Web scraping with Scrapy

Tomáš Bartek, Pycon.cz 2019

Goals of the workshop

1) write own spider

2) have a good idea how Scrapy works

Plan

- Scrapy installation (docker/direct installation)
- Scrapy architecture
- XPath
- Quotation spider - http://quotes.toscrape.com/

Scrapy installation

- Github repo:

  https://github.com/itsx/scrapy-workshop-pycon2019

```
git clone https://github.com/itsx/scrapy-workshop-pycon2019.git
cd scrapy-workshop-pycon2019
sudo docker-compose up -d
sudo docker-compose ps
sudo docker exec -it scrapy bash

# inside running container
scrapy --version
```

Scrapy installation (direct install)

- https://docs.scrapy.org/en/latest/intro/install.html

Scrapy scope

- **Requests** – just requests
- **Beautiful Soup** – extraction of html
- **Scrapy** (since 2009) – requests, extraction, and more ...

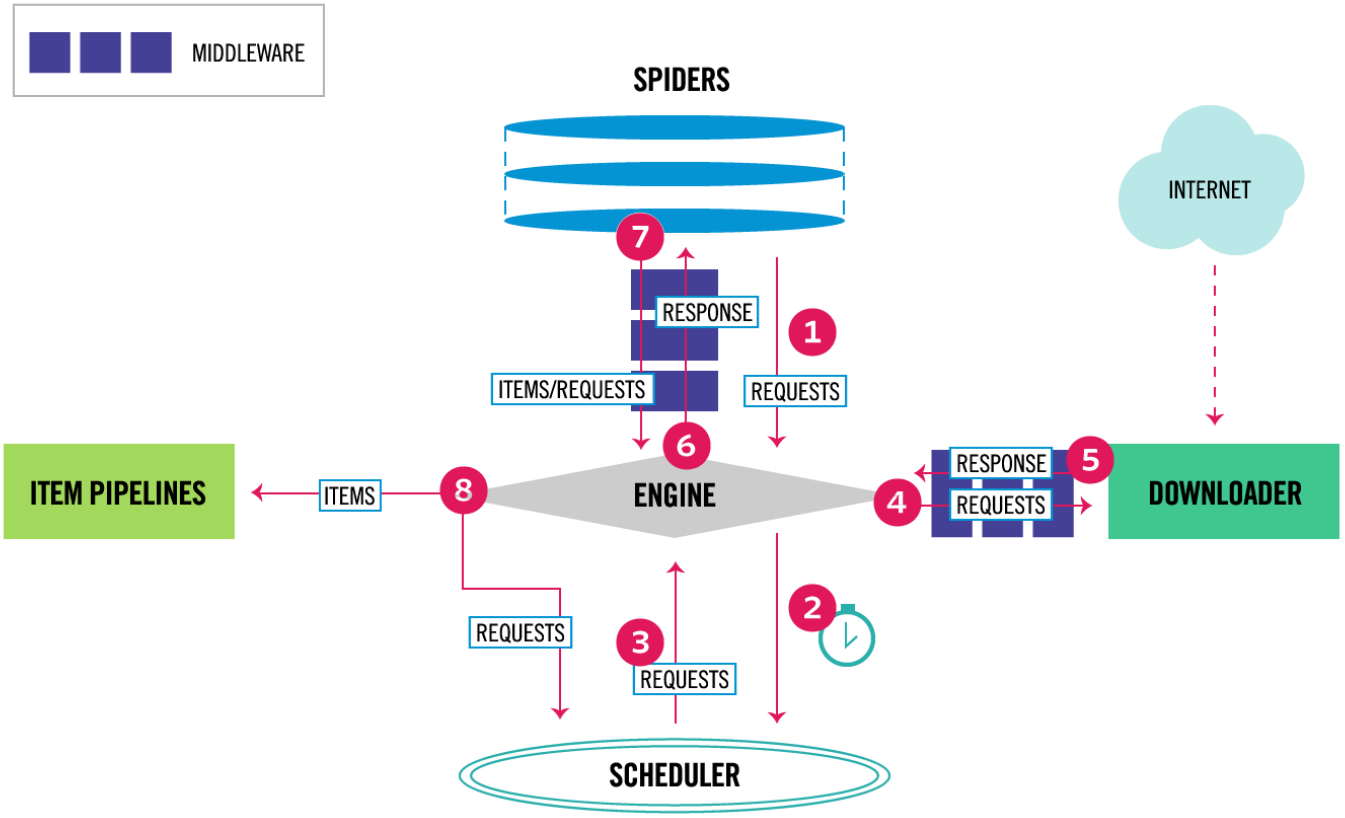
Request

```
r = requests.get('https://api.github.com/user', auth=('user', 'pass'))
r.status_code
r.headers['content-type']
```

Beautiful Soup

```
soup = BeautifulSoup(html_doc, 'html.parser')
soup.title
soup.find_all('a')
```

Scrapy architecture



https://docs.scrapy.org/en/latest/topics/architecture.html

Why Scrapy?

- Open source: https://scrapy.org/
- Handles complete scraping process
- User can focus only on his domain specific code
- Very fast (asynchronous http requests)
- Simple
- Extensible
- Huge community
- Can be deployed to a cloud (scrapinghub)

Scrapy architecture – spiders

- Place for domain (web page) specific code
- Basic spider:

  - **scrapy.Spider**
- Generic spiders:

  - **CrawlSpider**
- def parse(self, response):

Quotes spider

- http://quotes.toscrape.com/
- scrapy genspider toscrape-xpath quotes.toscrape.com
- scrapy shell quotes.toscrape.com
- Use devtool in a browser – F12 key
- scrapy crawl toscrape-xpath -o items.json --output-format=json

Extraction from html

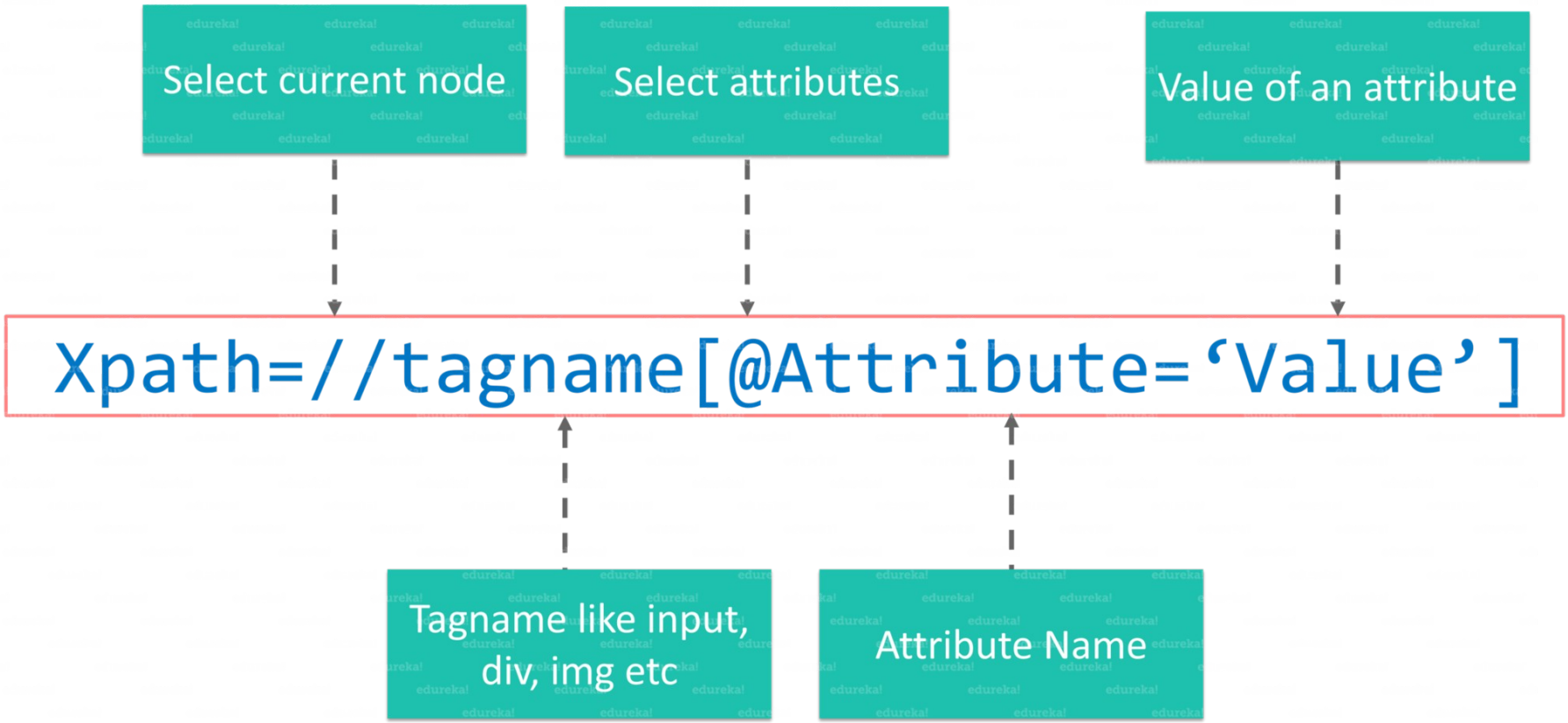
- Both **Css selectors** and **XPaths**
- Css selectors selects nodes according html structure
- Xpaths can select node also according values

  (for example value of a html link)

XPath basics

- A query language for selecting nodes from an XML (and HTML) document
- Select nodes on a specific xml path

XPath anatomy - simplified



https://www.edureka.co/blog/xpath-in-selenium/

XPath selectors

```
#If we want to get html node
response.xpath("/html").extract()
#If we want to get body node, which is the child of html node
response.xpath("/html/body").extract()
#If you want to get all div descendant of this html
response.xpath("/html//div").extract()
#we can also drill down without having to start with /html, this expression would extract all div nodes
response.xpath("//div").extract()
```

Resources - Scrapy

- Scrapinghub:

  https://github.com/scrapinghub/sample-projects
- Scrapinghub:

  https://github.com/scrapinghub
- https://docs.scrapy.org/en/latest/
- https://scrapy.org/community/
- https://blog.scrapinghub.com/

Resources - XPath

- Nice tutorial:

  http://www.zvon.org/comp/r/tut-XPath_1.html#Pages~List_of_XPaths