Items tab 
3. Then click on the export button and choose the CSV option 

This will download a CSV file on your computer with the data that was scrapped.

# Code

## Overview

The spiders are written in Python 3 and use the scrapy library to run. The documentation for the scrapy library can be found here <https://doc.scrapy.org/en/latest/>. The directory hierarchy for this project should never be changed and is the default for any scrapy project. The spiders themselves are very easy to understand and the exact details for each spider are defined later in the document. Besides scrapy there are two main tools that our project utilizes. These are Splash (<https://github.com/scrapy-plugins/scrapy-splash>) and Crawlera (<https://github.com/scrapy-plugins/scrapy-crawlera>). The documentation for those products can be found on their github pages. What Splash does for the spiders is it allows them to render JavaScript. This is extremely important. A lot of pages use a JavaScript checker to make sure that the request for the page is legit. Crawlera is a proxy management service for the spiders. This allows the spiders to use multiple different IP’s to prevent them from getting banned or caught behind a CAPTCHA page. At the time of writing this documentation our project is on the C10 plan which allows for 150k requests per month which is plenty.

## Spiders

### NAPA

##### Init

The init function has a large list containing brake model numbers. In the original version of this project before it was pushed to the cloud these data points were read in through a CSV file. But to get the spiders to work on the cloud these data points needed to be hard coded. There is a second list called start URLs which contains the URL’s that the spider will go to. If more brake model numbers need to be added, they can be added to the writer list.

##### Parse

This function contains CSS selectors that will tell the spider which items to grab off the webpage. If the website is ever changed these values will need to be adjusted. Beware the website is bound to change as it has happened many times during the development cycle. Most of the CSS selectors follow the following format (tag).(classname)::text. The ::text lets the spider know to extract the text between the html tags.

### Advanced Auto

##### Init

This init function is very simple. The way the start URL was formed was through attempting to view the brake inventory by hand and recognizing a pattern in the URL as the page changed. This URL is potentially subject to change as well.

##### Parse

This function uses CSS selectors as well. The only difference between this spider and NAPA is there is some regex selectors contained in the file.

### AutoZone

##### Init

This is the simplest init function. The start URL will most likely stay the same and it contains a for loop that adds a page number to the end of the URL. The only thing about this that might change is the number of pages contained.

##### Parse

This function uses xpath selectors instead of CSS selectors. This is because the Autozone page has very complex class paths and xpath is better suited to handle that. The xpaths follow a similar pattern to the CSS selectors and should be understandable.

### O’reilly

##### Init

This init function also has hardcoded values and the reason is the same as in the NAPA spider. The search URL should stay the same.

##### Parse

This function uses xpath selectors instead of CSS selectors for a similar reason to the Autozone spider. This spider does not collect pricing information because the price is not listed on the search page. This spider will attempt to follow the link on the page to the brake in question but so far it has failed every time.

## Settings File

When creating a scrapy project a few default python files will be created. All those files are left alone except for the settings.py file.

### Bot Name

This value should be left at FDP and never changed.

### Spider Modules

This will tell the scrapy cloud where to find the spiders in the file directory. The spiders should never be moved from the spider’s folder and this value should stay the same.

### New Spider Modules

This value lets scrapy know where new spiders will be placed. This should not be changed.

### Spider Middlewares

This setting is important for splash to run and lets the spider know where the splash instance is. Since these spiders are now hosted in the cloud this setting might not be as useful but it also should not be changed.

### Dupefilter, HTTPCache, Splash URL

All these settings are only important if the spider is being run locally. Which should no longer happen. If they need to be changed please refer to the Splash documentation.

### User Agent

Currently the spiders will all use the same user agent which will mimic a Windows 10 computer making a request using chrome. If for whatever reason all the spiders begin failing at the same time this is the first place to check. It is recommended that a user agent from a desktop computer is used. There are many websites online that will return a list of popular user agents. And it is possible to figure out a host’s user agent online as well. If any changes are made to the selectors within the spiders and the spiders fail to return the correct data, it is recommended to change the user agent to the user agent of the programmer that made the changes. As time moves forward the user agent will most likely have to be updated.

### Robots.txt

This should be set to True. The scraping should be done ethically and if a website does not wish to have certain pages viewed then it should be respected. If for whatever reason a website updates its robots.txt and includes pages that the spiders used to look at. They will no longer be able to go to those pages.

### Download Delay

This setting is the number of seconds the spider waits before it loads a new page. This should be kept around 5 seconds as it allows plenty of time for a page to load. This also helps in preventing the spiders from being banned from certain pages.

### Download Middlewares

This defines which additional scrapy packages are being used. The settings for this are included in the Splash and Crawlera documentation.

### Crawlera

These settings are specific to Crawlera and their documentation can be found on the Crawlera github page.

# Final Notes

The project is hosted both on scrapy cloud and github (<https://github.com/kshammer/FDPBrakes>). Every push to github will also push the project to scrapy cloud. It will only update on scrapy cloud if the build is successful. The project was originally hosted here (<https://github.com/lobbypolice/FDP-Brakes>) but there is a bunch of junk in that github repository including many different attempts at scraping using different libraries and ideas that were ultimately unsuccessful. Because of all the excess the project could not build on scrapy cloud.