Web Scraping

Gathering Data from Websites Using Scrapy Framework (http://172.16.211.189:8088)

By,

Mr. Raghudathesh GP

Assistant Professor – Sr. Scale

Manipal School of Information Sciences

MAHE, Manipal

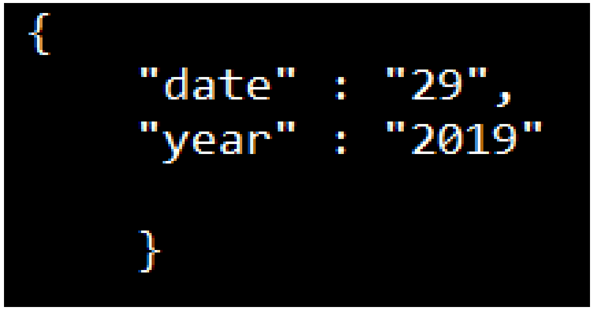
Prerequisites : Understanding HTTP Status Codes

- Status codes are issued by a server in response to a client's request made to the server
  - **110 - Connection timed out**
  - **200 - Success**
  - **404 - Not Found**

https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

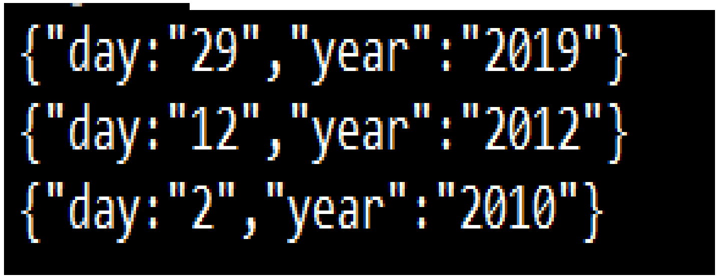
Prerequisites : JSON and JL - formats for document store

- .json - is a minimal, readable format for structuring data. It is used primarily to transmit data between a server and web application.



```
{
      "date" : "29",
      "year" : "2019"

      }
```

- JL - json lines: Every line is a json. Great for streaming data and easy for appending new jsons



```
{"day:"29","year":"2019"}
{"day:"12","year":"2012"}
{"day:"2","year":"2010"}
```

http://jsonlines.org/

Why Scrapy ?

- Scrapy is a open source and collaborative framework for crawling/scraping the web
  - Scrapy is an excellent choice for focused crawls
  - Scrapy is written in Python
  - Scrapy is faster than Heritr

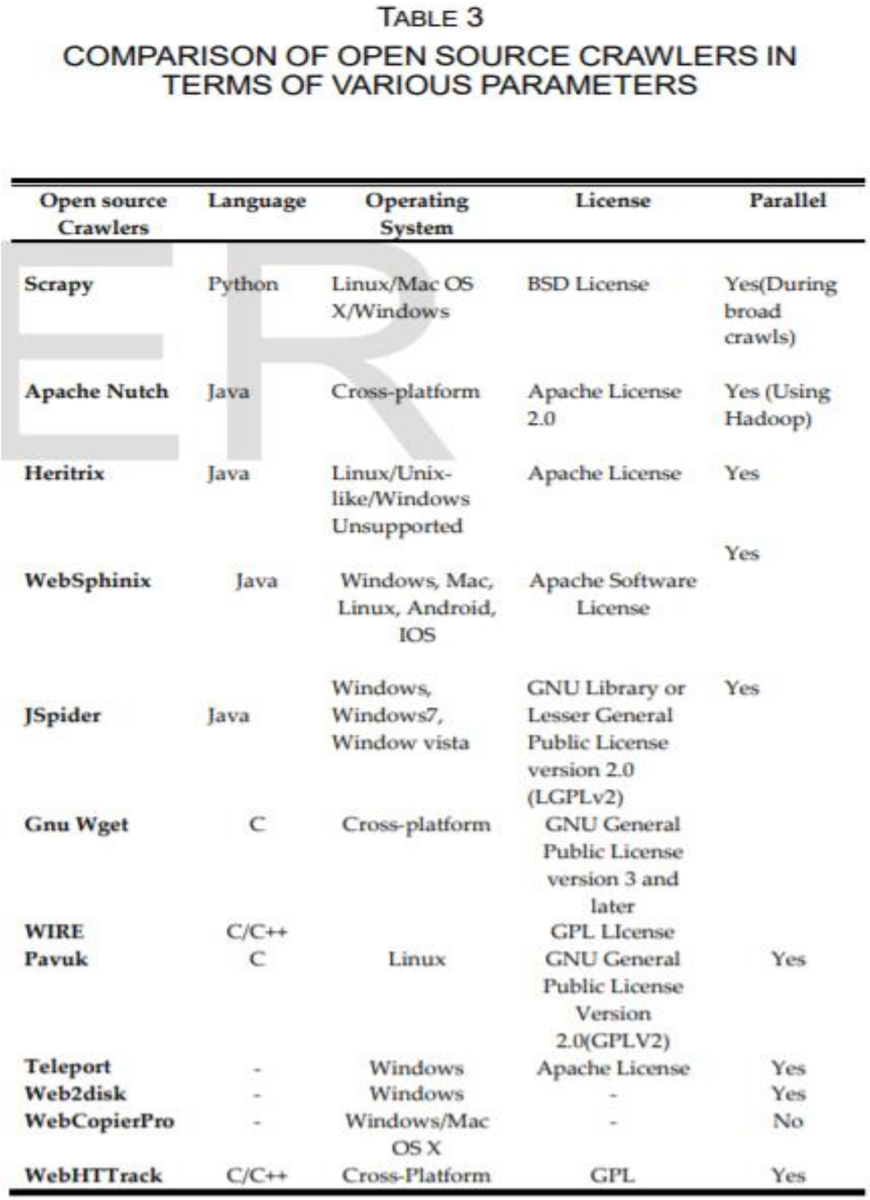
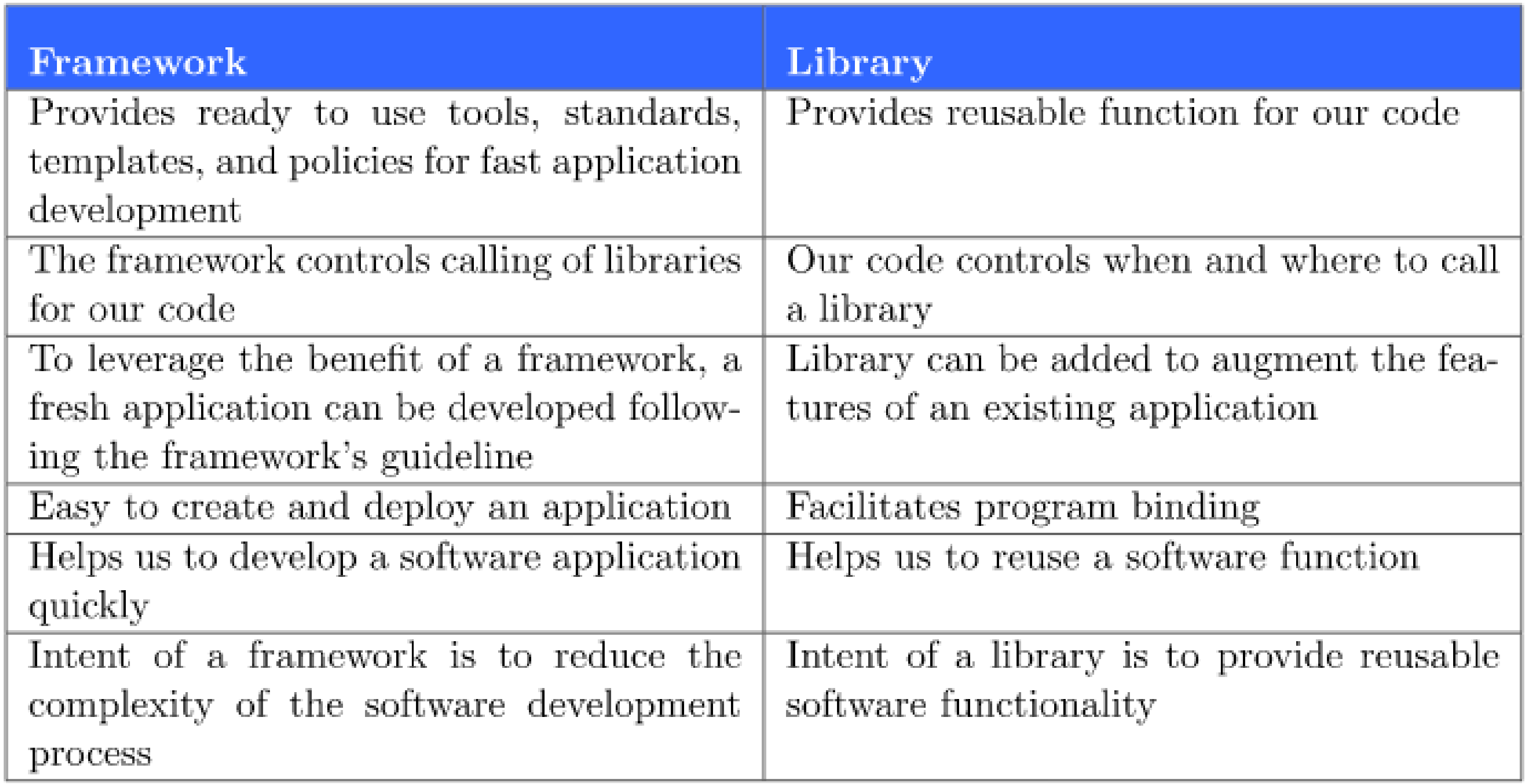


TABLE 3
COMPARISON OF OPEN SOURCE CRAWLERS IN TERMS OF VARIOUS PARAMETERS

Open source Crawlers	Language	Operating System	License	Parallel
Scrapy	Python	Linux/Mac OS X/Windows	BSD License	Yes(During broad crawls)
Apache Nutch	Java	Cross-platform	Apache License 2.0	Yes (Using Hadoop)
Heritrix	Java	Linux/Unix-like/Windows Unsupported	Apache License	Yes
WebSphinix	Java	Windows, Mac, Linux, Android, IOS	Apache Software License	Yes
JSpider	Java	Windows, Windows7, Window vista	GNU Library or Lesser General Public License version 2.0 (LGPLv2)	Yes
Gnu Wget	C	Cross-platform	GNU General Public License version 3 and later	
WIRE	C/C++		GPL LIcense	
Pavuk	C	Linux	GNU General Public License Version 2.0(GPLV2)	Yes
Teleport	-	Windows	Apache License	Yes
Web2disk	-	Windows	-	Yes
WebCopierPro	-	Windows/Mac OS X	-	No
WebHTTrack	C/C++	Cross-Platform	GPL	Yes

Yadav, M., & Goyal, N. (2015). Comparison of Open Source Crawlers-A Review. *International Journal of Scientific & Engineering Research*, *6*(9), 1544-1551.

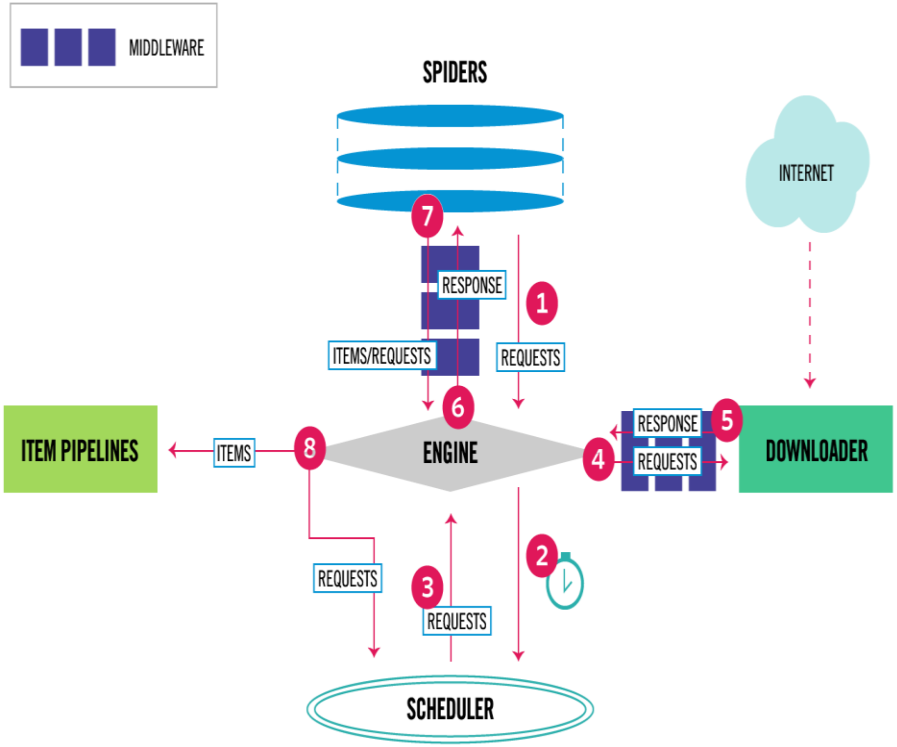
Difference Between Library and Framework



Framework	Library
Provides ready to use tools, standards, templates, and policies for fast application development	Provides reusable function for our code
The framework controls calling of libraries for our code	Our code controls when and where to call a library
To leverage the benefit of a framework, a fresh application can be developed following the framework's guideline	Library can be added to augment the features of an existing application
Easy to create and deploy an application	Facilitates program binding
Helps us to develop a software application quickly	Helps us to reuse a software function
Intent of a framework is to reduce the complexity of the software development process	Intent of a library is to provide reusable software functionality

https://www.baeldung.com/cs/framework-vs-library/

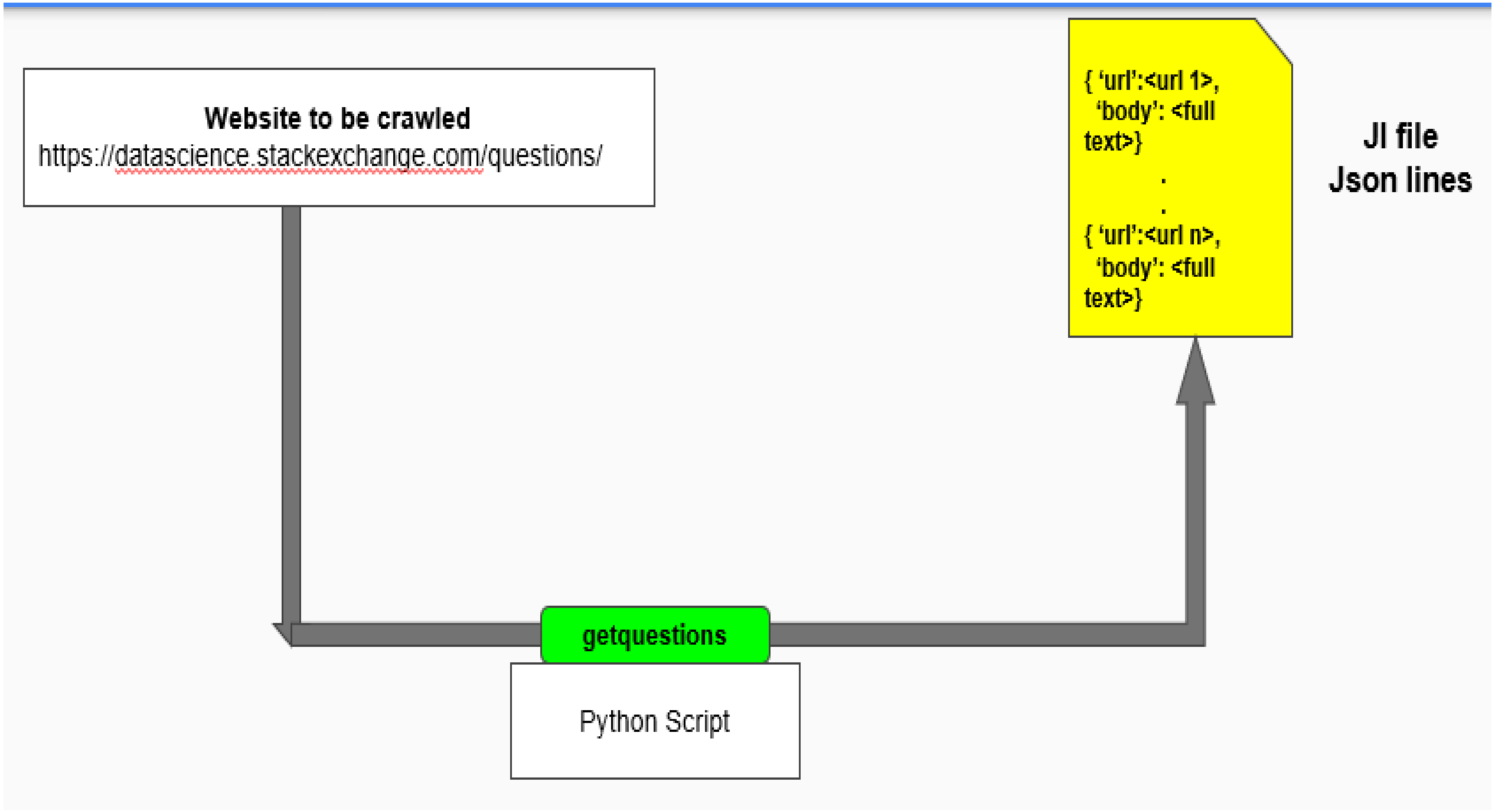
Scrapy Architecture



Source - Architecture overview — Scrapy 2.4.1 documentation

1. The **Engine** gets the initial Requests to crawl from the **Spider**.
2. The **Engine** schedules the Requests in the **Scheduler** and asks for the next Requests to crawl.
3. The **Scheduler** returns the next Requests to the **Engine** .
4. The **Engine** sends the Requests to the **Downloader**, passing through the Downloader Middlewares (see process_request()).
5. Once the page finishes downloading the **Downloader** generates a Response (with that page) and sends it to the Engine, passing through the Downloader Middlewares (see process_response()).
6. The **Engine** receives the Response from the **Downloader** and sends it to the **Spider** for processing, passing through the Spider Middleware (see process_spider_input()).
7. The **Spider** processes the Response and returns scraped items and new Requests (to follow) to the **Engine** , passing through the Spider Middleware (see process_spider_output()).
8. The **Engine** sends processed items to Item Pipelines, then send processed Requests to the **Scheduler** and asks for possible next Requests to crawl.
9. The process repeats (from step 1) until there are no more requests from the **Scheduler** .

Roadmap



Website to be crawled
https://datascience.stackexchange.com/questions/

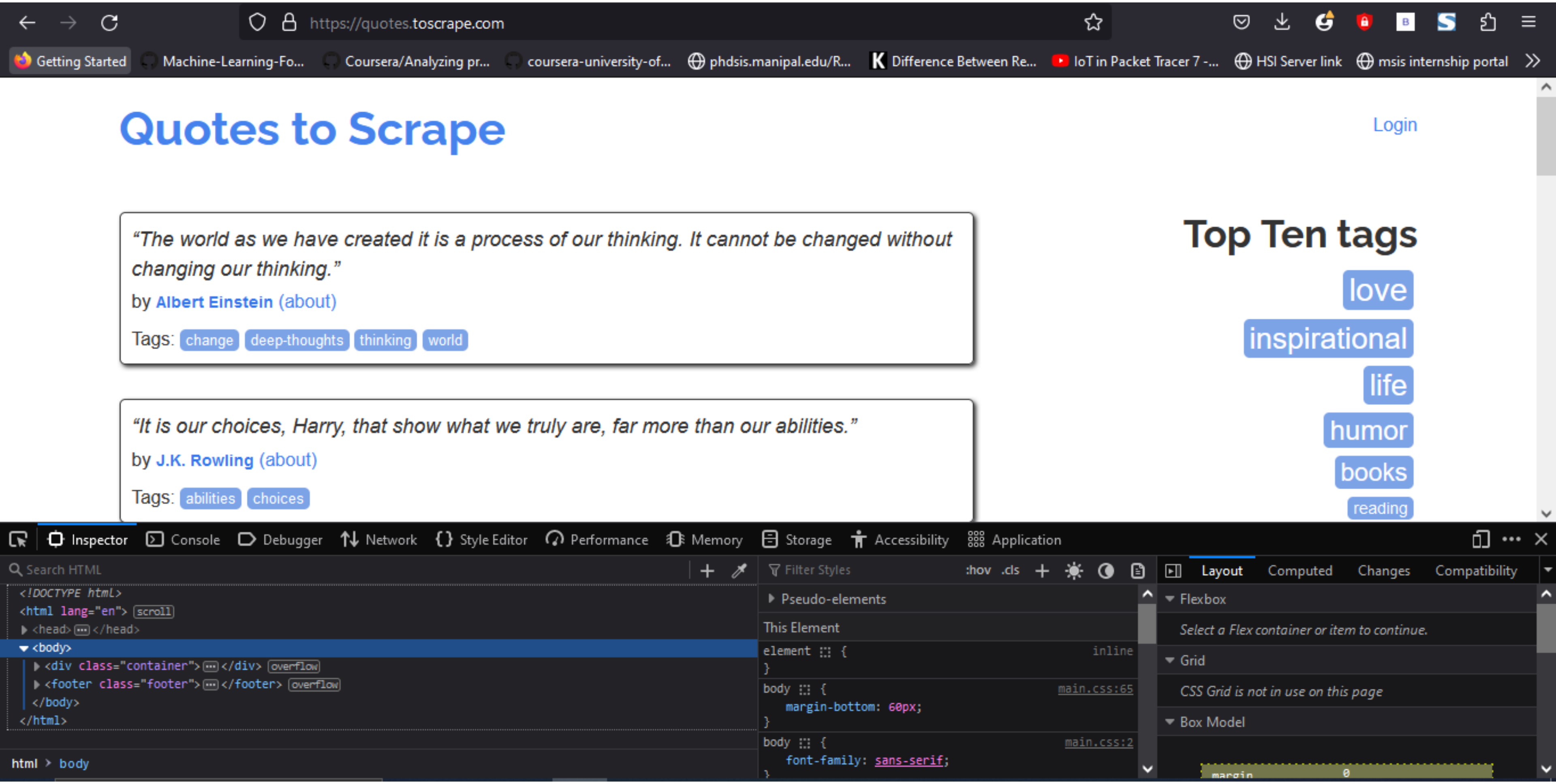
getquestions

Python Script

{ 'url':<url 1>,
'body': <full
text>}
.
.
{ 'url':<url n>,
'body': <full
text>}

Jl file
Json lines

Understand the webpage we are crawling



Quotes to Scrape

Login

"The world as we have created it is a process of our thinking. It cannot be changed without changing our thinking."

by **Albert Einstein** (about)

Tags: change deep-thoughts thinking world

"It is our choices, Harry, that show what we truly are, far more than our abilities."

by **J.K. Rowling** (about)

Tags: abilities choices

Top Ten tags

love
inspirational
life
humor
books
reading

```
<!DOCTYPE html>
<html lang="en"> scroll
  <head>…</head>
  <body>
    <div class="container">…</div> overflow
    <footer class="footer">…</footer> overflow
  </body>
</html>
```

Pseudo-elements

This Element

```
element ⁝⁝⁝ {                    inline
}
body ⁝⁝⁝ {                 main.css:65
  margin-bottom: 60px;
}
body ⁝⁝⁝ {                  main.css:2
  font-family: sans-serif;
}
```

Flexbox

Select a Flex container or item to continue.

Grid

CSS Grid is not in use on this page

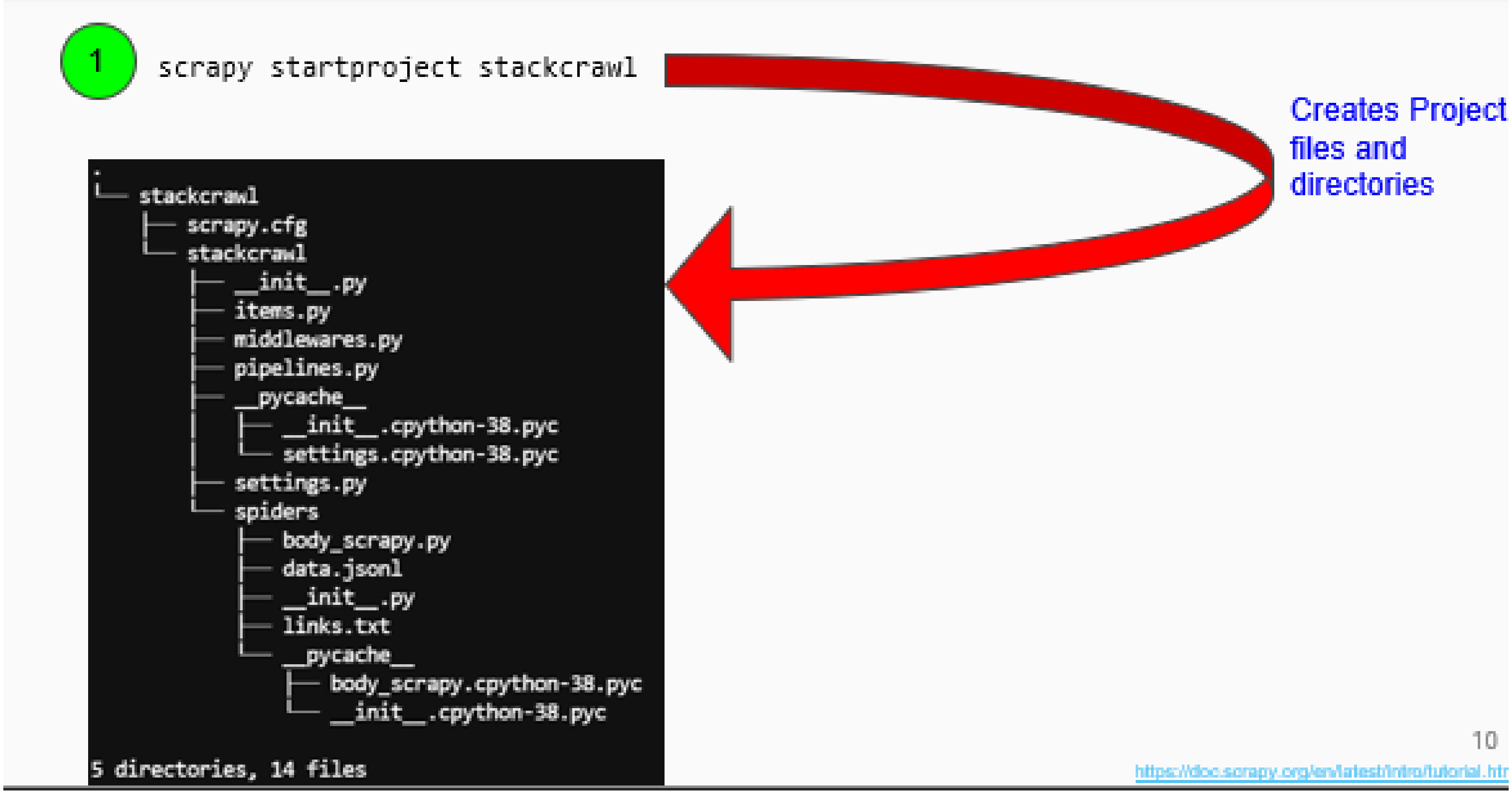
Box Model

margin 0

Scrapy basics

```
> scrapy startproject tutorial

New Scrapy project 'tutorial', using template directory
'/usr/local/lib/python3.5/site-packages/scrapy/templates/project',
created in:
    /Users/mtodor/Projects/meetups/tutorial

You can start your first spider with:
    cd tutorial
    scrapy genspider example example.com

> cd tutorial

> scrapy genspider example example.com

> scrapy crawl example -t json -o output.json
```

Starting a Project in Scrapy



1

```
scrapy startproject stackcrawl
```

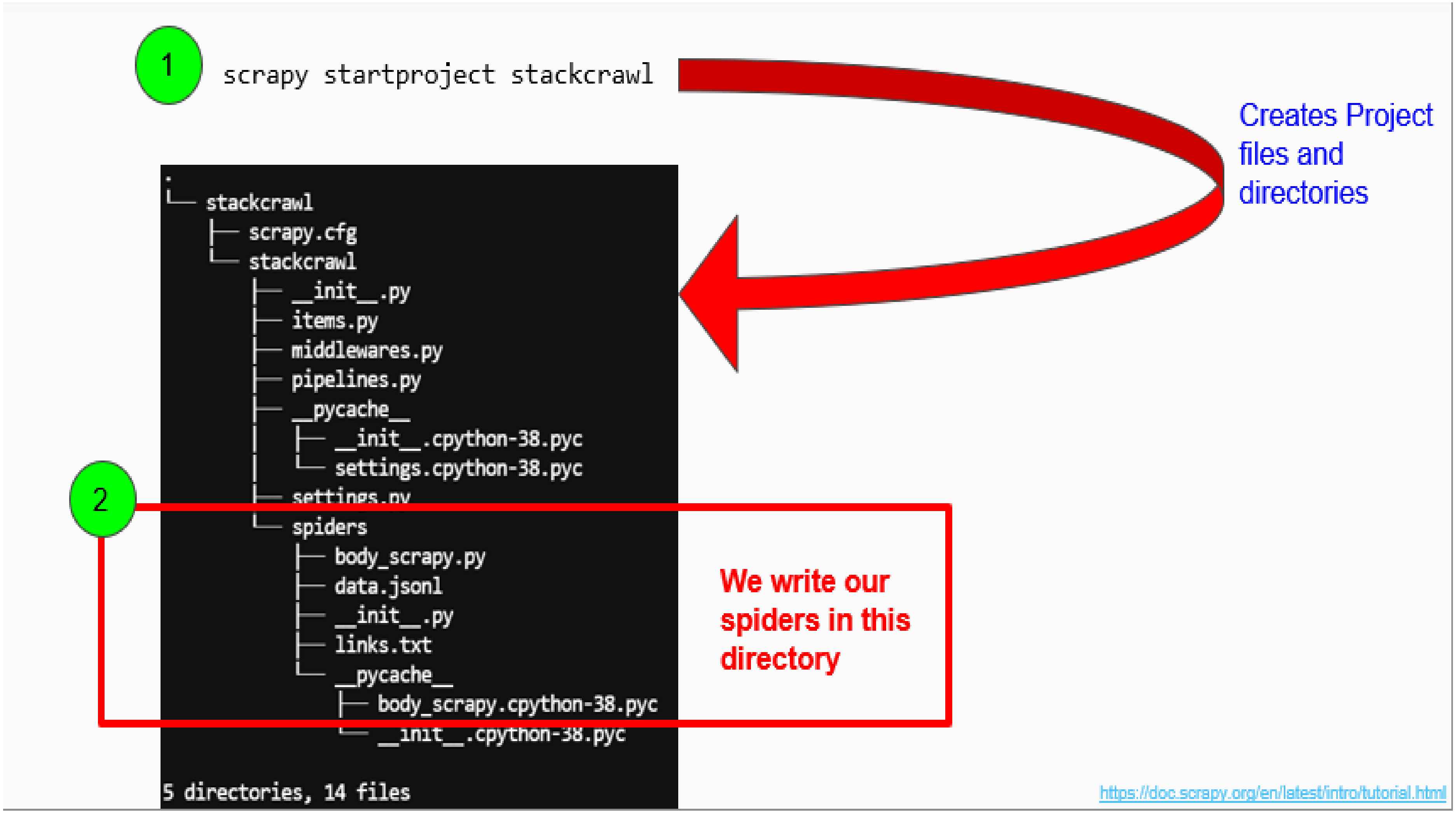
Creates Project files and directories

```
.
└── stackcrawl
    ├── scrapy.cfg
    └── stackcrawl
        ├── __init__.py
        ├── items.py
        ├── middlewares.py
        ├── pipelines.py
        ├── __pycache__
        │   ├── __init__.cpython-38.pyc
        │   └── settings.cpython-38.pyc
        ├── settings.py
        └── spiders
            ├── body_scrapy.py
            ├── data.jsonl
            ├── __init__.py
            ├── links.txt
            └── __pycache__
                ├── body_scrapy.cpython-38.pyc
                └── __init__.cpython-38.pyc

5 directories, 14 files
```

10

https://doc.scrapy.org/en/latest/intro/tutorial.htr

Starting a Project in Scrapy



```
scrapy startproject stackcrawl
```

Creates Project files and directories

```
.
└── stackcrawl
    ├── scrapy.cfg
    └── stackcrawl
        ├── __init__.py
        ├── items.py
        ├── middlewares.py
        ├── pipelines.py
        ├── __pycache__
        │   ├── __init__.cpython-38.pyc
        │   └── settings.cpython-38.pyc
        ├── settings.py
        └── spiders
            ├── body_scrapy.py
            ├── data.jsonl
            ├── __init__.py
            ├── links.txt
            └── __pycache__
                ├── body_scrapy.cpython-38.pyc
                └── __init__.cpython-38.pyc

5 directories, 14 files
```

We write our spiders in this directory

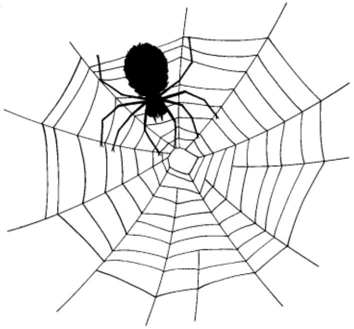
https://doc.scrapy.org/en/latest/intro/tutorial.html

Scrapy basics

```
> cd tutorial && ls *
__init__.py
items.py
pipelines.py
settings.py
…
spiders:
__init__.py
__pycache__
fivethirtyeight.py
```

Scrapy spiders

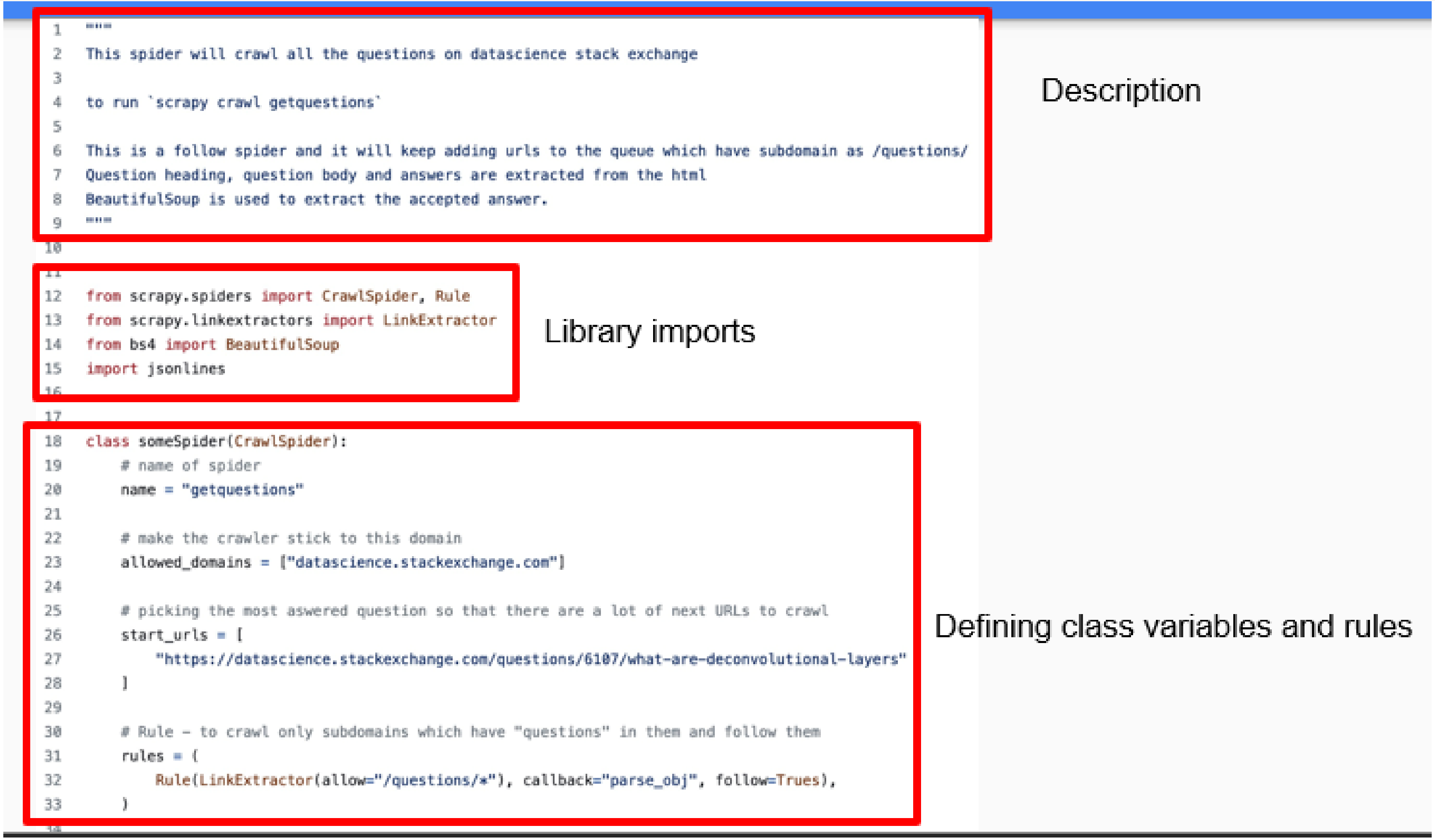
- Spiders are the place where you define the custom behaviour for crawling and parsing pages for a particular site (or, in some cases, a group of sites). http://doc.scrapy.org/en/latest/topics/spiders.html



```
class ExampleSpider(scrapy.Spider):
    name = "example"
    allowed_domains = ["example.com"]
    start_urls = ['http://www.example.com/’]
    def parse(self, response):
        ...
        process response
        ...
```

- Process **response** using selectors
  - xpath()
  - css()
  - extract()
  - re()
- Follow links
  - yield scrapy.Request(url, callback=self.parse_link)

Spider class to crawl



```
"""
This spider will crawl all the questions on datascience stack exchange

to run `scrapy crawl getquestions`

This is a follow spider and it will keep adding urls to the queue which have subdomain as /questions/
Question heading, question body and answers are extracted from the html
BeautifulSoup is used to extract the accepted answer.
"""


from scrapy.spiders import CrawlSpider, Rule
from scrapy.linkextractors import LinkExtractor
from bs4 import BeautifulSoup
import jsonlines


class someSpider(CrawlSpider):
    # name of spider
    name = "getquestions"

    # make the crawler stick to this domain
    allowed_domains = ["datascience.stackexchange.com"]

    # picking the most aswered question so that there are a lot of next URLs to crawl
    start_urls = [
        "https://datascience.stackexchange.com/questions/6107/what-are-deconvolutional-layers"
    ]

    # Rule - to crawl only subdomains which have "questions" in them and follow them
    rules = (
        Rule(LinkExtractor(allow="/questions/*"), callback="parse_obj", follow=Trues),
    )
```

Description

Library imports

Defining class variables and rules

scrapy crawl quotes-1

scrapy crawl <name_of_spider>

Scrapy item exporters

- Default exporters:
  - json': 'scrapy.exporters.JsonItemExporter'
  - 'jsonlines': 'scrapy.exporters.JsonLinesItemExporter'
  - 'jl': 'scrapy.exporters.JsonLinesItemExporter'
  - 'csv': 'scrapy.exporters.CsvItemExporter'
  - 'xml': 'scrapy.exporters.XmlItemExporter'
  - 'marshal': 'scrapy.exporters.MarshalItemExporter'
  - 'pickle': 'scrapy.exporters.PickleItemExporter’

http://doc.scrapy.org/en/latest/topics/exporters.html

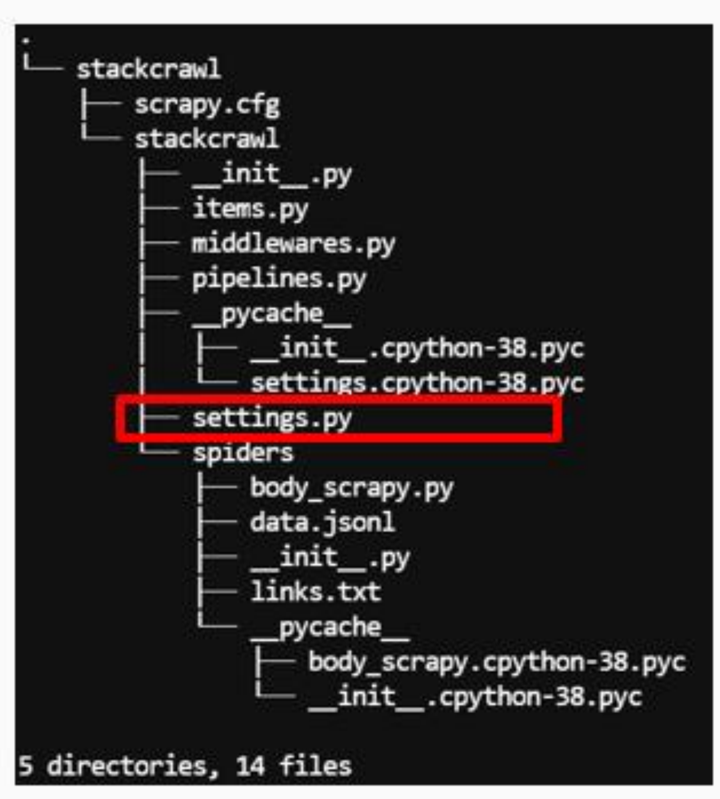
Scrapy Item Pipelines

- After an item has been scraped by a spider, it is sent to the Item Pipeline which processes it through several components that are executed sequentially.

http://doc.scrapy.org/en/latest/topics/item-pipeline.html

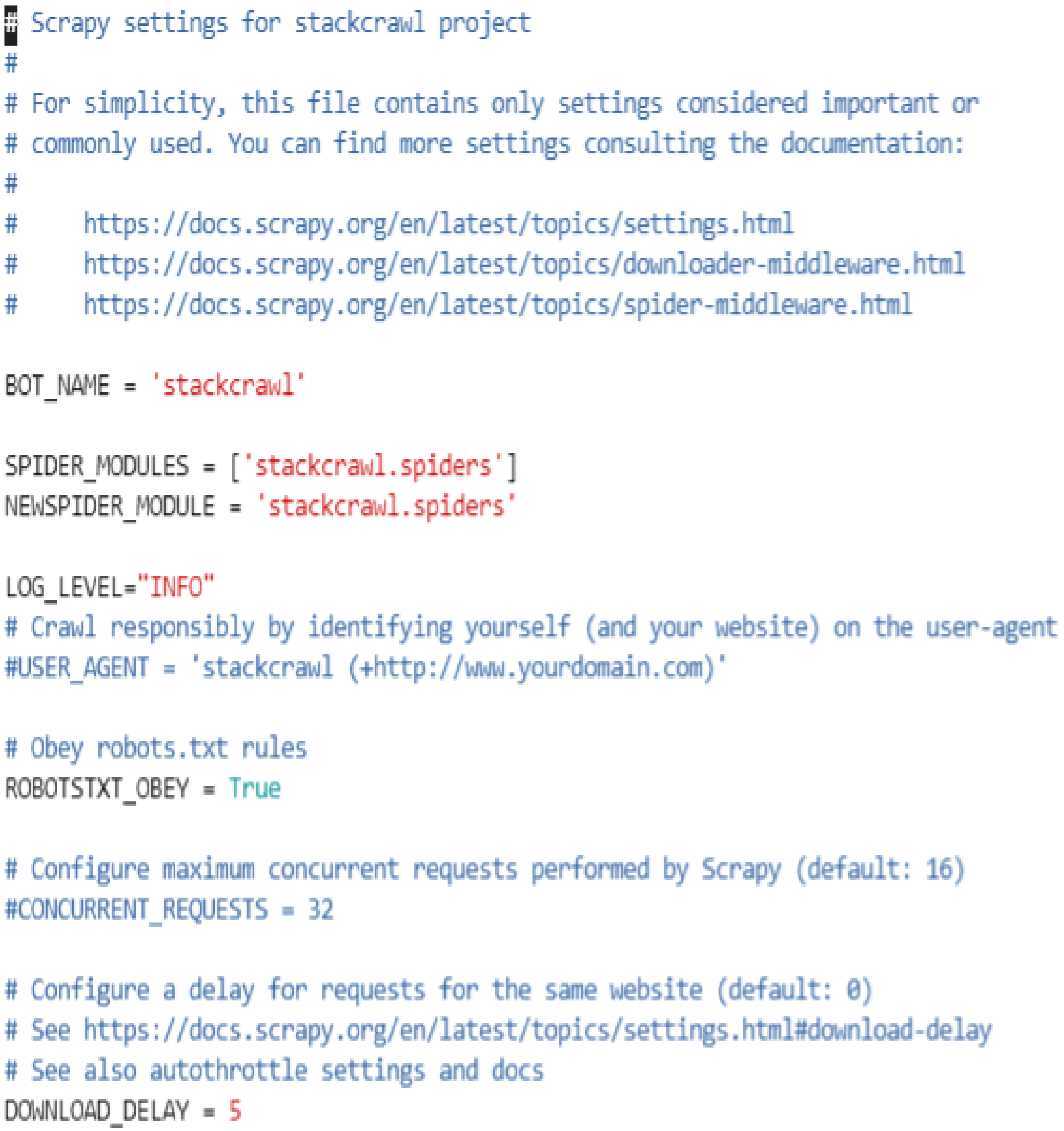
Scrapy Settings.py - the devil is in the details

- settings.py
  - LOG_LEVEL = 'INFO’
  - FEED_EXPORTERS = {'json': 'wiki_logs.exporters.UnicodeJsonItemExporter'}
    - Create a custom JSON exporter because the builtin one is brain damaged and forces ASCII output
  - ROBOTSTXT_OBEY = False
    - or, be polite and respect robots.txt



```
.
└── stackcrawl
    ├── scrapy.cfg
    └── stackcrawl
        ├── __init__.py
        ├── items.py
        ├── middlewares.py
        ├── pipelines.py
        ├── __pycache__
        │   ├── __init__.cpython-38.pyc
        │   └── settings.cpython-38.pyc
        ├── settings.py
        └── spiders
            ├── body_scrapy.py
            ├── data.jsonl
            ├── __init__.py
            ├── links.txt
            └── __pycache__
                ├── body_scrapy.cpython-38.pyc
                └── __init__.cpython-38.pyc

5 directories, 14 files
```

Before Crawling - Understanding settings.py



```
# Scrapy settings for stackcrawl project
#
# For simplicity, this file contains only settings considered important or
# commonly used. You can find more settings consulting the documentation:
#
#     https://docs.scrapy.org/en/latest/topics/settings.html
#     https://docs.scrapy.org/en/latest/topics/downloader-middleware.html
#     https://docs.scrapy.org/en/latest/topics/spider-middleware.html

BOT_NAME = 'stackcrawl'

SPIDER_MODULES = ['stackcrawl.spiders']
NEWSPIDER_MODULE = 'stackcrawl.spiders'

LOG_LEVEL="INFO"
# Crawl responsibly by identifying yourself (and your website) on the user-agent
#USER_AGENT = 'stackcrawl (+http://www.yourdomain.com)'

# Obey robots.txt rules
ROBOTSTXT_OBEY = True

# Configure maximum concurrent requests performed by Scrapy (default: 16)
#CONCURRENT_REQUESTS = 32

# Configure a delay for requests for the same website (default: 0)
# See https://docs.scrapy.org/en/latest/topics/settings.html#download-delay
# See also autothrottle settings and docs
DOWNLOAD_DELAY = 5
```

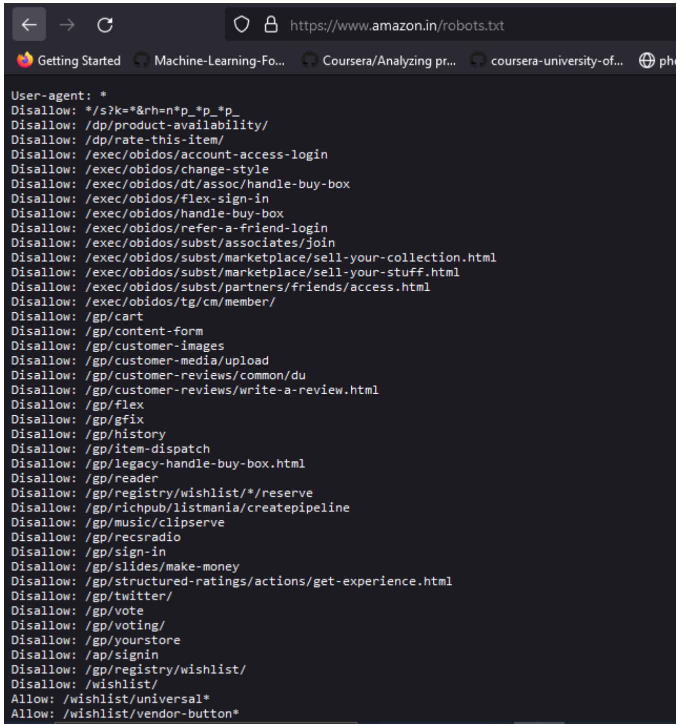
Avoid getting banned

- rotate your user agent from a pool of well-known ones from browsers (google around to get a list of them)
- disable cookies (see **COOKIES_ENABLED**) as some sites may use cookies to spot bot behaviour
- use download delays (5 or higher). See **DOWNLOAD_DELAY** setting.
- if possible, use Google cache to fetch pages, instead of hitting the sites directly

How to Crawl the Web Politely

- What Makes a Crawler Polite?
  - A polite crawler respects **robots.txt**
  - A polite crawler never degrades a website’s performance
  - A polite crawler identifies its creator with contact information
  - A polite crawler is not a pain a for system administrators

https://blog.scrapinghub.com/2016/08/25/how-to-crawl-the-web-politely-with-scrapy/



```
User-agent: *
Disallow: */s?k=*&rh=n*p_*p_*p_
Disallow: /dp/product-availability/
Disallow: /dp/rate-this-item/
Disallow: /exec/obidos/account-access-login
Disallow: /exec/obidos/change-style
Disallow: /exec/obidos/dt/assoc/handle-buy-box
Disallow: /exec/obidos/flex-sign-in
Disallow: /exec/obidos/handle-buy-box
Disallow: /exec/obidos/refer-a-friend-login
Disallow: /exec/obidos/subst/associates/join
Disallow: /exec/obidos/subst/marketplace/sell-your-collection.html
Disallow: /exec/obidos/subst/marketplace/sell-your-stuff.html
Disallow: /exec/obidos/subst/partners/friends/access.html
Disallow: /exec/obidos/tg/cm/member/
Disallow: /gp/cart
Disallow: /gp/content-form
Disallow: /gp/customer-images
Disallow: /gp/customer-media/upload
Disallow: /gp/customer-reviews/common/du
Disallow: /gp/customer-reviews/write-a-review.html
Disallow: /gp/flex
Disallow: /gp/gfix
Disallow: /gp/history
Disallow: /gp/item-dispatch
Disallow: /gp/legacy-handle-buy-box.html
Disallow: /gp/reader
Disallow: /gp/registry/wishlist/*/reserve
Disallow: /gp/richpub/listmania/createpipeline
Disallow: /gp/music/clipserve
Disallow: /gp/recsradio
Disallow: /gp/sign-in
Disallow: /gp/slides/make-money
Disallow: /gp/structured-ratings/actions/get-experience.html
Disallow: /gp/twitter/
Disallow: /gp/vote
Disallow: /gp/voting/
Disallow: /gp/yourstore
Disallow: /ap/signin
Disallow: /gp/registry/wishlist/
Disallow: /wishlist/
Allow: /wishlist/universal*
Allow: /wishlist/vendor-button*
```

Queries

THANK YOU…