Извличане на информация 2019/2020

Упражнение "Crawling and Scraping with Scrapy" 04.11.2019

Въведение

Web Crawling vs. Scraping

- **Web Crawling**: the process of iteratively finding and fetching web links
- **Web Scraping**: the process of processing a web document and extracting information out of it

Source: *stackoverflow.com*

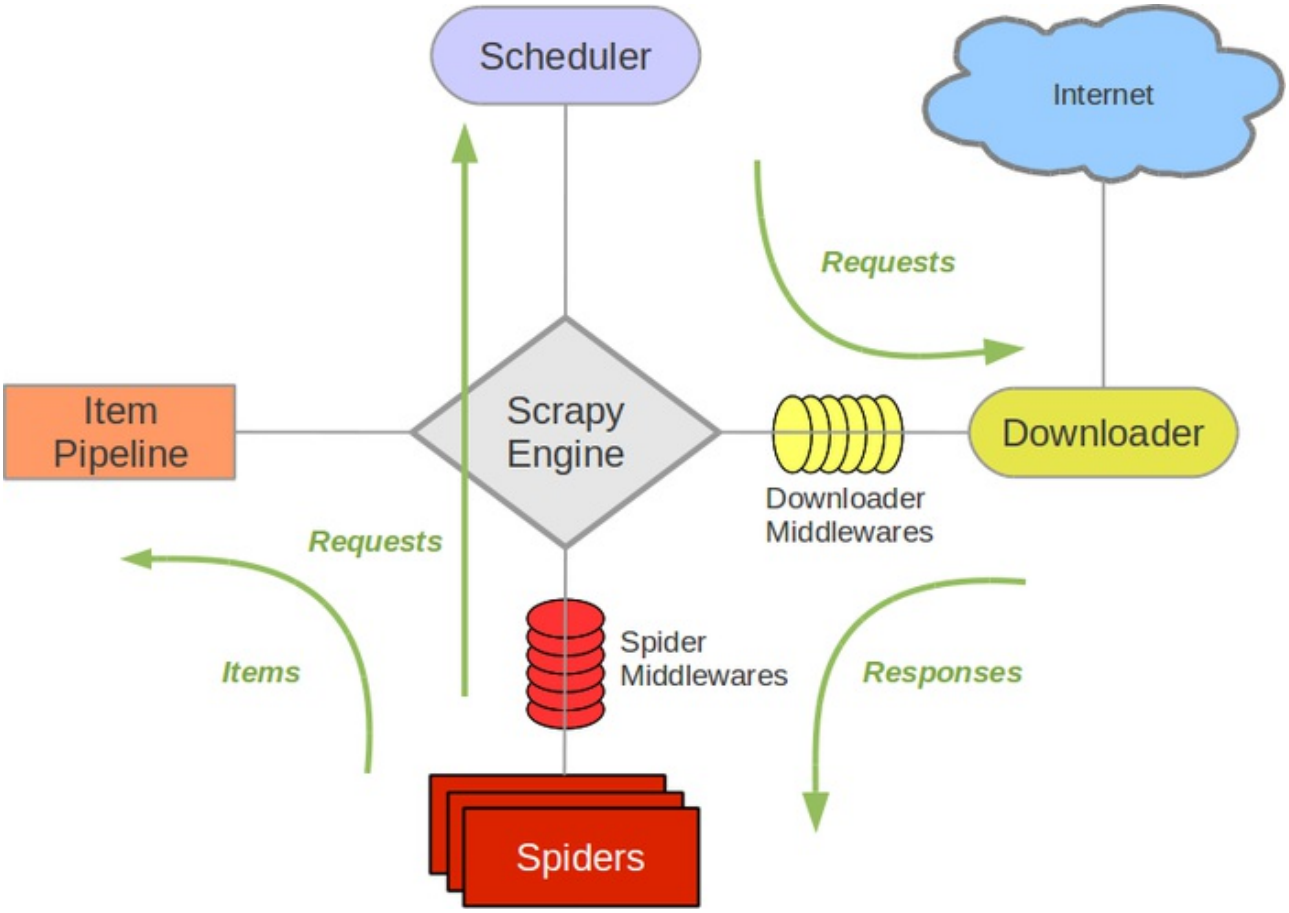
Use-Cases

1. Extracting infomration from a particilar website ("focused crawl")
2. Saving web pages into a structured, easy to use format, e.g. JSON, CSV, XML.
3. Examples: Extract documents from `.gov` web sites, extract forum data, download news articles, etc.

Why Scrapy

1. Easy to use and setup
2. Written in Python
3. Rich documentation
4. There is a Scrapy Cloud (https://scrapinghub.com/scrapy-cloud)

Architecture



Предварителни изисквания

Трябва да имате инсталирана версия на Python3.6+. Препоръчително е да използвате venv или conda.

За да инсталирате пакетите нужни на проекта използвате следната команда:

```
pip install -r requirements.txt
```

Упражнение

Условие на упражнението

Напишете спайдър (spider), който да извлече информация за преподавателите във ФМИ от
[регистъра с преподаватели (https://www.fmi.uni-sofia.bg/bg/faculty-staff)](https://www.fmi.uni-sofia.bg/bg/faculty-staff) качен на уеб сайта на факултета.

Обиколете всички (16 към 04.11.2019) страници с преподаватели.
Запишете извлечените записи в избран от вас формат *(напр. CSV, JSON, JSON-lines, XML, и т.н.)*.

За всеки преподавател извлечете следните полета:

1. Име
2. Титли (като масив от низове)
3. Катедра
4. Ел. Поща

5. Телефон
6. Кабинет
7. Снимка

Допълнителни условия:

1. Запишете всички снимки като отделни файлове на диска в папка ./images, а във файла запишете само пътя до
   снимката.
2. Преименувайте всички снимки с името на преподавателя. (Use Pipeline)
3. Създайте още два файла със същата структура - Преподаватели с кабинети във ФМИ и такива в други сгради.
   (Use Pipeline)

Всички файлове запишете в папка ./assets/

Използвайте документация на [Scrapy (http://doc.scrapy.org/en/latest/)](http://doc.scrapy.org/en/latest/).

Описание на проекта

Създаване на Scrapy проект

Преди да започнете трябва да създадете нов Scrapy проект.
От главната директория на проекта, в която искате да съхраните кода си, стартирайте:

```
scrapy startproject NAME_OF_YOUR_PROJECT
```

Това ще създаде следните файлове:

```
NAME_OF_YOUR_PROJECT/
    scrapy.cfg            # deploy configuration file
    NAME_OF_YOUR_PROJECT/             # project's Python module, you'll import
your code from here
        __init__.py
        items.py          # project items definition file
        middlewares.py    # project middlewares file
        pipelines.py      # project pipelines file
        settings.py       # project settings file
        spiders/          # a directory where you'll later put your spiders
            __init__.py
```

Тази стъпка е направена за вас и имате създаден проект.

Извличане на данни през Scrapy shell

Най-добрият начин да научите как да извличате данни с Scrapy е да експериментирате с различни селектори с помощта на
[Scrapy shell (https://docs.scrapy.org/en/latest/topics/shell.html#topics-shell)](https://docs.scrapy.org/en/latest/topics/shell.html#topics-shell).

```
scrapy shell 'URL_TO_CRAWL'
```

Примери

Отваряме страницата на проф. Иван Койчев през конзолата:

```
scrapy shell 'https://www.fmi.uni-sofia.bg/bg/faculty/ivan-koychev'
```

Извличаме пътя до снимката чрез следния CSS селектор:

```
>>> response.css('a.image-popup::attr(href)').get()
'https://www.fmi.uni-sofia.bg/sites/default/files/pictures/staff/koychev.jpg'
```

Други валидни конструкции:

CSS selectors

```
>>> response.css('title')
[<Selector xpath='descendant-or-self::title' data='<title>Quotes to
Scrape</title>'>]
response.css('title::text').getall()
['Quotes to Scrape']
response.css('title').getall()
['<title>Quotes to Scrape</title>']
```

Regex extractors

```
>>> response.css('title::text').re(r'Quotes.*')
['Quotes to Scrape']
>>> response.css('title::text').re(r'Q\w+')
['Quotes']
>>> response.css('title::text').re(r'(\w+) to (\w+)')
['Quotes', 'Scrape']
```

Създаване на спайдър

```
import scrapy


class QuotesSpider(scrapy.Spider):
    name = "YOUR_SPIDER_NAME"
    start_urls = [
        'START_URL',
    ]

    def parse(self, response):
        # Extract your data and turn it into a python dictionary, or an object
which you can define in items.py
        for item in SOME_ITEMS_FROM_REPONSE:
            yield DICT_OR_ITEM_CLASS

        # You can also yield multiple links, but you need to extract them
properly
        next_page = EXTRACT_YOUR_NEXT_PAGE
        if next_page is not None:
            yield response.follow(next_page, callback=self.parse)
```

Настройка на спайдър

Във файлът settings.py можете да добавите всякакви настройки свързани със спайдъра. Някой неща, които биха били полезни за упражнението:

```
FEED_EXPORT_ENCODING = 'utf-8'

# Example usage of default ImagesPipeline
# IMAGES_URLS_FIELD = 'image_url'
# ITEM_PIPELINES = {'scrapy.pipelines.images.ImagesPipeline': 1}

ITEM_PIPELINES = {'ITEM_CLASS1': PRIORITY, 'ITEM_CLASS2':PRIORITY2}

IMAGES_STORE = './assets/images'
```

Стартиране на спайдър

За да стартирате вашия спайдър трябва да изпълните следната команда от началната директория на вашия проект.

```
scrapy crawl YOUR_SPIDER_NAME
```

Ако искате да запишете резултата във файл, то трябва да добавите опция `-o`. Спрямо разширението на файла задавата и
типа на експоритрания файл:

```
scrapy crawl YOUR_SPIDER_NAME  -o assets/faculty-staff.json
```

Наличните формати можете да намерите
[тук (https://docs.scrapy.org/en/latest/topics/feed-exports.html#topics-feed-exports)](https://docs.scrapy.org/en/latest/topics/feed-exports.html#topics-feed-exports).